Robust Spline Path Following for Redundant Mechanical Systems

by

Rajan Gill

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, 2015

© Rajan Gill 2015

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

Path following controllers make the output of a control system approach and traverse a pre-specified path with no a priori time-parametrization. The first part of the thesis implements a path following controller for a simple class of paths, based on transverse feedback linearization (TFL), which guarantees invariance of the path to be followed. The coordinate and feedback transformation employed allows one to easily design control laws to generate arbitrary desired motions on the path for the closed-loop system. The approach is applied to an uncertain and simplified model of a fully actuated robot manipulator for which none of the dynamic parameters are measured. The controller is made robust to modelling uncertainties using Lyapunov redesign. The experimental results show a substantial improvement when using the robust controller for path following versus standard state feedback.

In the second part of the thesis, the class of paths and systems considered are extended. We present a method for path following control design applicable to framed curves generated by spline interpolating waypoints in the workspace of kinematically redundant mechanical systems. The class of admissible paths include self-intersecting curves. Kinematic redundancies of the system are resolved by designing controllers that solve a suitably defined constrained quadratic optimization problem that can be easily tuned by the designer to achieve various desired poses. The class of redundant systems considered include mobile manipulators for a large class of wheeled ground vehicles. The result is a path following controller that simultaneously controls the manipulator and mobile base, without any trajectory planning performed on the mobile base. The approach is experimentally verified using the robust controller developed in the first part of the thesis on a 4-degree-of-freedom (4DOF) redundant manipulator and a mobile manipulator system with a differential drive base.

Acknowledgements

I am grateful to have been supervised by Christopher Nielsen and Dana Kulic. They have provided me with profound guidance over the past couple of years. I greatly appreciate their support in not only this research, but with my life in general. I have learned so much under their guidance, and I could not have asked for a better arrangement.

I am also thankful for being surrounded by the many passionate people who have helped me on this journey. I am particularly grateful to Steven Waslander, Kaan Erkorkmaz and Dan Davison, for their support and advice. They have influenced me positively on a personal and intellectual level.

I appreciate all the love and support of my friends and family. This thesis would not have been possible without them.

Dedication

To my family and friends.

Table of Contents

List of Figures

Chapter 1

Introduction

The problem of requiring a system to follow a desired motion can be broadly classified as either a trajectory tracking or path following. As previously suggested in [46], trajectory tracking consists of tracking a curve with an assigned time parametrization.

Hauser and Hindman provided an intuitive example on the notion of path following (also known as manoeuvre regulation) in [46]. Consider an automobile travelling on varying altitude terrain. Now suppose that as the car travels uphill, it cannot sustain a pre-defined desired speed due to its low torque engine. Once it starts going downhill, the driver could force the vehicle to travel faster than the pre-defined speed in order to catch up to where it otherwise would have been had it not been for the slow climb. If the road bends to the left on the descent, the driver, in an effort to make up for lost time, can cut the road and go off-road. However this approach may not be appropriate, since leaving the road and travelling fast in a vehicle can be dangerous, or it may even be infeasible for the vehicle's engine. A safer approach would ask the driver to maintain the pre-defined speed on the descent while staying on the road.

This example illustrates the difference between the concept of trajectory tracking and path following. In trajectory tracking, the objective is for the system to track a reference point that is changing based on some pre-defined time schedule. In path following, the objective is for the system to stay on the path.

Trajectory Tracking

More precisely, trajectory tracking is concerned with asymptotically driving the output trajectory of a system $y(t)$ to a desired reference signal $y^{\text{ref}}(t)$, i.e., $y(t) \to y^{\text{ref}}(t)$ as $t \to \infty$.

Trajectory tracking has been extensively researched, with a wide variety of tools at the disposal of a control engineer [13,34,39,45,46,49,53,56,62,69,70,84,86,96,97,100,101,105]. Because of this, many path following problems are converted to a trajectory tracking problem [45, 84]. Suppose a robot is to follow a path. Arguably the simplest approach is to parametrize the motion of approaching the path and traversing the path with a particular trajectory, and then have the robot's output match this trajectory at each time step. However this approach may not be suitable for certain applications as it can be unnecessarily demanding or infeasible, as illustrated in the automobile example.

Path Following

Path following or manoeuvre regulation controllers, as previously defined in [46], drive a system's output to a desired path and make the output traverse the path without a pre-specified time parametrization. More precisely, suppose there is a desired path $\mathcal{P}$ in the output space of a system. The goal is to make $y \to \mathcal{P}$ as $t \to \infty$ and if $y(t) \in \mathcal{P}$ for some time t, and is moving tangential to the path, then $y(t) \in \mathcal{P}$ for all future time. These two properties are known as attractiveness and output invariance, respectively. This is a useful property in robotics since, if a disturbance or obstacle is preventing the output of a robot to proceed along the path, the robot will stay on the path until the disturbance is removed [48]. Furthermore, if the robot's output is perturbed so that it leaves the path, the path following controller will drive the output back to the desired path. Notice that there is no specific time-schedule in which the approach to the path should occur. That is, path following controllers are not restricted to a particular trajectory on the desired path.

In general, path following results in smoother convergence to the desired path compared to trajectory tracking, and the control signals are less likely to saturate [65]. It is also shown in [72] for the linear case, and [1] for the nonlinear case, that for non-minimum phase systems, path following controllers remove performance limitations compared to trajectory tracking controllers, which have a lower bound on the achievable $\mathcal{L}_2$-norm of the tracking error.

1.1 Problem Formulation

In this thesis, we design robust path following controllers for redundant mechanical systems with modelling uncertainty to follow splined paths. A nonlinear mechanical system can be

modelled as

$$\begin{aligned} \dot{x}_c &= F_c(x) \\ \dot{x}_v &= F_v(x) + G_v(x)u \end{aligned} \tag{1.1}$$

where $x = (x_c, x_v) \in \mathbb{R}^n$ is the state of the system, $u \in \mathbb{R}^m$ is the vector of control inputs, $F_c(x), F_v(x)$, and $G_v(x)$ are smooth functions. The output of (1.1) takes the form

$$y = h(x) \tag{1.2}$$

where $h : \mathbb{R}^n \to \mathbb{R}^p$ is smooth. The particular redundant systems that this thesis uses in experimental verifications that have the form (1.1) are manipulators with more degrees of freedom than the dimension of the workspace p, and mobile manipulator systems after applying dynamic extension [3, 38, 52], which implies $n/2 > p$. The systems used can be found in Chapter 2.

Suppose a path is given in the output space of the system (1.1) as a regular parametrized curve

$$\sigma : \mathbb{R} \to \mathbb{R}^p \tag{1.3}$$

where the map σ can be a piece-wise continuous function, such as a spline [34]. In Chapter 3, geometric restrictions are imposed on the class of curves considered, as the focus in this chapter is robustifying path following controllers. These restrictions are alleviated in Chapter 4 in order to generalize the class of paths to splines, a powerful class of paths. In Chapter 5, the path following framework developed in the Chapters 3 and 4 is applied to yield a novel mobile manipulator path following controller. In this thesis, the ultimate goal is to design a path following controller to:

PF1 Make the output (1.2) of system (1.1) asymptotically approach the desired spline path, i.e., $y(t) \to \sigma(\mathbb{R})$ as $t \to \infty$, in the presence of modelling uncertainties of the system.

PF2 Make the path output invariant, i.e., if the system is initialized on the path with the velocity vector tangent to the curve, the system will remain on the spline path $\sigma(\mathbb{R})$ $\forall t \geq 0$, in the presence of modelling uncertainties of the system.

PF3 On the path, the system tracks a desired velocity profile.

PF4 Any redundant degrees of freedom are controlled to ensure boundedness of their dynamics, avoid actuator and joint limits and ensure closed-loop stability.

1.2 Literature Review

Numerous approaches have been proposed for approaching and traversing a spline. The most common are based on some type of trajectory tracking approach, although these approaches do not actually satisfy **PF1** to **PF3**. Set stabilization has also been used to directly stabilize a desired path. Application tasks include machine tools, mobile robots, and redundant systems such as mobile manipulators, as in this thesis.

1.2.1 Contour Tracking in CNC Machines

In the machining community, typically the path following requirement is converted to a trajectory tracking problem [34, 69, 91]. The desired path σ is typically generated by linear, circular, or spline interpolation of a set of reference points. The reference points are typically generated by a CAD description of the surface [105]. Spline interpolated curves have the advantage of being continuous to an arbitrary degree, resulting in better machining quality, reduced vibration, and longer tool life [105].

A desired tangential velocity profile η^{ref} is also designed (typically jerk-limited) in order to stay compliant with machining limitations and to smoothly machine the desired contour [34]: $\eta^{\text{ref}} : \mathbb{R} \to \mathbb{R}$. In the case that the argument λ is not the arclength of the curve, as is the case for splines, an interpolator is used to approximate the corresponding trajectory for the argument $\lambda(t)$ to the contour σ, to yield a desired trajectory $\sigma(\lambda(t))$ [34, 69].

In the machining community, one important performance metric is the contour error between the actual and the desired path. To mitigate the contour error, researchers initially focused on perfecting the trajectory tracking capabilities with the use of feed-forward control [36, 62, 86]. Feed-forward controllers are used to increase the bandwidth of control systems, with the effect of having better tracking performance for tracking trajectories. The idea is that perfect trajectory tracking capabilities would imply zero contour error. A significant contribution was made by Tomizuka who developed the Zero-Phase-Error-Tracking-Controller (ZPETC), a feed-forward controller that provides zero phase error from the reference to the output at all frequencies for linear systems, even when the plant is non-minimum phase [101].

Even if the control system can ideally provide perfect tracking for each individual axis of the machine tool, there are additional sources of contour error. These come from mechanical deficiencies such as backlash, cutting effects such as tool deflection and wear, drive dynamics mismatch and other disturbances [62]. In [61], Koren quantifies the steady state contour error in the case of machining linear contours in the presence of friction disturbance

and controller parameter mismatch when using a proportional feedback controller. In [83], the steady state contour error is quantified when machining circular trajectories, showing that this error intuitively depends on the ratio of the radial frequency of the circular trajectory to the bandwidth of the control system.

Contour error mitigation controllers have also been proposed in the machining community. These controllers focus on reducing the contour error instead of only reducing tracking error [21, 23, 24]. They are based on the cross-coupled controller (CCC) proposed by Koren [61], which requires some tuning depending on the system dynamics and the desired contour. However, all CCCs are time-varying for the case of curved contours [99]. They also depend on some approximation of the contour error [62, 91], and thus do not guarantee that the desired path will be invariant.

1.2.2 Path Planning and Spline Tracking in Robotics

Splined paths have also found use in the robotics community as the output of a path planning problem, confirming that path following control of splines, one of the goals of this thesis, is useful. For nonholonmoic mobile robots, the goal of reaching some desired end-point with some given initial point and initial heading, has been formulated as a time-optimal path-planning problem whose result is a spline path for the robot to follow [66]. Other works have used splines as the output of a optimal path-planning problem considering dynamic constraints of ground vehicles [67], robot manipulators [7], and unmanned-aerial-vehicles [54], due to the linearity of the parameters of spline curves and fast mathematical manipulations [7].

In [26, 27, 41], the authors design an approach for generating cubic spline paths for ground vehicles for point-to-point navigation while avoiding obstacles. The approach entails generating the spline from some desired start to end position. If this path collides with obstacles on the map (determined by using Bresenham's pixel algorithm [15]), the intersections of the path and the obstacles are determined, yielding entry and exit collision points. Then a new intermediate waypoint is generated that is on the perpendicular bisector to the line joining the two collision points. The process is repeated for remaining collisions, and a spline is used to define piecewise polynomial curves that pass through the generated Cartesian points. The use of cublic splines also guarantees continuous vehicle heading and steering angle, which is desired for ground vehicles.

Another approach for obstacle avoidance path planning is by performing a particle swarm optimisation, as is done in [89]. Here the output of the optimization is also a cubic spline. They compare their results to other path planning approaches, such as potential

fields and visibility graphs. It was found that the spline based particle swarm approach yields short and smooth plan towards the end goal while avoiding obstacles, whereas the virtual graph approach found the shortest approach but with sharp turns, forcing robots to slow down. The potential fields method found the longest route [89].

Once the spline path has been generated, the robot is to track a trajectory such that it is on the path. In most works, the spline path is also treated as the desired trajectory. In other words, the argument to the generated curve $\sigma(t)$, t, is used as the time parametrization for the spline trajectory [45, 84]. However, the problem with this approach is that the spline is not unit-speed parametrized (since when generating the spline, the argument t cannot represent the arc length along the path since the spline itself hasn't been constructed). This means that a constant tangential velocity profile cannot be achieved along the path, let alone any arbitrary tangential velocity profile along the path. The trajectory is fixed by the *a-priori* path/trajectory generation.

One way to address this is to use the approach in the machining community, where an interpolation is used to design a trajectory for the argument of the spline based on some desired tangential velocity profile (see Section 1.2.1).

Another approach is to generate a trajectory that is path-constrained. This has been done in an optimal fashion, minimizing the total trajectory time, subject to the constraints of the dynamics system, and subject to the output of the dynamic system be on the desired spline [11, 29]. However these approaches rely on dynamic programming and can be computationally expensive [29]. The approach in our thesis requires just that the path be available to the real time controller.

1.2.3 Integrating Motion Planning with Trajectory Tracking

Early work considered how to plan a trajectory to follow a path under the constraints of the dynamic system, such that a realizable trajectory can be generated [13, 49, 70]. However in [97, 100], it was argued that this classical two layered hierarchical approach of path planning and trajectory tracking has disadvantages. Namely, the motion plan cannot be modified in response to a dynamic environment where disturbances can cause a deviation from the path. The authors of [97, 100] then designed a combined controller involving a change of variable from time to the arclength along the path. A time-optimal trajectory is then designed for an equivalent dynamic system that is parametrized by the arclength distance along the path. The resulting motion plan is then decomposed back to the original coordinates, but parametrized by the path variable as opposed to time. However this approach does not guarantee that the desired path will be attractive in the

neighbourhood of the path. The approach is also limited to paths that are parametrized by their arclengths, and also faces a singularity when the tangential velocity command is zero. The approach used in this thesis overcomes these problems, making curves including splines attractive and invariant, without singularities from the tangential velocity command.

Trajectory tracking controllers have also been combined with path following. The papers [46,56,96] project the current system state onto the desired path in real time. Then a tracking controller is used to stabilize this generated reference point. Here they assume some reference trajectory is created in the state space, $x^{\text{ref}}(t) \in \mathbb{R}^n$, and that a trajectory tracking controller has already been designed such that $x \to x^{\text{ref}}(t)$ as $t \to \infty$. They design a controller based on the Lyapunov-like function

$$\min_{\tau \in \mathbb{R}} (1-\lambda)^2 \left\| x - x^{\text{ref}}(\tau) \right\|^2 + \lambda^2 (t-\tau)^2$$

where $\lambda \in \mathbb{R}$ is a design parameter that governs the priority of tracking the original trajectory and getting to the path. The parameter τ that minimizes the above optimization is treated as the reference signal. Similar work is done in [47] for aircraft manoeuvres. However, these approaches do not guarantee that the desired path will be rendered invariant, as the desired parameter of the path will ultimately change over time. They also have been used for simpler class of curves, notably circles and ellipses.

1.2.4 Set stabilization and Virtual Holonomic Constraints

The classical notion of point stabilization is a special case of set stabilization. Similarly, orbital stability of limit cycles is a special case of the stability of sets. Limit cycle stability can be traced back to the work of Poincaré [82], who studied the stability of orbits of celestial objects in the solar system. Later, Van der Pol discovered and analysed limit cycles in nonlinear electrical circuits, calling them "relaxed oscillations" [104].

Moving towards more general sets, Barbashin [10] and Zubov [109] studied the stability of invariant sets for dynamic systems. Their work, along with others that built upon it (see [68,87,88]), characterizes the relationship between set stability and Lyapunov functions.

In terms of stabilizing sets, the authors in [8,9] looked at periodic orbits of unforced nonlinear systems. They then provide conditions on which these orbits can be made stable using control, namely via feedback linearization techniques. Isidori, Nijmeijer and Van der Schaft [52,78] provide complementary results that find the largest controlled invariant sub manifold on which the system output is zero. These results were extended in [77], where the sets were extended to embedded submanifolds of the state space of a dynamic system.

Other approaches for set stabilization have been investigated, such as passivity [33], and sliding mode control [103].

Virtual holonomic constraints can be used to guarantee invariance of a path via set stabilization. A virtual holonomic constraint is a constraint imposed by control on the configuration variables of the dynamic system. In [93], generating and orbital stabilization of periodic solutions is done for nonlinear systems, with application to any mechanical system. The approach guarantees invariance of the chosen virtual holonomic constraint. This methodology is used for orbital stabilization for a Furuta pendulum [94], helicopters [106], and, in the sense of hybrid systems, for the walking/running of bipedal robots [22]. However in our work, the path to be followed is not limited to closed curves in the output space of the system and the dynamics on the curve are not fixed.

In [76] the authors proposed an approach in which an arbitrary non-periodic path in the output space can be made attractive and invariant using transverse feedback linzearization (TFL). This idea was further extended in [48] by providing conditions for which desired motions can be achieved on the path; as opposed to orbital stabilization where the dynamics along the closed curve in the state space are fixed. The approach in [48] simplifies controller design and decomposes the path following design problem into two independent sub-problems: staying on the path and moving along the path. However, in [48], perfect knowledge of the dynamic parameters of the system was assumed, performance was demonstrated only for a linear, decoupled system, and the path is restricted to a special case of simple curves such as circles.

1.2.5 Redundancy Resolution and Mobile Manipulators

Resolving redundancies of a system in the context of following a desired path also shares close affinity with this thesis (**PF4**). In robotics, a kinematically redundant system is one that possesses more degrees of freedom than that required to execute a given task. A manipulator mounted on a moving vehicle is a system that is kinematically redundant, due to the extra degrees of freedom added by the vehicle, either the orientation and position of the manipulator or the vehicle itself can be changed to achieve some desired end-effector position and orientation [6].

There has been extensive research for redundancy resolution for manipulators in the context of output point stabilization and trajectory tracking. Given some desired end-effector velocity trajectory, the corresponding joint-space trajectory can be generated by using the pseudoinverse of the Jacobian of the robot manipulator. The null space of the Jacobian can be used to satisfy optimality in some static cost function over the joint

positions and velocities [50]. However, this approach only indirectly effects the applied torques, and there is no direct way of incorporating joint torque limits into the kinematic redundancy resolution [50]. Khatib was one of the first researchers to extend the kinematic redundancy resolution approach to the acceleration level, thereby being able to incorporate the joint torques [58] directly into the resolution scheme. He showed that by analysing the dynamics of a manipulator in the task-space of the system, a certain choice of end-effector force can be made to stabilize the output of the system. To maintain asymptotic stability in the joint-space of the redundant system, an additional choice of joint torques needs to be applied, that corresponds to a null operational force vector, which act as dissipative joint forces. The redundancy resolution mechanism is shown to work only for a specific class of mechanical systems in [58], namely the torque-input model of robot manipulators. This approach is also used in the context of trajectory tracking or output point stabilization of robot manipulators. The redundancy resolution approach we develop in this thesis is used in the context of set stabilization and path following. Furthermore, our approach resolves the redundancies at the motor voltage level, while not having to assume that the dynamics of an inner torque controller can be ignored.

The primary objective of a mobile manipulator tracing a contour in its output space in household-service tasks [12, 81], maintenance/inspection of high-voltage transmission lines [18], material handling in construction [31], large-scale manufacturing [44] and space exploration [92] is to accurately traverse a path. The majority of applications and experiments of mobile manipulators do not simultaneously control the manipulator and the mobile base. That is, they control the manipulator and the mobile base in a sequential, not parallel (as is done in this thesis), manner [17, 73, 81, 107]. Mobile manipulator systems can also be modelled as a single redundant system [108]. Previous works on trajectory tracking have tackled the trajectory tracking problem by resolving the redundancies at the kinematic level, but, unlike in this thesis, a trajectory for the end-effector as well as the mobile base is specified [51, 92]. Khatib extended the dynamic-level redundancy resolution developed for torque-input model of robot manipulators [58] to holonomic mobile platforms [59], whereas the approach in this paper can also be applied directly to combined motor-input manipulator mounted on nonholonomic ground vehicles. In [108], the authors look at the coordinated mobile manipulator problem, where, unlike in this paper, the manipulator dynamics are neglected, to track a reference trajectory in the workspace of the mobile platform while maximizing the manipulability measure.

1.3 Contributions and Organization

As outlined in the problem formulation (Section 1.1), the overall goal of this thesis is to design and implement spline path following controllers for redundant systems with modelling uncertainties. Each chapter of this thesis progressively builds up methodology to achieve this objective, by addressing various aspects needed for path following control.

First, the work of [48] is extended in Chapter 3 by making the controller robust to modelling inaccuracies. The approach is demonstrated experimentally on a nonlinear, coupled manipulator system (Chapter 2). The work in this chapter will partly address **PF1** to **PF3**.

In Chapter 4, we extend the class of allowable paths of [48], allowing self-intersecting paths and curves that are generated through spline interpolation of given waypoints. Spline interpolated paths are a useful class of paths in the robotics communities (see Section 1.2.2). This is accomplished by designing a numerical algorithm and by using a systematic approach to generate zero level set representations of the curves. Another contribution of this work is extending the class of systems of [48] to mechanical systems with kinematic redundancies, by employing a novel redundancy resolution technique at the dynamic level, yielding bounded internal dynamics. The approach in this chapter is to resolve the redundancies directly in the context of path following via feedback linearization, unlike in [58], and is thus applicable to general mechanical systems, which may include a dynamic map between the system input and the output forces. To the best of the author's knowledge, redundancy resolution in the context of feedback linearization is an unsolved problem. We show that, under the assumption that the redundant system has viscous friction on each degree of freedom, the control law obtained by solving a static, constrained, quadratic optimization problem yields bounded closed-loop internal dynamics. The control strategy is then validated on a non-trivial redundant platform with significant modelling uncertainties using the robust controller developed in Chapter 3. This controller then fully achieves **PF1** to **PF3**, and **PF4** for a specific class of redundant systems, namely redundant Euler-Lagrange systems.

In Chapter 5, we apply the path following control approach from Chapter 4 to general mobile manipulator systems. The approach in this chapter also uses dynamic extension [3, 38, 52] to convert the dynamics of the mobile manipulator system that are tangent and transversal to the path to be linear. The remaining dynamics, that are redundant to following the path, will show up as internal dynamics of the entire closed-loop system. The redundancy resolution technique developed in Chapter 4 are used at the dynamic level, and it is experimentally shown to yield bounded internal dynamics, while maintaining a

preferred manipulator posture. This scheme can easily be tuned by the designer to achieve various desired poses. The result is an automatic unified path following controller that compensates for the dynamics of the mobile manipulator system, controlling the mobile base and manipulator simultaneously, rendering a desired path in the output space of the system to be attractive and invariant. There is no trajectory that needs to be planned and tracked by the manipulator or the mobile base, and the coordination between the two is done automatically by our proposed path following controller. The non-trivial experimental verification on a mobile platform (Chapter 2) including robustification (from Chapter 3) is also included. This approach also achieves **PF1** to **PF3**, as well as **PF4** for a broader class of redundant mechanical systems than in Chapter 4.

1.4 Notation

Let $\langle x,\ y\rangle$ denote the inner product of vectors x and y in $\mathbb{R}^n$. The Euclidean norm of a vector and induced matrix norm are both denoted by $\|\cdot\|$. The notation $s \circ h : A \to C$ represents the composition of maps $s : B \to C$ and $h : A \to B$. If $f : A \to B$ is a function, the image of f is written $f(A) \subseteq B$. The ith element of a vector x is denoted x_i, and the row i to j and column k to l submatrix of A is denoted as $A_{i:j,k:l}$. The symbol $\coloneqq$ means equal by definition. Given a C^1 mapping $\phi : \mathbb{R}^n \to \mathbb{R}^m$ let $\mathrm{d}\phi_x$ be its Jacobian evaluated at $x \in \mathbb{R}^n$. If $f,\ g : \mathbb{R}^n \to \mathbb{R}^n$ are smooth vector fields we use the following standard notation for iterated Lie derivatives $L_f^0\phi \coloneqq \phi$, $L_f^k\phi \coloneqq L_f(L_f^{k-1}\phi) = \langle \mathrm{d}L_f^{k-1}\phi_x, f(x)\rangle$, $L_gL_f\phi \coloneqq L_g(L_f\phi) = \langle \mathrm{d}L_f\phi_x, g(x)\rangle$. Finally, $0_{n\times m}$ is the n by m zero matrix and $I_{n\times m}$ is a n by m matrix of zeros with ones on its principal diagonal.

Chapter 2

Mechanical System Modelling

This chapter gives a brief overview of the modelling procedure for a broad class of nonlinear systems. We further model the specific experimental platforms used for the experimental validation in Chapers 3 to 5, namely, a 4-Degree-of-Freedom (DOF) manipulator actuated by linear and revolute actuators in Section 2.1.1, and a differential drive mobile manipulator system in Section 2.2.3

2.1 Euler-Lagrange Systems

In Chapters 3 and 4, we study Euler-Lagrange systems with an N-dimensional configuration space contained in $\mathbb{R}^N$ and N configuration forces $\tau \in \mathbb{R}^N$. The model is given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial \mathscr{L}}{\partial \dot{q}} - \frac{\partial \mathscr{L}}{\partial q} = \tau$$

where $\mathscr{L}(q, \dot{q})$ is the Lagrangian function. We assume that $\mathscr{L}$ is smooth and has the form $\mathcal{L}(q, \dot{q}) = \mathscr{K}(q, \dot{q}) - \mathscr{P}(q)$ where $\mathscr{K}(q, \dot{q}) = (1/2)\dot{q}^\top D(q)\dot{q}$ is the system's kinetic energy and $\mathscr{P} : \mathbb{R}^N \to \mathbb{R}$ is the system's potential energy. The functions $\mathscr{K}$ and $\mathscr{P}$ depend on the robot's mechanical parameters such as the link mass, center of mass, and inertia for each link. The inertia matrix $D(q)$ is positive definite for all q.

The system can be rewritten in the standard vector form

$$D(q)\ddot{q} + \bar{C}(q, \dot{q})\dot{q} + G(q) = \tau \tag{2.1}$$

where $\bar{C}(q,\dot{q}) \in \mathbb{R}^{N\times N}$ represents the centripetal and Coriolis terms, and $G(q) \coloneqq (\mathrm{d}\mathscr{P}_q)^\top = \nabla\mathscr{P}(q) \in \mathbb{R}^N$ represents the gravitation effects [98].

When the dynamics of the motor model are included, the Euler-Lagrange system (2.1) can be augmented to the form

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = A(q)u_m \tag{2.2}$$

where $u_m \in \mathbb{R}^N$ is the system control input (usually motor voltages [98]). The term $M : \mathbb{R}^N \to \mathbb{R}^{N\times N}$ is the positive-definite inertia matrix that includes motor inertias, $C : \mathbb{R}^{2N} \to \mathbb{R}^N$ represents centripetal, Coriolis, and motor damping terms, and $A : \mathbb{R}^N \to \mathbb{R}^{N\times N}$ is the input matrix and is assumed to be nonsingular for all q [98].

Defining $x_c \coloneqq q$, $x_v \coloneqq \dot{q}$, $n \coloneqq 2N$, and $x \coloneqq \mathrm{col}\,(x_c, x_v) \in \mathbb{R}^N \times \mathbb{R}^N$, and $u \coloneqq u_m$, we can express (2.2) in state model form

$$\dot{x} = f(x) + g(x)u \coloneqq \begin{bmatrix} x_v \\ f_v(x) \end{bmatrix} + \begin{bmatrix} 0 \\ g_v(x_c) \end{bmatrix} u, \tag{2.3}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ are assumed to be smooth, $g_v(x_c) \coloneqq D^{-1}(x_c)A(x_c)$ and $f_v(x) \coloneqq -D^{-1}(x_c)\left(C(x)x_v + G(x_c)\right)$.

Motivated by the task space of a robotic manipulator, we take the output of system (2.3) to be

$$y = h(q), \qquad \frac{\partial h(q)}{\partial \dot{q}} = 0 \tag{2.4}$$

where $h : \mathbb{R}^N \to \mathbb{R}^P$ is smooth, and P is the dimension of the output space of system (2.2). In addition, let $J : \mathbb{R}^N \to \mathbb{R}^{P\times N}, q \mapsto \frac{\partial h(q)}{\partial q}$ represent the output Jacobian [98]. The systems considered in this thesis are redundant in the following sense

Definition 2.1.1. *System (2.3),(2.4) is kinematically* ***redundant*** *if $N > P$.*

2.1.1 Modelling of the CPM

The test platform is a Clearpath Robotic Manipulator (CPM) (see Figure 2.1) which is a four degree of freedom, fully actuated system. The shoulder, elbow, and wrist links are actuated by DC linear actuators.

The link center of mass and inertias are constant in the Denavit-Hartenberg (DH) coordinate frame attached to each link [98]. However, the linear actuator center of mass and inertias are not constant in the corresponding DH frame, as the linear actuator extension

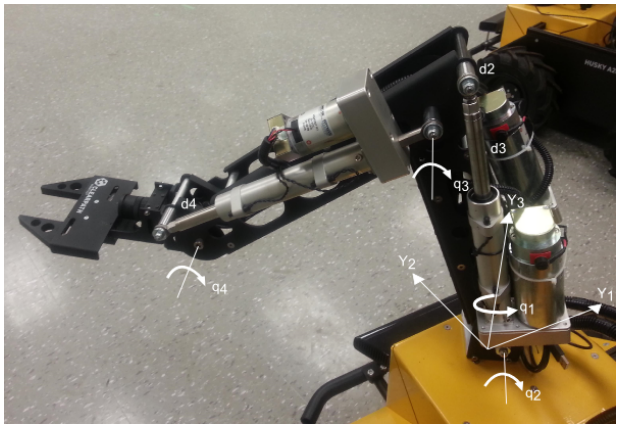

Figure 2.1: The Clearpath Manipulator (CPM) with the joint coordinates and output space labelled.

d_i and orientation vary with the joint angles. This varying center of mass and inertia of the linear actuators significantly complicates the dynamic model. To simplify the model an average is taken over the CPM workspace to yield constant parameters in each DH frame. In particular, let $d_i \in \mathbb{R}, i \in \{2,3,4\}$ be the linear actuator extension for the ith joint, and $I_{LA_i} : \mathbb{R} \to \mathbb{R}^{3\times3}, m_{c_{LA_i}} : \mathbb{R} \to \mathbb{R}^3$ be the inertia tensor and center of mass location for the ith actuator, respectively, with respect to the DH frame of the link the actuator is attached to. Then the average is taken as

$$\overline{I}_{LA_i} \coloneqq \frac{1}{d_i^{max}} \int_0^{d_i^{max}} I_{LA_i}(d_i)\mathrm{d}d_i$$

$$\overline{m}_{c_{LA_i}} \coloneqq \frac{1}{d_i^{max}} \int_0^{d_i^{max}} m_{c_{LA_i}}(d_i)\mathrm{d}d_i$$

where d_i^{max} is the maximum extension of linear actuator i. These values are used as constant approximations of the center of mass and inertia tensors of the actuators in their respective DH frame. All of the CPM mechanical parameters are expected to be inaccurate because they are generated by SolidWorks© and no system identifications were performed.

The waist actuator is a standard DC motor passing through a worm gearbox. Let the angle of this motor be q_{m_1}. The linear actuators consists of a rotational brushed DC motor

and a leadscrew. Based on a standard DC motor model, the motor dynamics become

$$\begin{aligned} J_{m_1}\ddot{q}_{m_1} + B_1\dot{q}_{m_1} &= K_1 u_1 - \frac{\tau_1}{r_1} \\ J_{m_i}\ddot{d}_i + B_i\dot{d}_i &= K_i u_i - \frac{F_i}{r_i\rho_i},\ i \in \{2,3,4\} \end{aligned} \tag{2.5}$$

where J_{m_i}, B_i, K_i are the motor parameters, r_i are the gearbox ratios, F_i are the linear actuator load forces, ρ_i comprises the lead screw parameters [79], and u_i are the motor voltages. The parameters were obtained from the actuator data sheets and were not physically measured. To convert the waist angle q_1 to the motor angle q_{m_1}, note that $q_{m_1} = r_1 q_1$. An invertible map $l_i : \mathbb{R} \to \mathbb{R}, i \in \{2,3,4\}$, is used to convert the joint angles q_i to the linear actuator extension d_i:

$$d_i = l_i(q_i) = \sqrt{a_i^2 + h_i^2 - 2a_i h_i \cos(q_i + \beta_i)} - L_i. \tag{2.6}$$

Here, for $i \in \{2,3,4\}$, a_i are the link lengths, h_i, β_i are parameters that depend on the motor mounting scheme, and L_i are the lengths of the linear actuator housing. The joint torques $\tau_i, i \in \{2,3,4\}$ in (2.1) are related to the linear actuator force F_i by a map $\zeta_i : \mathbb{R} \to \mathbb{R}$,

$$\zeta(q_i) := \frac{\sin(q_i + \beta_i)}{L_i + d_i} h_i,$$

such that $\tau_i = F_i \zeta_i(q_i), i \in \{2,3,4\}$. Using these maps and the chain rule, (2.1) and (2.5) become

$$\left.\begin{aligned} \left[D(q)\ddot{q} + \bar{C}(q,\dot{q})\dot{q} + G(q)\right]_1 &= \tau_1 \\ r_1^2 J_{m_1}\ddot{q}_1 + r_1^2 B_1 \dot{q}_1 &= r_1 K_1 u_1 - \tau_1 \\ \left[D(q)\ddot{q} + \bar{C}(q,\dot{q})\dot{q} + G(q)\right]_i &= F_i\zeta_i(q_i) \\ r_i\rho_i J_{m_i}\frac{\partial^2 l_i}{\partial q_i^2}(q_i)\ddot{q}_i + r_i\rho_i\left(B_i\frac{\partial l_i}{\partial q_i}(q_i) J_{m_i}\frac{\partial^2 l_i}{\partial q_i^2}(q_i)\right)\dot{q}_i &= r_i\rho_i K_i u_i - F_i \end{aligned}\right\} i \in \{2,3,4\}$$

which can be written compactly in form (2.2)

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = A(q)u. \tag{2.7}$$

The dynamic model of the robot neglects friction and uses crude estimates of the parameters. This model uncertainty is addressed by the controller design in the subsequent chapter.

2.2 Mobile Manipulator Systems

We consider a fully actuated manipulator mounted on a mobile ground vehicle. For simplicity, the combined systems are assumed to be decoupled. This crude model also helps demonstrate that the proposed technique is somewhat tolerant to modelling errors. Essentially, the combined dynamic model is the manipulator dynamics (2.2) concatenated with the vehicle dynamics. This is a valid system model assumption for general mobile manipulator systems [25] and for our experimental platform (Section 5.3), and in general, manipulator dynamics are relatively fast compared to that of the vehicle [59].

The output of system (2.2), the task space of the manipulator (2.4), is given in local frame $\mathcal{B}$ attached to the vehicle base (see Figure 2.3). We assume, without loss of generality, that $P \geq 3$ and the first 3 components of h correspond to the Euclidean position of the end-effector in the local frame $\mathcal{B}$ (Figure 2.3). Any additional rows ($P > 3$) of h represent orientations of the end-effector in the world frame. For example, if $P = 4$, the fourth row of h could represent the end-effector angle with respect to the horizontal ground plane [80].

2.2.1 Vehicle subsystem

The position $x^b \in \mathbb{R}^2$ and orientation $\theta \in \mathbb{R}$ of frame $\mathcal{B}$ in the inertial frame $\mathcal{O}$ (see Figure 2.3) are governed by the vehicle's dynamics. A general vehicle kinematic model was derived in [30] for a large class of wheel configurations, including castor wheels, steering wheels, fixed wheels, and/or Swedish wheels, and is given as

$$\begin{bmatrix} \dot{x}^b \\ \dot{\theta} \end{bmatrix} = R(\theta)\Sigma(\sigma)\gamma_b \tag{2.8}$$

$$\dot{\sigma} = \gamma_s \tag{2.9}$$

where $R(\theta)$ is the rotation matrix

$$R(\theta) := \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.10}$$

$\Sigma : \mathbb{R}^{\delta_s} \to \mathbb{R}^{3\times\delta_b}$ is a full rank matrix [30, pg. 283] that depends on the steering angle $\sigma \in \mathbb{R}^{\delta_s}$ of the steering wheels, $\delta_s \in \{0, 1, 2\}$ is the *degree of steerability* [30, pg. 273], $\delta_b \in \{1, 2, 3\}$ is the *degree of mobility* of the mobile base [30, pg. 272], and $\gamma_b \in \mathbb{R}^{\delta_b}, \gamma_s \in \mathbb{R}^{\delta_s}$ are the inputs to the vehicle. The degrees of mobility and steerability δ_b, δ_s depend on the

design of the vehicle and satisfy the constraint $\delta_b + \delta_s \in \{2, 3\}$. The kinematic model (2.8) encompasses wheeled-ground vehicle designs including car-like vehicles, unicycles (see Sec. 5.3), and omnidirectional vehicles [30, Chapter 7.2]. If the vehicle has no steering wheels, such as those found on differential drive robots (see Sec. 5.3), then $\delta_s = 0$ and Σ is a constant matrix.

Example 1. (Car-like base) Consider a car-like vehicle with 2 front steering wheels and 2 rear fixed wheels (see Figure 2.2). For such a system, $\delta_s = 1$ and $\delta_b = 1$, and the general kinematic model (2.8) reduces to [3]

$$\dot{x}_1^b = \gamma_b \cos(\theta) \tag{2.11}$$
$$\dot{x}_1^b = \gamma_b \sin(\theta) \tag{2.12}$$
$$\dot{\theta} = \frac{\gamma_b}{l} \tan(\sigma) \tag{2.13}$$
$$\dot{\sigma} = \gamma_s \tag{2.14}$$

where l is the length of the vehicle.

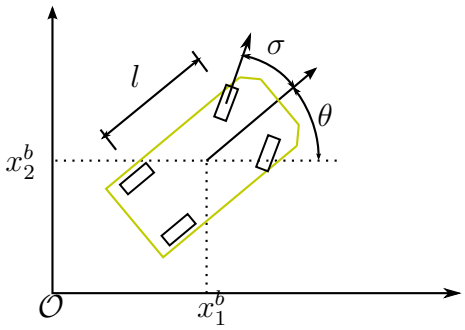

Figure 2.2: Car-like robot.

△

2.2.2 Output Model

The output is chosen to be manipulator position $h(q)$ in the inertial frame as follows

$$y \coloneqq H(q, x^b, \theta) = \begin{bmatrix} R(\theta)_{1:2,1:2} & 0_{2\times(P-2)} \\ 0_{(p-2)\times 2} & I_{(p-2)\times(P-2)} \end{bmatrix} h(q) + \begin{bmatrix} x_1^b \\ x_2^b \\ 0_{(p-2)\times 1} \end{bmatrix}. \tag{2.15}$$

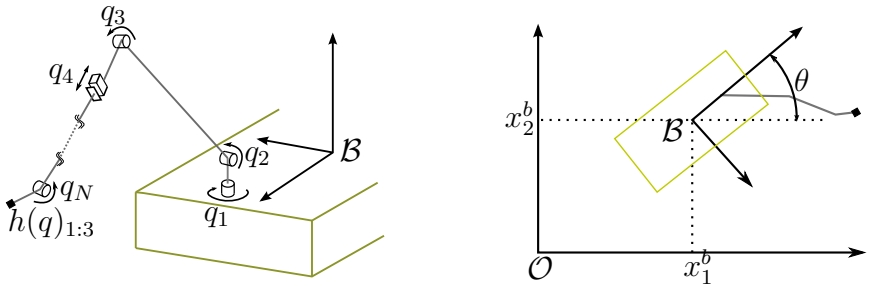

Figure 2.3: Schematic of the mobile manipulator. The end-effector output $h(q)$ is expressed in the local-frame $\mathcal{M}$ attached to the bottom of the mobile base. The position and orientation of the local-frame in the inertial frame $\mathcal{O}$ are determined by x^b, θ.

where $P \geq p \geq 2$ is the dimension of the output space. Recall that $P \geq 3$, with the first three coordinates of $h(q)$ representing the Euclidean position in the local frame $\mathcal{B}$, and any additional coordinates representing the orientation in the body frame. If $p = 2$, then only the planar position of the end-effector in the inertial frame is of concern. If both $P > 3$ and $p > 3$, then a component of the end-effector orientation is also to be specified.

Assumption 2.2.1 (Dexterity)**.** *The mobile manipulator satisfies $N + \delta_b \geq p$ (see Remark 5.2.1), i.e., the system has enough degrees of freedom to follow arbitrary paths in the reachable workspace in $\mathbb{R}^p$.* □

The combined manipulator and mobile base dynamics, including the state space formulation, are discussed in greater detail in Chapter 5.

2.2.3 Modelling of the Clearpath Husky

The mobile manipulator system is a platform designed by Clearpath Robotics (see Figure 2.4). The mobile platform runs a low level control loop that controls the vehicle's heading rate and tangential velocity. The vehicle has no steering wheels. Thus, for this system $\delta_s = 0$ and $\delta_b = 2$ and the general kinematic model (2.8) reduces to the standard unicycle equations [30]

$$\dot{x}_1^b = \gamma_{b_1} \cos(\theta) \tag{2.16}$$

$$\dot{x}_2^b = \gamma_{b_1} \sin(\theta) \tag{2.17}$$

$$\dot{\theta} = \gamma_{b_2}. \tag{2.18}$$

The output space is taken to be the 3-dimensional Euclidian space, that is $p = 3$.

Figure 2.4: Clearpath manipulator mounted on a Clearpath A200 mobile platform.

Chapter 3

Robust Path Following

This chapter implements a robust path following controller, based on transverse feedback linearization (TFL), which guarantees invariance of the path to be followed. Portions of this chapter are presented in our paper [42]. A review of TFL [48] is provided in Section 3.1. The class of paths are limited and are extended in the next chapter. The coordinate and feedback transformation employed allows one to easily design control laws to generate arbitrary desired motions on the path for the closed-loop system. Redundancy resolution is ignored by adding additional constraints to the system, and a more general redundancy resolution is developed in the next chapter. The approach is applied to the CPM, and the controller is made robust to modelling uncertainties using Lyapunov redesign in Section 3.2. The experimental results (Section 3.3) show a substantial improvement when using the robust controller for path following versus standard state feedback.

3.1 Path following for mechanical systems

In this section we review the path following control design methodology from [48] which applies to Euler-Lagrange systems. In [48] geometric restrictions on the allowable class of curves are imposed:

Assumption 3.1.1. *The path $\sigma(\mathbb{R})$ is a 1-dimensional embedded submanifold of $\mathbb{R}^P$.*

Assumption 3.1.2. *There exists a smooth map $s : \mathbb{R}^P \to \mathbb{R}^{P-1}$ such that $\operatorname{rank}(\frac{\partial s}{\partial y}) = P-1$ on $s^{-1}(0)$ and $\sigma(\mathbb{R}) = s^{-1}(0)$. Let $\mathcal{P} \coloneqq s^{-1}(0)$. Moreover, the lift of $\mathcal{P}$ to $\mathbb{R}^n$*

$$\Gamma \coloneqq (s \circ h)^{-1}(0) = \{x \in \mathbb{R}^n : s(h(x)) = 0\} \tag{3.1}$$

is a submanifold of $\mathbb{R}^n$.

Assumption 3.1.1 ensures that the path has no self-intersections or corners. Assumption 3.1.2 requires that the entire path be represented as the zero level set of a smooth function $s : \mathbb{R}^P \to \mathbb{R}^{P-1}$ in the output space of the system (2.4), and that its Jacobian is full rank at each point of the curve $\mathcal{P}$. Locally this is always possible if Assumption 3.1.1 holds [48].

Making the state x of (2.3) approach the set (3.1) is equivalent to making the system output (2.4) approach its desired path $\mathcal{P} = s^{-1}(0)$. Unfortunately, the set Γ is generally not invariant under the dynamics of (2.3) and as a result cannot be stabilized. Therefore, rather than stabilizing Γ, we instead stabilize the largest controlled invariant set contained in Γ. This set, denoted $\Gamma^\star$, is called the path following manifold of (2.3) with respect to the path $\mathcal{P}$. The path following manifold consists of all the motions of the system (2.3) for which the output (2.4) remains on the curve $\mathcal{P}$ by suitable choice of control input u. Let the dimension of $\Gamma^\star$ be $n^\star > 0$. Equivalently $\Gamma^\star$ can be thought of as the zero dynamics manifold [52] of (2.3) with output $\gamma(x) \coloneqq s \circ h(x)$. The existence of $\Gamma^\star$ is assured as long as the path $\mathcal{P}$ is feasible for the system (2.3) [48].

If the virtual output $\gamma(x)$ yields a well-defined relative degree at some point $x^\star$ on $\Gamma^\star$ then, locally, input-output feedback linearization for non-square systems can be performed at $x^\star$. In particular, this guarantees the existence of a coordinate transformation $T : U \to T(U)$, $x \mapsto (\eta, \xi)$ where U is a neighbourhood of $x^\star$, and a regular feedback transformation[1] $u = \alpha(x) + \beta(x)v$ such that $T(\Gamma^\star \cap U) = \{(\eta, \xi) : \xi = 0\}$. Defining $v \coloneqq \mathrm{col}(v_\xi, v^\eta)$, in the new coordinates, system (2.3) becomes

$$\begin{aligned} \dot{\eta} &= f^0(\eta, \xi) + g^\xi(\eta, \xi)v_\xi + g^\eta(\eta, \xi)v^\eta \\ \dot{\xi} &= A^\xi \xi + B^\xi v_\xi \end{aligned} \tag{3.2}$$

where $(\eta, \xi) \in \mathbb{R}^{n^\star} \times \mathbb{R}^{n-n^\star}$, the pair (A^ξ, B^ξ) is controllable and the various functions are all smooth [48]. The ξ-subsystem describes the motion off the set $\Gamma^\star$ and for this reason is called the transversal dynamics. The outer transversal control input v_ξ can be used to stabilize the origin of this subsystem using linear control techniques. This ensures attractiveness of $\Gamma^\star$ and invariance of the path.

The zero dynamics of the system, $\dot{\eta} = f^0(\eta, 0) + g^\eta(\eta, 0)v^\eta$, are called the tangential dynamics and govern the motion of the system restricted to the desired path. The outer tangential control input v^η can, in principle, be designed to achieve the desired motion on the path following manifold. However, since the tangential dynamics are in general nonlinear, designing the control law v^η can be difficult [28] or impossible.

[1] The feedback transformation $u = \alpha(x) + \beta(x)v$ is **regular** in U if for all $x \in U$, $\beta(x)$ has rank N.

3.1.1 Tangential Control

In [48], the structure of Euler-Lagrange systems (2.3) is leveraged to refine the normal form (3.2), so that the η-subsystem is partially linear, time-invariant and controllable. By Assumption 3.1.1, there exists a map

$$\sigma : \mathbb{R} \to \mathbb{R}^P \tag{3.3}$$

where, in the case of closed paths, there exists a number $L > 0$ such that for all $\lambda \in \mathbb{R}$, $\sigma(\lambda + L) = \sigma(\lambda)$. If we assume, without loss of generality, that σ is regular, then the smallest such $L > 0$ is the arc-length of the closed curve.

A projection operator is introduced that maps a point y, sufficiently close to the desired path, to a unique $\lambda \in \mathbb{R}$ ($\lambda \in [0, L)$ for closed-paths) such that $\sigma(\lambda)$ is closest to y. In particular,

$$\begin{aligned} \varpi : \mathcal{P}_\epsilon &\to \mathbb{R} \\ y &\mapsto \arg\inf_{\lambda \in \mathbb{R}} \|y - \sigma(\lambda)\| \end{aligned} \tag{3.4}$$

where $\mathcal{P}_\epsilon$ is a tubular neighbourhood of the path $\mathcal{P}$. Using the map (3.4) and $s \circ h(x)$ from Assumption 2, a "virtual" output is defined as:

$$\hat{y} \coloneqq \begin{bmatrix} \pi(x) \\ \gamma(x) \end{bmatrix} \coloneqq \begin{bmatrix} \varpi \circ h(x) \\ s \circ h(x) \end{bmatrix}. \tag{3.5}$$

In [48, Theorem 3.2], it is shown that if for some $x^\star \in \mathbb{R}^n$, $h(x^\star) \in \mathcal{P}$,

$$\mathrm{Im}\left(\frac{\partial h}{\partial x} g_v(x_c) \bigg|_{x=x^\star} \right) \simeq \mathbb{R}^p \tag{3.6}$$

holds for the system (2.3)-(2.4), and for a path $\mathcal{P}$ satisfying Assumptions 3.1.1 and 3.1.2, then via input-output feedback linearization on the virtual output (3.5), the system can be transformed [75] to a refined version of the the normal form (3.2) given by

$$\begin{aligned} \dot{\zeta} &= f^0(\zeta, \eta, \xi) + g^\xi(\zeta, \eta, \xi) v_\xi + g_1^\eta(\zeta, \eta, \xi) v_\eta + g_2^\eta(\zeta, \eta, \xi) v_\zeta \\ \dot{\eta} &= A^\eta \eta + B^\eta v_2^\eta \\ \dot{\xi} &= A^\xi \xi + B^\xi v_\xi. \end{aligned} \tag{3.7}$$

Here $\dim(\zeta) = n^\star - 2$, $\dim(\eta) = 2$, the pairs (A^η, B^η), (A^ξ, B^ξ) are controllable, and $v = \mathrm{col}(v^\eta, v_\xi) = \mathrm{col}(v_\zeta, v_\eta, v_\xi) \in \mathbb{R}^{N-P} \times \mathbb{R} \times \mathbb{R}^{P-1}$. Notice that a basic feasibility

requirement for condition (3.6) is $N \geq P$. If this is true, then for a standard robotic manipulator, (3.6) will hold on its entire state space except at its singular configurations.

As before, stabilizing the path following manifold is equivalent to stabilizing the origin of the ξ-subsystem. On the path following manifold, the dynamics reduce to

$$\begin{aligned} \dot{\zeta} &= f^0(\zeta, \eta, 0) + g_1^\eta(\zeta, \eta, 0)v_\zeta + g_2^\eta(\zeta, \eta, 0)v_\eta \\ \dot{\eta} &= A^\eta \eta + B^\eta v_\eta. \end{aligned} \tag{3.8}$$

At any two points on the path following manifold $(\hat{\zeta}, \hat{\eta})$ and $(\tilde{\zeta}, \tilde{\eta})$, the output (2.4) of (2.3) lies on the desired path $\mathcal{P}$. However, if the output lies on different points on the desired path then, by the definition of the projection operator (3.4), $\tilde{\eta} \neq \hat{\eta}$. Hence, the η-subsystem determines the motion of the output along the desired path. Thus, v_η can be used to control the motion along the path. The ζ-subsystem represents the dynamics on the path following manifold that do not produce motion along the path in the output space. Note that the ζ-subsystem does not exist if $N = P$, thus, this subsystem captures the redundancy of the system (2.3).

In the next section, we force $N = P$ of the CPM by increasing the dimension of the output space to include the end-effector orientation. In Chapter 4, we will see how to resolve redundancies when $N > P$. Although this is a model based technique, we will see in Section 3.2 and 3.3 that a robust controller may be used as the control law for v to account for modelling uncertainty.

3.2 Control Design

3.2.1 Transverse Feedback Linearization

In accordance with the discussion in Section 3.1, consider the CPM (2.7) and let $x_c \coloneqq q, x_v \coloneqq \dot{q}$ where q and $\dot{q}$ are the variables in the CPM model. The model falls into the class of systems (2.2)-(2.4) with $f_v(x) = -M^{-1}(x_c)\left(C(x)x_v + G(x_c)\right)$, $g_v(x_c) = M^{-1}(x_c)A(x_c)$, and let the output be

$$y = h(q) \coloneqq \begin{bmatrix} \cos(q_1)\left(a_2\cos(q_2) + a_3\cos(q_2+q_3)\right) \\ \sin(q_1)\left(a_2\cos(q_2) + a_3\cos(q_2+q_3)\right) \\ a_2\sin(q_2) + a_3\sin(q_2+x_3) + d_1 \\ q_2 + q_3 + q_4 \end{bmatrix}$$

where $y_{1:3}$ denotes the position of the wrist with respect to the base frame $\mathcal{B}$ (see Figure 2.3) using manipulator forward kinematics (see Figure 2.1) and y_4 is the angle the end-effector makes with the ground. Our main focus in this chapter is to deal with robustness to modelling errors, thus we avoid dealing with the redundancy of the robot by augmenting the output y_4. Thus $N = P$ and we expect no ζ dynamics in (3.7). Furthermore, since $P = 4$, the path σ is not only restricting the wrist position of the CPM to a given path, but also the wrist orientation.

The objective is to make the output y approach and traverse a path $\mathcal{P}$ that satisfies Assumptions 3.1.1 and 3.1.2. One such path is a circle in a plane, which will be used for illustration, but more complicated paths such as splines can be used with the proposed approach (see Chapter 4). Without loss of generality, let the circle be centred at the coordinate $o_c \in \mathbb{R}^3$ with respect to the basis $\mathcal{B}$, have radius r, and be contained in the plane defined by the unit normal vector $Y_{c_3} \in \mathbb{R}^3$ passing through o_c. Furthermore, let $Y_{c_3} = Y_{c_1} \times Y_{c_2}$, where $\|Y_{c_i}\| = 1$, $i \in \{1, 2\}$. The coordinate transformation $T^Y_{Y_c} : \mathbb{R}^3 \to \mathbb{R}^3$,

$$y = T^Y_{Y_c}(y_c) = \begin{bmatrix} Y_{c_1} & Y_{c_2} & Y_{c_3} \end{bmatrix} y_c + o_c \tag{3.9}$$

transforms coordinates y_c in the frame $\{Y_{c_1}, Y_{c_2}, Y_{c_3}\}$ attached to the center of the circle to the output coordinate frame $\{Y_1, Y_2, Y_3\}$. Thus, for some desired end-effector angle $\psi \in [0, 2\pi)$ in the output space, the parametrized curve can be written as

$$\sigma(\lambda) = \begin{bmatrix} T^Y_{Y_c} \begin{pmatrix} r\cos(\lambda) \\ r\sin(\lambda) \\ 0 \end{pmatrix} \\ \psi \end{bmatrix}. \tag{3.10}$$

The corresponding lift of the path (see Assumption 3.1.2) is given by (3.1), where

$$s \circ h(x) \coloneqq \begin{bmatrix} \begin{pmatrix} y_{c_1}^2 + y_{c_2}^2 - r^2 \\ y_{c_3} \end{pmatrix}_{y_c = \left(T^Y_{Y_c}\right)^{-1} \circ h(x)} \\ x_2 + x_3 + x_4 - \psi \end{bmatrix} \tag{3.11}$$

where the elements of s correspond to being on the desired cylinder, plane, and end-effector angle, respectively.

To get to the normal form (3.7), condition (3.6) needs to be satisfied. It is necessary that $N \geq P$ for this to hold, which is true for this example. Furthermore, the matrix $\frac{\partial h}{\partial x} g_v(x_c)$ is full rank everywhere except at the singularities of the manipulator (shoulder link lines up with the elbow link) and when $A(x_c)$ is singular. Therefore, the path can be positioned to avoid these singularities and thus will satisfy (3.6).

Next the projection operator (3.4) is required in the construction of the virtual output (3.5). In the case of a circle,

$$\varpi(y) = r \arg\left((y_{c_1} + i y_{c_2})_{y_c = \left(T_{Y_c}^{Y}\right)^{-1}(y)} \right) \tag{3.12}$$

is the arc length of the output on the circle, and thus will uniquely project points off the circle to $(-r\pi, r\pi] \subset \mathbb{R}$ provided that $(y_{c_1}, y_{c_2}) \neq (0,0)$. The functions $\gamma(x) = s \circ h(x)$, $\pi(x) = \varpi \circ h(x)$ are obtained from (3.11) and (3.12), and the virtual output (3.5) has a well defined relative degree at each point on the circle. Thus the desired normal form can be obtained by applying the coordinate transformation,

$$\begin{aligned} T : U \subseteq \mathbb{R}^8 &\to T(U) \subseteq (-r\pi, r\pi] \times \mathbb{R}^7 \\ x \mapsto (\eta, \xi) = \begin{bmatrix} \eta_1 \\ \eta_2 \\ \xi_1^i \\ \xi_2^i \end{bmatrix} &= \begin{bmatrix} \pi(x) \\ L_f \pi(x) \\ \gamma_i(x) \\ L_f \gamma_i(x) \end{bmatrix}, \end{aligned} \tag{3.13}$$

for $i \in \{1,2,3\}$. The states $\xi_1^1, \xi_1^2, \xi_1^3$ correspond to the error to the cylinder, plane, and end-effector angle, respectively. Although the above coordinate transformation is only guaranteed to be well-defined in a neighbourhood of a point $x^\star$ on the set $\Gamma^\star$, it can be shown that in the case of the circular paths, U contains the entire set $\Gamma^\star$. The dynamics in $(\eta, \xi)-$coordinates become

$$\left.\begin{aligned} \dot{\eta}_1 &= \eta_2 \\ \dot{\eta}_2 &= L_f^2 \pi(x)|_{x=T^{-1}(\eta,\xi)} + L_g L_f \pi(x)|_{x=T^{-1}(\eta,\xi)} u \\ \dot{\xi}_1^i &= \xi_2^i \\ \dot{\xi}_2^i &= L_f^2 \gamma_i(x)|_{x=T^{-1}(\eta,\xi)} + L_g L_f \gamma_i(x)|_{x=T^{-1}(\eta,\xi)} u \end{aligned}\right\} i \in \{1,2,3\}. \tag{3.14}$$

The feedback transformation, which is well defined in U,

$$\begin{aligned} u &= \begin{bmatrix} L_g L_f \pi(x) \\ L_g L_f \gamma(x) \end{bmatrix}^{-1}_{x=T^{-1}(\eta,\xi)} \left(- \begin{bmatrix} L_f^2 \pi(x) \\ L_f^2 \gamma(x) \end{bmatrix}_{x=T^{-1}(\eta,\xi)} + v \right) \\ &=: \beta^{-1}(\eta,\xi)(-\alpha(\eta,\xi) + v) \end{aligned} \tag{3.15}$$

yields the feedback linearized system

$$\begin{array}{llll} \dot{\eta}_1 = \eta_2 & \dot{\xi}_1^1 = \xi_2^1 & \dot{\xi}_1^2 = \xi_2^2 & \dot{\xi}_1^3 = \xi_2^3 \\ \dot{\eta}_2 = v_\eta & \dot{\xi}_2^1 = v_{\xi^1} & \dot{\xi}_2^2 = v_{\xi^2} & \dot{\xi}_2^3 = v_{\xi^3} \end{array} \tag{3.16}$$

where $v = \mathrm{col}(v_\eta, v_\xi) \in \mathbb{R} \times \mathbb{R}^3$. This is precisely the desired normal form (3.7) with $A^\eta = \left[\begin{smallmatrix} 0 & 1 \\ 0 & 0 \end{smallmatrix}\right], B^\eta = \left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right], A^\xi = \mathrm{diag}(A_1, A_2, A_3), A_i = \left[\begin{smallmatrix} 0 & 1 \\ 0 & 0 \end{smallmatrix}\right], B^\xi = \mathrm{diag}(B_1, B_2, B_3), B_i = \left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right]$. The outer control law for v can be designed by linear control techniques to stabilize the origin of the ξ-subsystem in order to stabilize the manifold $\Gamma^\star$ (can be done via state-feedback or robust linear control methods [32]), and to track desired motions along the path governed by the η_2-subsystem. Note that the decoupling matrix $\beta(\,\cdot\,) \in \mathbb{R}^{4\times 4}$ in (3.15) is full rank since (3.6) holds and the path satisfies Assumptions 3.1.1 and 3.1.2, except at the center of the circle and at the singular configurations of the manipulator. In Chapter 4, a numerical algorithm is employed which removes the singularity when the output is equidistant to multiple points on the path $\mathcal{P}$.

3.2.2 Robust Controller

To track a desired constant velocity η_2^{ref} along the path, the following outer control law may be used

$$v_\eta(\eta, t) = K_{p_1}\left(\eta_2^{\mathrm{ref}} - \eta_2(t)\right) + K_{I_1}\int_0^t \left(\eta_2^{\mathrm{ref}} - \eta_2(\tau)\right)\mathrm{d}\tau. \tag{3.17}$$

If a time-varying tangential velocity profile is to be tracked, one can add a feed-forward term. The above PI control law works well in practice for a majority of systems in the presence of disturbance [64]. It turns out that this controller works well for our tracking purposes for the tangential speed along the path. The ξ-subsystem in (3.14) with (3.15) can be rewritten as

$$\dot{\xi} = A^\xi + B^\xi\left[\alpha(\eta, \xi) + \beta(\eta, \xi)u\right]_{2:4} \tag{3.18}$$

Indeed the feedback transformation (3.15) results in perfect cancellation of the nonlinear terms $\alpha(\,\cdot\,), \beta(\,\cdot\,)$ resulting in the linearized system $\dot{\xi} = A^\xi\xi + B^\xi v_\xi$. However, with the unmodelled dynamics and parameter uncertainties described in Section 2.1.1, the resulting system takes the form [37]

$$\begin{aligned}\dot{\xi} &= A^\xi\xi + B^\xi\left[\alpha(\eta, \xi) + \Delta_\alpha(\eta, \xi) + \left(\beta(\eta, \xi) + \Delta_\beta(\eta, \xi)\right)u\right]_{2:4} \\ &=: A^\xi\xi + B^\xi\left[\alpha(\eta, \xi) + \beta(\eta, \xi)u\right]_{2:4} + \delta(\eta, \xi, u)\end{aligned}$$

where $\Delta_\alpha(\,\cdot\,) \in \mathbb{R}^4, \Delta_\beta(\,\cdot\,) \in \mathbb{R}^{4\times 4}$ represent the modelling uncertainties. Note that this model of uncertainty assumes that the inputs do not appear in earlier derivatives of the

output than expected. Thus we do not expect perfect cancellation of the nonlinear terms using the control input (3.15). The outer control input v_ξ can be designed to overcome the perturbation term $\delta(\eta, \xi, u)$. It is shown in [37] that if the uncertainty satisfies

$$\begin{aligned} \| \left(\Delta_\beta(\eta,\xi) + \beta(\eta,\xi)\right)^{-1} \beta(\eta,\xi) - I_{4\times 4} \| &\leq k_1 < 1 \\ \| \left(\Delta_\beta(\eta,\xi) + \beta(\eta,\xi)\right)^{-1} \Delta_\alpha(\eta,\xi) \| &\leq k_2 \| \operatorname{col}(\eta,\xi) \| \end{aligned} \tag{3.19}$$

then the inner control loop (3.15) and the outer controller of the form

$$v_\xi = (K + K_0)\,\xi + \begin{cases} K_1 \frac{\xi}{\|\xi\|} & : \|\xi\| \geq \mu > 0 \\ K_2 \|\xi\| \xi & : \|\xi\| < \mu. \end{cases} \tag{3.20}$$

where $(A^\xi + B^\xi K)$ is Hurwitz, $K_0, K_1, K_2, \in \mathbb{R}^{3\times 3}, \mu \in \mathbb{R}$ are constants that depend on k_1, k_2, then the origin of ξ will be asymptotically stable. In practice, finding k_1, k_2 is difficult, which implies difficulty in selecting μ and the gains K_i. However we treat μ and K_i as tuning parameters in the following section.

3.3 Experimental Results

We implement the path following controller on the CPM via Labview Real-Time Module© to control the motor PWM amplifiers and to read the optical encoders. The encoders read the d_i of equation (2.5) (the linear actuator extension). Hence the inverse mapping of (2.6), l_i^{-1}, is used to retrieve the states x_c of the system. The derivative of x_c is numerically computed to retrieve the x_v states. The block diagram of the control system can be seen in Figure 3.1.

To check the validity of the robust controller (3.20), we examine how uncertain the nominal model is. Figure 3.2 compares two transversal controllers, one with simple state feedback $v_\xi = K\xi$ and the other with the robust controller (3.20). The feedback gain K is the same for both controllers, and thus, only the robustifying terms were tuned in the latter controller. Clearly the simple state feedback on ξ was not able to stabilize the transversal subsystem (ξ-subsystem) and hence the wrist position is far from the desired path, showing the need and usefulness of the robust controller to overcome model uncertainty. There is still slight error due to the bounds of the perturbation terms $\Delta_\alpha, \Delta_\beta$ not satisfying (3.19). Nevertheless the error is significantly reduced.

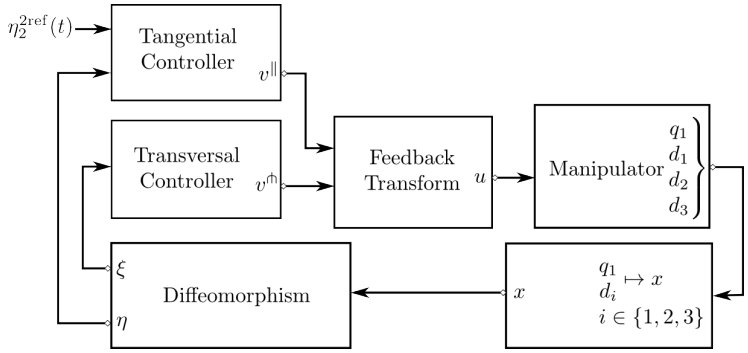

Figure 3.1: Block diagram of the robust path following control system.

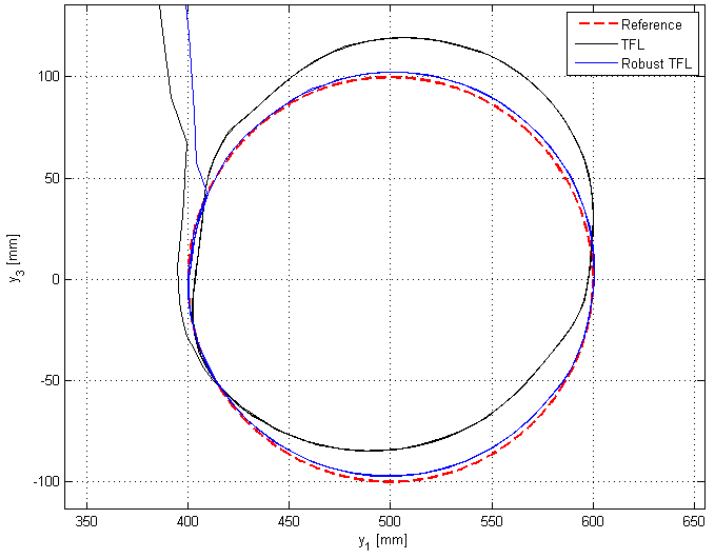

Figure 3.2: Experiment: Comparison of our path following controller with and without the robust component (3.20) on the transversal states. The path parameters in (3.9) and (3.10) are $Y_{c_1} = [100]^\top, Y_{c_2} = [001]^\top, Y_{c_3} = [010]^\top, \psi = 0$, $r = 100\text{mm}$, and $o_c = [50000]^\top$.

The robust control law is also compared with the simple trajectory tracking controller of independent-joint PID [98]:

$$u_i = K_{P_i}\left(x_{c_i}^{\text{ref}}(t) - x_{c_i}(t)\right) + K_{D_i}\left(x_{v_i}^{\text{ref}}(t) - x_{v_i}(t)\right) + K_{I_i}\int_0^t \left(x_{c_i}^{\text{ref}}(\tau) - x_{c_i}(\tau)\right)\mathrm{d}\tau,\ i \in \{1,2,3,4\},$$

where the reference signal x_c^{ref} is generated by solving the inverse kinematics for the desired path (3.10) with the desired speed parametrization. Namely, the path parameter $\lambda(t)$ is such that $\dot{\lambda}(t) = \frac{\eta_2^{\text{ref}}}{r}$. The joint velocities x_v^{ref} are generated by $x_v^{\text{ref}} = J^{-1}(x_c^{\text{ref}})\sigma'(t)$, where $J = \frac{\partial h(x_c)}{\partial x_c}$ is the manipulator Jacobian [98].

We test the path of a circle in the horizontal plane shown in Figure 3.3 with a constant reference velocity of $\eta_2^{2\text{ref}} = 100\frac{\mathbf{mm}}{\mathbf{s}}$ tangent to the circle. The transformed state positions are shown in Figure 3.4. From (3.11) and (3.13), the states $\xi_1^1, \xi_2^1, \xi_3^1$ correspond to the error to the cylinder, plane, and end-effector angle, respectively.

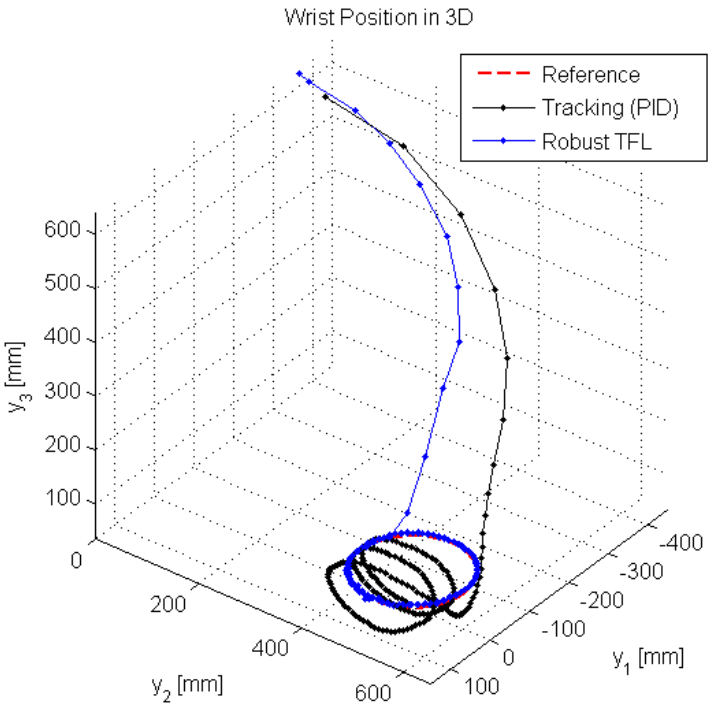

Figure 3.3: Experiment: Comparison of the robust path following controller with a trajectory tracking controller in the output space. The path parameters in (3.9) and (3.10) are $Y_{c_1} = [1\,0\,0]^\top, Y_{c_2} = [0\,1\,0]^\top, Y_{c_3} = [0\,0\,1]^\top, \psi = 0$, $r = 100$ mm, and $o_c = [0\,500\,100]^\top$.

These results demonstrate one of the main disadvantages of classical trajectory tracking control. The large initial error causes a poor transient response, and since the controller is trying to catch up to the trajectory, forces the manipulator to actually cut across the cylinder. On the other hand, the path following controller naturally converges to the closest point on the path due to the explicit stabilization of the path following manifold $\Gamma^\star$. The

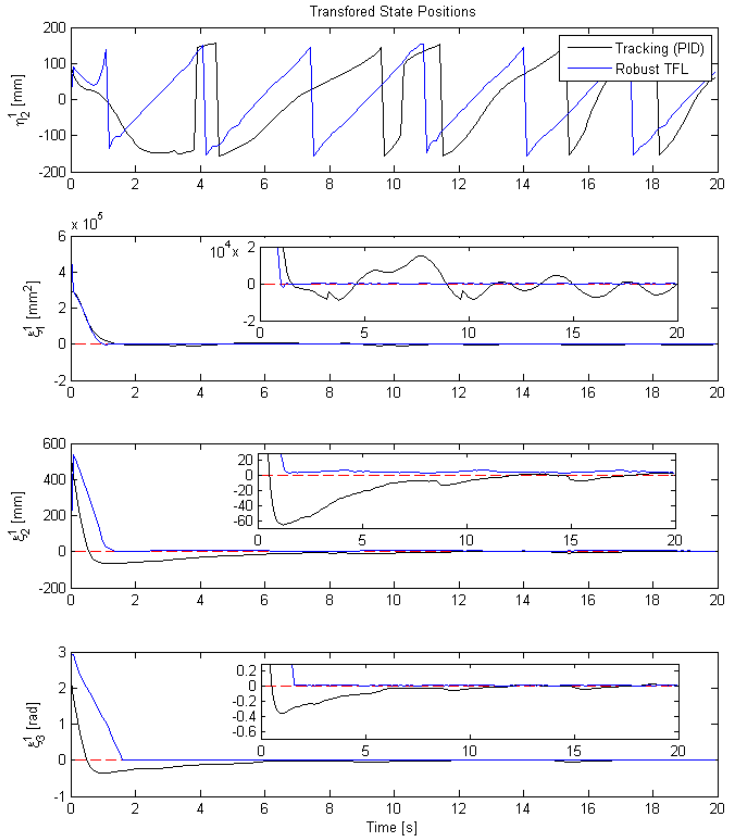

Figure 3.4: Experiment: Transformed state positions with the transformation T applied from (3.13).

dynamics are being accounted for in a robust manner, thus gain-tuning for the outer loop controller on v becomes easier as the dynamics appear linear in this space.

The inputs are shown in Figure 3.3. The two controllers are operating in a similar range of control effort.

3.4 Summary

This chapter proposes a robust path following controller design approach for articulated manipulators. The path following controller uses transverse feedback linearization to stabilize the path following manifold. Then, by Lyapunov redesign, a robust controller is used

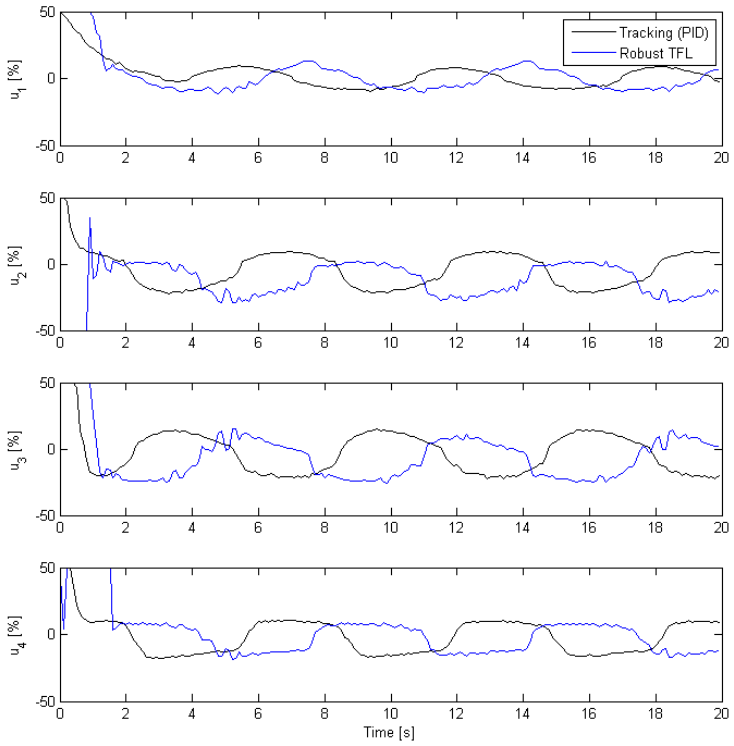

Figure 3.5: Experiment: Comparison of the control signal for the path following controller and the trajectory tracking controller.

to enforce asymptotic stability of the transverse dynamics in the presence of modelling uncertainty. In addition, the controller uses a refinement of the normal form to facilitate desired motion on the path via a linear controller. The approach was validated on a 4-DOF robot manipulator using circles as the paths, and performed well despite the modelling inaccuracies. This achieves the robustness aspects of **PF1** to **PF3**. The generalization of the paths in **PF1** to **PF3** will be addressed in the next chapter.

We also fully linearized the system by defining a four dimensional output space. Had we removed the constraint ψ on the end-effector angle, the resulting system would have been redundant. Dealing with redundancy in the context of path following will be addressed in the next chapter, in order to achieve **PF4** for Euler-Lagrange systems.

Chapter 4

Spline Path Following and Redunancy Resolution

In this chapter, we extend the class of paths of the previous chapter to framed curves generated by splines in the workspace of redundant Euler-Lagrange systems (2.3) with output (2.4). The class of admissible paths includes self-intersecting curves (Section 4.1). Kinematic redundancies are resolved by designing controllers that solve a suitably defined constrained quadratic optimization problem (Section 4.2.3). By employing partial feedback linearization, the proposed path following controllers have a clear physical meaning. The approach is experimentally verified on the CPM using the robust controller developed in Chapter 3 (Section 4.3). Portions of this chapter have been submitted to the IEEE Transactions on Robotics journal and is currently under review [43].

4.1 Admissible paths

We assume that we are given a finite number, $n_{\text{spline}} \in \mathbb{N}$, of $(P+1)$-times continuously differentiable, parametrized curves in the output space of (2.3), (2.4)

$$\sigma_k : \mathbb{R} \to \mathbb{R}^P, \quad k \in \{1, 2, \ldots, n_{\text{spline}}\}\,. \tag{4.1}$$

The curves need not be polynomials, but we nevertheless use the term spline in accordance with common practice. We are also given n_{spline} non-empty, closed, intervals of the real

line $\mathbb{I}_k \coloneqq [\lambda_{(k,\min)}, \lambda_{(k,\max)}] \subset \mathbb{R}$. The desired path is the set

$$\mathcal{P} \coloneqq \bigcup_{k=1}^{n_{\text{spline}}} \sigma_k\left(\mathbb{I}_k\right). \tag{4.2}$$

If $P > 3$, then the path also specifies a desired orientation of the end-effector. We assume that the overall path is $(P+1)$-times continuously differentiable in the following sense:

Assumption 4.1.1 (smoothness of path)**.** *The curves* (4.1) *satisfy*

$$(\forall k \in \{1, 2, \ldots, n_{\text{spline}} - 1\})\,(\forall i \in \{0, 1, \ldots, P+1\}) \quad \left.\frac{\mathrm{d}^i \sigma_k}{\mathrm{d}\lambda^i}\right|_{\lambda=\lambda_{(k,\max)}} = \left.\frac{\mathrm{d}^i \sigma_{k+1}}{\mathrm{d}\lambda^i}\right|_{\lambda=\lambda_{(k+1,\min)}}.$$

In the case of closed paths, the same equality must hold between spline index 1 *and* n_{spline}*.*

This assumption is required to ensure continuous control effort over the entire path. In the case of spline interpolation (fitting polynomials between desired waypoints), Assumption 4.1.1 can be enforced by setting it as a constraint in the spline fitting formulation and using polynomials of sufficiently high degree. Typically the parameter λ of each curve σ_k represents the chord length between waypoints; namely, $\lambda_{(k,\min)} = 0$ and $\lambda_{(k,\max)} = l_k$ where l_k is the Euclidean distance between the k'th and $(k+1)$ waypoints [35].

Remark 4.1.1. Assumption 4.1.1 can be relaxed if continuity of the control signal over the entire path isn't required. Furthermore, if Assumption 4.1.1 is only true for $i = 0$, i.e., the path is not continuously differentiable, the control input can still be kept continuous by designing the desired tangential velocity profile in such a way that the output slows down and stops at $\sigma_k(\lambda_{(k,\max)}) = \sigma_{k+1}(\lambda_{(k+1,\min)})$, and then resumes motion along curve $k+1$ in finite time. See Remark 4.2.4 •

We further assume that the curves (4.1) are framed.

Assumption 4.1.2 (framed curves)**.** *For each spline* k*, the first* P *derivatives of* σ_k *are linearly independent, i.e.,*

$$(\forall k \in \{1, 2, \ldots, n_{\text{spline}}\})\,(\forall \lambda \in \mathbb{I}_k)\,\operatorname{span}_{\mathbb{R}}\left\{\sigma_k'(\lambda), \sigma_k''(\lambda), \ldots, \sigma_k^{(P)}(\lambda)\right\} = \mathbb{R}^P.$$

Assumption 4.1.2 allows the use of Gramm-Schmidt orthogonalization to construct Frenet-Serret frames (FSFs) in the output space $\mathbb{R}^P$. The use of FSFs for control of mobile robots is a well-known technique [20]. In this paper we use Assumption 4.1.2 to

generate a zero-level set representation of $\mathcal{P}$ in the state space of (2.3). Previous works [48] (see Chapter 3), relied on the zero-level set representation to already be available by restricting the class of paths to one-dimensional embedded submanifolds, which precludes self-intersecting curves.

Remark 4.1.2. Assumption 4.1.2 can be relaxed to only requiring that the curve be regular by not using the FSF at each point along the curve. Consider a straight line $\sigma(\lambda) = \begin{bmatrix}\lambda & 0\end{bmatrix}^\top \in \mathbb{R}^2$. This curve violates Assumption 4.1.2 at all points λ. The first basis vector is taken to be the unit tangent vector $\begin{bmatrix}1 & 0\end{bmatrix}^\top$(as is the case in the FSF). The normal vector cannot be constructed via FSF, since the curvature is zero. However, one can choose the normal vector to be $\begin{bmatrix}0 & 1\end{bmatrix}^\top$.

Furthermore, consider a regular torsion free curve in the xy-plane of a 3-dimensional workspace. This curve also violates Assumption 4.1.2, since $\sigma^{(3)}(\lambda)$ will be a linear combination of σ' and σ''. However, one can choose the first basis vector to again be the unit tangent vector $\frac{\sigma'(\lambda)}{\|\sigma'(\lambda)\|}$. The unit normal vector can be chosen to be the 90 degree rotation of the first basis vector, and the final basis vector can be chosen to be $\begin{bmatrix}0 & 0 & 1\end{bmatrix}^\top$. Once the appropriate frame is chosen, the proposed approach for coordinate transformation and control design (Section 4.2) can be applied without any additional modifications. •

Unlike in the previous chapter, we do not require that the path $\mathcal{P}$ be an embedded, nor immersed, submanifold of the output space. In other words the curve may be self-intersecting. Removing this restriction is particularly useful because it allows the curve to go through an arbitrary set of waypoints in the output space of system (2.3) and can be constructed using classical spline fitting.

The problem studied in this chapter is to find a continuous feedback controller for system (2.3), with n_{spline} curves that satisfy Assumptions 4.1.1 and 4.1.2, that renders the path $\mathcal{P}$ in (4.2) invariant and attractive (see Section 1.1). The motion along the path should also be controllable, so that a tangential position or velocity profile ($\eta^{\text{ref}}(t)$) can be tracked. The robust control approach of Chapter 3 should also be applied to handle modelling uncertainty in the system. These points collectively satisfy **PF1** to **PF3**. Furthermore, any remaining redundant dynamics (will show up as internal dynamics for the closed-loop system) must be bounded, thus achieving **PF4** for Euler-Lagrange systems. In Chapter 5, **PF4** will be extended to mobile manipulator systems.

4.2 Multiple Spline Path Following Design

We begin by defining a coordinate transformation in Section 4.2.1 that depends on a given path segment (4.1) and the system dynamics (2.3). These coordinates partition the system dynamics into (3.7), that is a sub-system that governs the motion transversal to the desired path and a tangential sub-system that governs the motion of the system along the path (Section 4.2.2). One of the keys to this construction in this chapter is the ability to unambiguously determine the spline that is currently closest to the system output (2.4). By doing so, we are able to deal with self-intersecting curves and curves (4.2) that aren't embedded submanifolds. A novel redundancy resolution approach is then proposed to ensure the redundant dynamics remain bounded (Section 4.2.3). The overall block diagram can be found below in Figure 4.1.

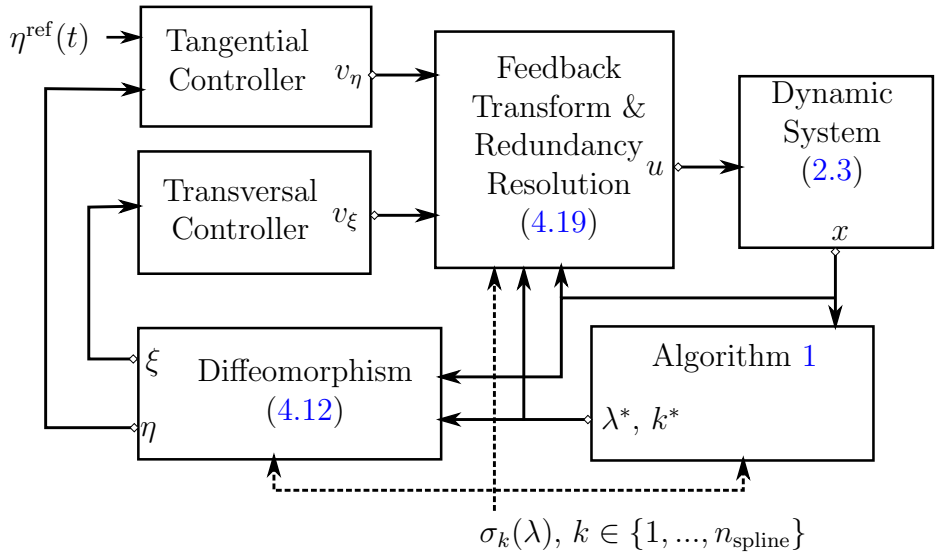

Figure 4.1: Spline Path following control system block diagram for redundant Euler-Lagrange systems.

4.2.1 Coordinate Transformation

We construct a coordinate transformation T that maps the state x of system (2.3) to a new set of coordinates that correspond to the tangential position/velocity along the path, and the distance and velocity towards/transversal to the path.

Tangential States

For the moment assume that the closest spline to the output y is known; denote it as k^*. In the case that $n_{\text{spline}} = 1$, the k notation can be dropped to ease readability. Furthermore, denote the parameter of spline k^* that corresponds to the closest point to the output y as $\lambda^* \in \mathbb{I}_{k^*}$ [48] (cf. Equation (3.4)):

$$\lambda^* \coloneqq \varpi_{k^*}(y) \coloneqq \underset{\lambda \in \mathbb{I}_{k^*}}{\arg\inf} \left\| y - \sigma_{k^*}(\lambda) \right\|. \tag{4.3}$$

The first tangential state is the projected, traversed arclength along the path (cf. Equation (3.4), (3.5)):

$$\eta_1 = \eta_{1_{k^*}}(x) \coloneqq s_{k^*} \circ \varpi_{k^*} \circ h(x) + \sum_{k=1}^{k^*-1} s_k(\lambda_{(k,max)}) \tag{4.4}$$

where

$$s_k(\lambda) \coloneqq \int_{\lambda_{(k,min)}}^{\lambda} \left\| \frac{\mathrm{d}\sigma_{k^*}(\lambda)}{\mathrm{d}\lambda} \right\| \mathrm{d}\lambda.$$

This integral does not have to be computed for real time implementation if only tangential velocity control is required, which will be the case in the experiment (Section 4.3). Note the slight abuse of notation for using η_1 to represent both the state and the map. The summation term on the right in (4.4) is used to get the arclength over the entire path $\mathcal{P}$, so that η_1 remains continuous even as the current spline number k^* increments/decrements. Note that this summation can be computed offline and accessed as a look-up-table during real-time control.

With η_1 representing the projected tangential position of the output y along the path, tangential velocity is computed by taking the time derivative. By the fundamental theorem of calculus $\frac{\mathrm{d}s_{k^*}}{\mathrm{d}\lambda^*} = \|\sigma'_{k^*}(\lambda^*)\|$. Geometric arguments provide that $\frac{\mathrm{d}\varpi_{k^*}}{\mathrm{d}y} = K_{k^*}(y)\sigma'_{k^*}(\lambda^*)^\top$ where $K_{k^*} : \mathbb{R}^p \to \mathbb{R}$ is a smooth scalar function [28]. To solve for $K_{k^*}(y)$, the following identity may be used:

$$\varpi_{k^*}(y) = \varpi_{k^*}(\sigma_{k^*}(\varpi_{k^*}(y))).$$

Differentiating both sides:

$$\begin{aligned}
&\frac{\mathrm{d}\varpi_{k^*}}{\mathrm{d}y} = \frac{\mathrm{d}\varpi_{k^*}}{\mathrm{d}y}\,\sigma'(\lambda)|_{\lambda=\lambda^*}\,\frac{\mathrm{d}\varpi_{k^*}}{\mathrm{d}y} \\
\Rightarrow &\frac{\mathrm{d}\varpi_{k^*}}{\mathrm{d}y}\,\sigma'_{k^*}(\lambda)|_{\lambda=\lambda^*} = 1 \\
\Rightarrow &\frac{\mathrm{d}\varpi_{k^*}}{\mathrm{d}y}\,\sigma'_{k^*}(\lambda)|_{\lambda=\lambda^*} = \left(K_{k^*}(y)\sigma'_{k^*}(\lambda^*)^\top\right)\sigma'_{k^*}(\lambda^*) = 1 \\
\Rightarrow &K_{k^*}(y) = \frac{1}{\|\sigma'_{k^*}(\lambda^*)\|^2}.
\end{aligned}$$

Thus η_2 can be compactly written as

$$\begin{aligned}
\eta_2 = \eta_{2_{k^*}}(x) \coloneqq \dot{\eta}_1 &= \frac{\mathrm{d}s_{k^*}}{\mathrm{d}\lambda^*}\frac{\mathrm{d}\varpi_{k^*}}{\mathrm{d}y}\frac{\partial h}{\partial x_c} \\
&= \langle \mathbf{e}_{1_{k^*}}(\lambda^*), J(x_c)x_v\rangle|_{\lambda^*=\varpi\circ h(x)}
\end{aligned} \tag{4.5}$$

where $\mathbf{e}_1$ is the unit-tangent Frenet-Serret vector:

$$\mathbf{e}_{1_{k^*}}(\lambda^*) \coloneqq \frac{\mathrm{d}\sigma_{k^*}}{\mathrm{d}s_{k^*}} = \frac{\mathrm{d}\sigma_{k^*}}{\mathrm{d}\lambda}\bigg|_{\lambda=\lambda^*}\frac{\mathrm{d}\lambda^*}{\mathrm{d}s_{k^*}} = \frac{\sigma'_{k^*}(\lambda^*)}{\|\sigma'_{k^*}(\lambda^*)\|}. \tag{4.6}$$

.

Numerical Optimization for λ^* and k^*

The tangential coordinates defined by (4.4), (4.5) rely on the computation of the parameters k^* – which characterizes which of the n_{spline} curves $\sigma_k(\mathbb{I}_k)$ is closest to the system's output – and λ^* – which characterizes the point $\sigma_{k^*}(\lambda^*)$ closest to the system's output.

Assumption 4.2.1 (Initial λ^* and k^* are known)**.** *Based on the initial conditions of the system at time $t = 0$, the corresponding $k^* \in \{1, ..., n_{\text{spline}}\}$ and $\lambda^* \in \mathbb{I}_{k^*}$ are known.*

Assumption 4.2.1 can be satisfied by *apriori* global minimum brute-force optimization solvers to find the closest point of $h(x(0))$ to the path $\mathcal{P}$. One can do this by quantizing the path $\mathcal{P}$ and searching for the point with minimum distance to the output, whilst keeping track of the curve number k and the path parameter λ. The output to this approach will not yield the exact solution (due to the quantization of $\mathcal{P}$), but will be adequate for sufficiently small quantization. In the case that multiple global minima exist, one can be

chosen arbitrarily. Then for $t > 0$, a numerical local optimizer like gradient descent can be used online, initialized at the previous time step's k^*, λ^*. The algorithm we developed is Algorithm 1 below.

Due to the nature of numerical algorithms, the computation of λ^* and k^* for $t > 0$ is done discretely. At the next time-step $t = \Delta t$ (where Δt is sampling time of the computer) and as the output y moves, the next, closest local minimum of $\|y - \sigma_{k^*}(\lambda)\|$ is the global minimum, assuming that the output y hasn't moved far within the time step. Thus, simple numerical algorithms like gradient descent can be used to find the nearest local minimum which will end up being the global minimum. The idea is illustrated in Figure 4.2 below.

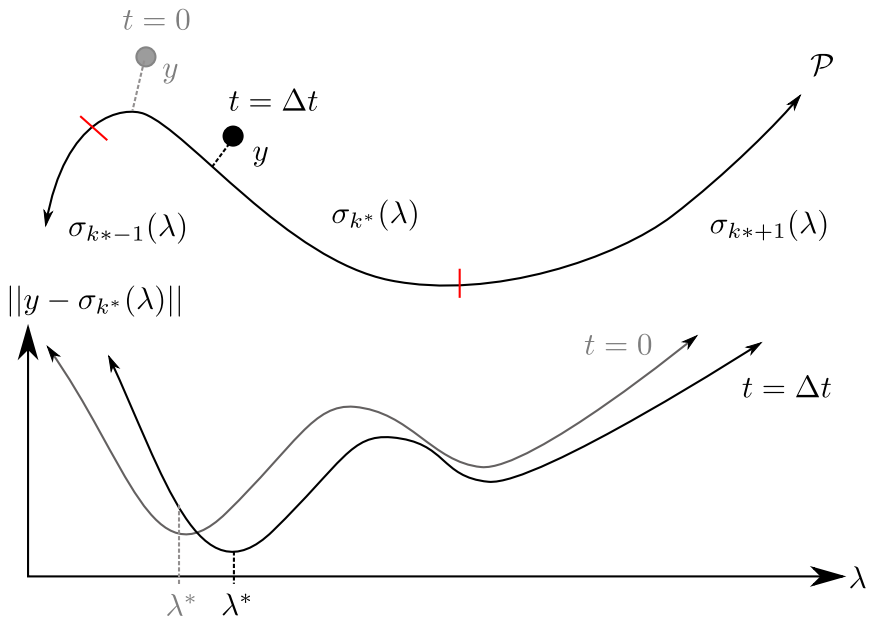

Figure 4.2: Numerical optimization: At the start of the run, the true λ^* is known by Assumption 4.2.1. As the output y moves, at each subsequent time-step a numerical optimizer is likely to find the global minimum of $\|y - \sigma_{k^*}(\lambda)\|$ if initialized at the previous time-step's λ^*, since the global minimum will be near.

For a moment assume that the output y is already on the path ($y = \sigma_{k^*}(\lambda^*)$). If the output y moves too far within one time step (corresponds to a large change in the path parameter λ^*), then there may be a local minimum between the initial guess (the previous time-step's solution) and the true global minimum. Thus, we can ensure that a numerical algorithm will not get stuck at such a well by ensuring the path parameter only changes by some amount $\Delta\lambda$, which keeps the function $\|y - \sigma_{k^*}(\lambda)\|_{y=\sigma_{k^*}(\lambda^*)}$ convex for

$\lambda \in [\lambda^* - \Delta\lambda, \lambda^* + \Delta\lambda]$. If σ_{k^*} is unit-speed parametrized, i.e., $\|\sigma'(\lambda)\| = 1$ for all λ, then $\Delta\lambda$ corresponds to the allowable distance along the curve that can be travelled.

This $\Delta\lambda$ is a conservative estimate for the allowable change in parameter in one time-step, since requiring the function to be convex is sufficient, not necessary to ensure convergence of a numerical algorithm to λ^* if initialized properly. In logic notation, $\Delta\lambda$ must satisfy

$$(\forall \lambda^* \in \mathbb{I}_{k^*})\,(\forall \lambda \in [\lambda^* - \Delta\lambda, \lambda^* + \Delta\lambda] \cap \mathbb{I}_{k^*})\, \frac{\mathrm{d}^2 \left(\|y - \sigma_{k^*}\lambda)\|_{y=\sigma_{k^*}(\lambda^*)} \right)}{\mathrm{d}\lambda^2} > 0$$

which is equivalent to

$$(\forall \lambda^* \in \mathbb{I}_{k^*})\,(\forall \lambda \in [\lambda^* - \Delta\lambda, \lambda^* + \Delta\lambda] \cap \mathbb{I}_{k^*}) \\ \langle \sigma_{k^*}(\lambda^*) - \sigma_{k^*}(\lambda), \sigma''_{k^*}(\lambda) \rangle + \frac{\langle \sigma_{k^*}(\lambda^*) - \sigma_{k^*}(\lambda), \sigma'_{k^*}(\lambda) \rangle^2}{\|\sigma_{k^*}(\lambda^*) - \sigma_{k^*}(\lambda)\|^2} < \|\sigma'_{k^*}(\lambda)\|^2 . \quad (4.7)$$

The relation in (4.7) can be used as a test to infer the allowable speed at which the output y can traverse a path in order for a numerical optimizer to succeed in evaluating $\varpi_{k^*}(y) = \arg\inf_{\lambda \in \mathbb{I}_{k^*}} \|y - \sigma_{k^*}(\lambda)\|$. In particular for a unit-speed parametrized curve and assuming the output is on the path $\mathcal{P}$, if (4.7) holds for some $\Delta\lambda$, then the allowable speed along the curve (η_2) is $\frac{\Delta\lambda}{\Delta t}$.

Example 2. (Ellipse) Let $n_{\text{spline}} = 1$ and consider the path of a single ellipse (the k^* subscript will be dropped) $\sigma(\lambda) = (2\cos(\lambda), \sin(\lambda))$. When $\lambda^* = 0$, all $\lambda \in [-1.5136, 1.5136]$ satisfies the inequality in (4.7). This means that when the output is at $\sigma(0)$, the change in the path parameter in one time step is $+/-1.5136$. This value can be solved for analytically or numerically using the inequality in (4.7). We can run the same test for all λ^* over the path and take the smallest resulting range to be the $\Delta\lambda$. △

A numerical algorithm like steepest descent works for finding λ^* if it starts within $\Delta\lambda$ of λ^*, however it normally takes many iterations to converge since the algorithm takes small step sizes to the solution [14]. Another approach is to use some knowledge about η_2^{ref} to adjust the step size. In particular, if the path is unit-speed parametrized, one could use an initial step size of $\eta_2^{\text{ref}}\Delta t$ in the direction of the steepest descent to quickly approach the solution. If the path is not unit-speed parametrized, a map of $\eta_1^{\text{ref}}(t)$ and λ^* can be constructed using (4.4) sampled at each time step, and the smallest jump in λ^* can be used as the initial step size. An algorithm like monotonic gradient descent with adaptive step-size works nicely (Algorithm 1) using this initial step size [102].

Determining k^* is just a matter of checking if the λ^* computed by the numeric algorithm is outside the domain $\mathbb{I}_{k^*}$. If it is, then k^* must be incremented or decremented, as done in Algorithm 1. Let $\bar{\varpi}_{k^*}(\lambda) := \|y - \sigma_{k^*}(\lambda)\|$ be the function to be minimized. The algorithm for determining λ^* and k^* can be found in Algorithm 1.

Algorithm 1 Determining λ^* and k^* at each time step.

Require: The closest spline k^* and corresponding λ^*, stepsize α, the gradient $\frac{\partial \bar{\varpi}_{k^*}}{\partial \lambda}(\lambda^*)$, and an ϵ for the stopping criterion.

Ensure: The closest spline k^* and the λ^* minimizing $\bar{\varpi}_{k^*}(\lambda) = \|y - \sigma_{k^*}(\lambda)\|$.

1: Initialize $\bar{\varpi}_{\lambda^*} = \bar{\varpi}_{k^*}(\lambda^*), \quad g = \frac{\partial \bar{\varpi}_{k^*}}{\partial \lambda}(\lambda^*)^\top$
2: **repeat**
3: $\quad \lambda^{*\prime} \leftarrow \lambda^* - \alpha g / \|g\|, \qquad \bar{\varpi}_{\lambda^{*\prime}} \leftarrow \bar{\varpi}_{k^*}(\lambda^*)$
4: $\quad$ **if** $\bar{\varpi}_{\lambda^{*\prime}} < \bar{\varpi}_{\lambda^*}$ **then**
5: $\quad\quad \lambda^* \leftarrow \lambda^{*\prime}, \quad \bar{\varpi}_{\lambda^*} \leftarrow \bar{\varpi}_{\lambda^{*\prime}}$
6: $\quad\quad g \leftarrow \frac{\partial \bar{\varpi}_{k^*}}{\partial \lambda}(\lambda^{*\prime})^\top$
7: $\quad\quad \alpha \leftarrow 1.2\alpha$ {increase the stepsize}
8: $\quad$ **else**
9: $\quad\quad \alpha \leftarrow 0.5\alpha$ {decrease the stepsize}
10: $\quad$ **end if**
11: **until** $|\lambda^{*\prime} - \lambda^*| < \epsilon$
12: **if** $\lambda^* > \max(\mathbb{I}_{k*})$ **then**
13: $\quad k^* \leftarrow k^* + 1$
14: $\quad$ GOTO Line 1
15: **else if** $\lambda^* < \min(\mathbb{I}_{k*})$ **then**
16: $\quad k^* \leftarrow k^* - 1$
17: $\quad$ GOTO Line 1
18: **end if**

Algorithm 1 is ran at each time step in order to calculate the coordinate transformation. In Algorithm 1, lines 1 to 11 are the familiar monotonic gradient descent algorithm with stepsize adaptation [102]. Lines 12 to 18 determine which spline we are on. Depending on the path following application, if there is a known neighbourhood of the tangential velocity, the stepsize α in Algorithm 1 can be tuned in order to minimize the number of steps taken in the gradient descent and thus the computational load.

Remark 4.2.1. In [48], the path to be followed was assumed to be a one-dimensional embedded submanifold, which precludes self-intersecting curves. By employing Algorithm 1 under Assumption 4.2.1, this work removes that restriction and enlarges the class of admissible paths.

Consider first the case in which two curves σ_i, σ_j, $i, j \in \{1, \ldots, n_{\text{spline}}\}$ overlap. That is, there exist two closed, possibly disconnected, sets $\Lambda_i \subset \mathbb{I}_i$, $\Lambda_j \subset \mathbb{I}_j$ with $(\forall \lambda_i \in \Lambda_i)(\exists \lambda_j \in$

$\Lambda_j)\sigma_i(\lambda_i) = \sigma_j(\lambda_j)$. Suppose that at the current time-step, we have $k^* = i$ and $\lambda^* = \lambda_i \in \Lambda_i$. Then the coordinate transformation employed is completely determined by the curve σ_i. At the next time-step, the algorithm will search $\mathbb{I}_k$ for its next update. This local search implies that the algorithm will not "jump" to $\mathbb{I}_j$ despite the existence of intersection points.

The second case, the one in which there is single curve that is not injective, is handled similarly. Suppose there exist two values $\lambda_1, \lambda_2 \in \mathbb{I}_{k^*}$, $\lambda_1 < \lambda_2$, with $\sigma_{k^*}(\lambda_1) = \sigma_{k^*}(\lambda_2)$. By continuity, it is possible to find two open intervals $\mathbb{I}_1$, $\mathbb{I}_2 \subset \mathbb{I}_{k^*}$ with $\lambda_1 \in \mathbb{I}_1$ and $\lambda_2 \in \mathbb{I}_2$ and $\mathbb{I}_1 \cap \mathbb{I}_2 = \varnothing$. Now suppose that at the current time-step, we have $\lambda^* = \lambda_1$. At the next time-step, the algorithm will search $\mathbb{I}_1$ for its next update. This local search implies that the algorithm will not "jump" to $\mathbb{I}_2$ despite the existence of an intersection point.

In addition, when the output y is equidistant to multiple points on the path, the algorithm will again just choose the λ^* that is closest to the previous time-step's λ^*, due to the local nature of the monotonic gradient descent algorithm. •

Transversal States

The transversal states represent the cross track error to the path using the remaining FSF (normal) vectors as follows. The k^* notation will be dropped for brevity. The generalized Frenet-Serret (FS) vectors are constructed applying the Gram-Schmidt Orthonormalization process to the vectors $\sigma'(\lambda)$, $\sigma''(\lambda)$, $\ldots$, $\sigma^{(P)}(\lambda)$:

$$\mathbf{e}_j(\lambda) \coloneqq \frac{\bar{\mathbf{e}}_j(\lambda)}{\|\bar{\mathbf{e}}_j(\lambda)\|} \tag{4.8}$$

where

$$\bar{\mathbf{e}}_j(\lambda) \coloneqq \sigma^{(j)}(\lambda) - \sum_{i=1}^{j-1} \left\langle \sigma^{(j)}(\lambda), \mathbf{e}_i(\lambda) \right\rangle \mathbf{e}_i(\lambda),$$

for $j \in \{1, \ldots, P\}$. This formulation is well-defined by Assumption 4.1.2. For curves that violate Assumption 4.1.2, other ad-hoc orthonormalization processes can be used, see Remark 4.1.2. The transversal positions can be computed by projecting the error to the path, $y - \sigma(\lambda^*)$, onto each of the FS normal vectors:

$$\xi_1^{j-1} = \xi_1^{j-1}(x) \coloneqq \langle \mathbf{e}_j(\lambda^*), h(x) - \sigma(\lambda^*) \rangle|_{\lambda^* = \varpi \circ h(x)} \tag{4.9}$$

for $j \in \{2, \ldots, P\}$.

To compute the transversal velocities, we take the time derivative of (4.9). First note the identity $\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{e}_j(\lambda^*) = \mathbf{e}_j'(\lambda^*)\frac{\mathrm{d}\varpi}{\mathrm{d}y}\dot{y}$, and $\frac{\mathrm{d}\varpi}{\mathrm{d}y} = \frac{\sigma'(\lambda^*)^\top}{\left\|\sigma'_{k^*}(\lambda^*)\right\|^2}$ from Section 4.2.1. Then,

$$\begin{aligned}
\xi_2^{j-1} := \dot{\xi}_1^{j-1} = &\left(\mathbf{e}_j'(\lambda^*)\left.\frac{\mathrm{d}\varpi}{\mathrm{d}y}\right|_{y=h(x)} J(x_c)x_v\right)^\top (h(x)-\sigma(\lambda^*)) \\
&+ \mathbf{e}_j(\lambda^*)^\top\left(J(x_c)x_v - \sigma'(\lambda^*)\left.\frac{\mathrm{d}\varpi}{\mathrm{d}y}\right|_{y=h(x)} J(x_c)x_v\right) \\
&= \left(\mathbf{e}_j'(\lambda^*)\frac{\sigma'(\lambda^*)^\top}{\|\sigma'(\lambda^*)\|^2}J(x_c)x_v\right)^\top (h(x)-\sigma(\lambda^*)) \\
&\qquad + \mathbf{e}_j(\lambda^*)^\top\left(J(x_c)x_v - \mathbf{e}_1(\lambda^*)\mathbf{e}_1(\lambda^*)^\top J(x_c)x_v\right)
\end{aligned}$$

and using the simplification that $\langle \mathbf{e}_j, \mathbf{e}_1\rangle = 0$ for $j > 1$ and $\eta_2(x) = \langle \mathbf{e}_1, J(x_c)x_v\rangle$ yields

$$\begin{aligned}
\xi_2^{j-1} := \xi_2^{j-1}(x) = \dot{\xi}_1^{j-1} = &\frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|}\left\langle \mathbf{e}_j'(\lambda^*), h(x)-\sigma(\lambda^*)\right\rangle \\
&+ \left\langle \mathbf{e}_j(\lambda^*), J(x_c)x_v\right\rangle|_{\lambda^*=\varpi\circ h(x)}, \quad (4.10)
\end{aligned}$$

$j \in \{2,\dots,P\}$, and $\mathbf{e}_j'(\lambda^*)$ are given by the generalized FS equations (4.11)

$$\begin{bmatrix} \mathbf{e}_1'(\lambda) \\ \mathbf{e}_2'(\lambda) \\ \vdots \\ \mathbf{e}_{p-1}'(\lambda) \\ \mathbf{e}_p'(\lambda) \end{bmatrix} = \|\sigma'(\lambda)\| \begin{bmatrix} 0 & \mathcal{X}_1(\lambda) & \dots & 0 & 0 \\ -\mathcal{X}_1(\lambda) & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & \mathcal{X}_{p-1}(\lambda) \\ 0 & 0 & \dots & -\mathcal{X}_{p-1}(\lambda) & 0 \end{bmatrix} \begin{bmatrix} \mathbf{e}_1(\lambda) \\ \mathbf{e}_2(\lambda) \\ \vdots \\ \mathbf{e}_{p-1}(\lambda) \\ \mathbf{e}_p(\lambda) \end{bmatrix} \quad (4.11)$$

where $\mathcal{X}_i(\lambda) = \frac{\left\langle \mathbf{e}_i'(\lambda), \mathbf{e}_{i+1}(\lambda)\right\rangle}{\|\sigma'(\lambda)\|}$, $i \in \{1,...,P-1\}$ are the generalized curvatures [5]. If $P = 2$, then $\mathbf{e}_2'(\lambda^*)$ and $h(x) - \sigma(\lambda^*)$ are orthogonal and ξ_2^1 simplifies to $\langle \mathbf{e}_2(s(\lambda^*)), J(x_c)x_v\rangle$. Figure 4.3 illustrates the $P = 3$ case.

Diffeomorphism

In this section, we show that the tangential and transversal maps defined above can be used to construct a diffeomorphism. For brevity we again drop the spline k notation. First, define the lift of curve σ to $\mathbb{R}^n$ as

$$\Gamma := \{x \in \mathbb{R}^n : \xi_1^1(x) = \xi_1^2(x) = ... = \xi_1^{P-1}(x) = 0\}$$

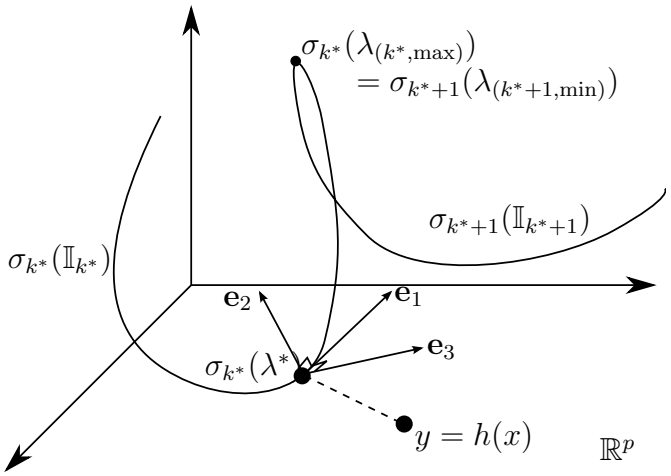

Figure 4.3: Example of the FSF when $P = 3$.

which represents the states x that correspond to the output y being on the curve. The above set is not necessarily a manifold since the curves considered can be self-intersecting or even self-overlapping, thus it is more general than the restrictive manifold case analysed in [48] and Chapter 3.

Lemma 4.2.2. *There exists a domain $U \subseteq \mathbb{R}^n$ with $\Gamma \subset U$ such that for all $x \in U$, and (4.3) is solved by Algorithm 1, the $2p$ functions (4.4), (4.5), (4.9), (4.10) have linearly independent differentials.*

Proof. Note that (4.3) is solved for by Algorithm 1, so if the output y is equidistant to multiple points on the path, then a λ^* is always well defined by the local search.

Next we write the differentials for each function.

$$\begin{aligned}
\frac{\partial \eta_1}{\partial x} &= \left[\tfrac{ds}{d\lambda}\tfrac{d\varpi}{dy} J(x_c) \quad , \quad 0_{1\times N}\right] \\
&= \left[\mathbf{e}_1(\lambda^*)^\top J(x_c) \quad , \quad 0_{1\times N}\right] \\
\frac{\partial \eta_2}{\partial x} &= \left[* \quad , \quad \mathbf{e}_1(\lambda^*)^\top J(x_c)\right] \\
\frac{\partial \xi_1^{j-1}}{\partial x} &= \Bigg[\Bigg(\left(h(x)-\sigma(\lambda^*)\right)^\top \left(\frac{1}{\|\sigma'(\lambda^*)\|}\mathbf{e}_j'(\lambda^*)\mathbf{e}_1(\lambda^*)^\top\right) \\
&\qquad + \left(\mathbf{e}_j(\lambda^*)^\top\right)\Bigg) J(x_c) \quad , \quad 0_{1\times N}\Bigg] \\
\frac{\partial \xi_2^{j-1}}{\partial x} &= \Bigg[* \quad , \quad \Bigg(\left(h(x)-\sigma(\lambda^*)\right)^\top \left(\frac{1}{\|\sigma'(\lambda^*)\|}\mathbf{e}_j'(\lambda^*)\mathbf{e}_1(\lambda^*)^\top\right) \\
&\qquad + \left(\mathbf{e}_j(\lambda^*)^\top\right)\Bigg) J(x_c)\Bigg]
\end{aligned}$$

where $*$ denotes a vector of dimension $1\times N$. Looking at the transversal terms, the product $\frac{1}{\|\sigma'(\lambda^*)\|}\left(h(x)-\sigma(\lambda^*)\right)^\top \left(\mathbf{e}_j'(\lambda^*)\right)$ produces a scalar, call it α. Thus the terms become

$$\begin{aligned}
\frac{\partial \xi_1^{j-1}}{\partial x} &= \left[\left(\alpha\mathbf{e}_1(\lambda^*)^\top + \mathbf{e}_j(\lambda^*)^\top\right) J(x_c) \quad , \quad 0_{1\times N}\right] \\
\frac{\partial \xi_2^{j-1}}{\partial x} &= \left[* \quad , \quad \left(\alpha\mathbf{e}_1(\lambda^*)^\top + \mathbf{e}_j(\lambda^*)^\top\right) J(x_c)\right]
\end{aligned}$$

Since the domain U does not include singularity points of $J(x_c)$ by the class of systems, the differentials will be linearly independent if the vectors $\mathbf{e}_i, i = 1, ..., p$ are orthogonal. These vectors are orthogonal because they are the FS orthonormal basis vectors. □

By [52, Proposition 5.1.2], for each $x^* \in U$, it is possible to find a function $\varphi : \mathbb{R}^n \to \mathbb{R}^{n-2p}$ so that by the Inverse Function Theorem [52] there exists a neighbourhood U_{x^*} such that the mapping

$$\begin{aligned}
T : U_{x^*} \subseteq \mathbb{R}^n &\to T(U_{x^*}) \subseteq \mathbb{R}^n \\
x &\mapsto (\eta_1, \eta_2, \xi_1^1, \xi_2^1, \ldots, \xi_1^{p-1}, \xi_2^{p-1}, \zeta) \\
&= (\eta_1(x), \eta_2(x), \xi_1^1(x), \xi_2^1(x), \ldots, \xi_1^{p-1}(x), \xi_2^{p-1}(x), \varphi(x))
\end{aligned} \tag{4.12}$$

is a diffeomorphism onto its image. In practice the domain of T can often be extended to all of U (see Section 4.3).

Let us bring back the spline notation k to explicitly indicate that the coordinate transformation is dependent on the curve k^*, which is determined using Algorithm 1. Namely, the coordinate transformation can be viewed as a collection of n_{spline} maps

$$T_k : U_k \subseteq \mathbb{R}^n \to T_k(U_k) \subseteq \mathbb{R}^n$$
$$x \mapsto (\eta_{1_k}(x), \eta_{2_k}(x), \xi^1_{1_k}(x), \xi^1_{2_k}(x), \ldots, \xi^{p-1}_{1_k}(x), \xi^{p-1}_{2_k}(x), \varphi_k(x)) =$$

$$\begin{bmatrix} s_k(\lambda^*) + \sum_{k=1}^{k-1} s_k(\lambda_{(k,max)}) \\ \langle \mathbf{e}_{1_k}(\lambda^*), J(x_c)x_v \rangle \\ \langle \mathbf{e}_{2_k}(\lambda^*), h(x) - \sigma_k(\lambda^*) \rangle \\ \frac{\eta_{2_k}(x)}{\|\sigma'_k(\lambda^*)\|} \left\langle \mathbf{e}'_{2_k}(\lambda^*), h(x) - \sigma_k(\lambda^*) \right\rangle + \langle \mathbf{e}_{2_k}(\lambda^*), J(x_c)x_v \rangle \\ \vdots \\ \langle \mathbf{e}_{p_k}(\lambda^*), h(x) - \sigma_k(\lambda^*) \rangle \\ \frac{\eta_{2_k}(x)}{\|\sigma'_k(\lambda^*)\|} \left\langle \mathbf{e}'_{p_k}(\lambda^*), h(x) - \sigma_k(\lambda^*) \right\rangle + \langle \mathbf{e}_{p_k}(\lambda^*), J(x_c)x_v \rangle \\ \varphi_k(x) \end{bmatrix}$$

where $\lambda^* = \varpi_k \circ h(x)$, $k \in \{1, \ldots, n_{\text{spline}}\}$. As we show later, satisfying a path following task depends on a state feedback involving the η and ξ states. Thus, it is desirable for η and ξ to remain continuous from one curve on the path to the next, i.e., as k changes via Algorithm 1, to avoid wear on equipment due to jerky motions. The following proposition will show this.

Proposition 4.2.3. *The coordinate transformation*

$$\bar{T}_k(x) := [\eta_{1_k}(x), \eta_{2_k}(x), \xi^1_{1_k}(x), \xi^1_{2_k}(x), ..., \xi^{p-1}_{1_k}(x), \xi^{p-1}_{2_k}(x)]^\top \tag{4.13}$$

is continuous (except for $\eta_{1_k}(x)$ for closed curves) as k changes by Algorithm 1.

Proof. Based on Algorithm 1, k changes when the output of the system crosses the hyperplane shown in Figure 4.4. Thus we must show that $\bar{T}_k(x) = \bar{T}_{k+1}(x)$ for x that correspond to the output $y = h(x)$ on this hyperplane. This is equivalent to showing

$$(\forall x \in \{x \in \mathbb{R}^n | x = (\varpi_k \circ h(x))^{-1}(\lambda_{(k,\max)})\}) \quad \bar{T}_k(x) = \bar{T}_{k+1}(x).$$

First, the coordinate transformation relies on the computation of $\lambda^* = \varpi_k \circ h(x)$, which can be easily verified to be continuous in x for a fixed k. The tangential position for a fixed k is continuous since $s_k(\lambda^*)$ is a function of $\sigma'(\lambda^*)$, which is continuous by Assumption 4.1.1. On the hyperplane, $\eta_{1_k}(x) = s_k(\lambda_{(k,max)}) + \sum_{i=1}^{k-1} s_i(\lambda_{(k,max)}) = \sum_{i=1}^{k} s_i(\lambda_{(k,max)}) = \eta_{1_{k+1}}(x)$

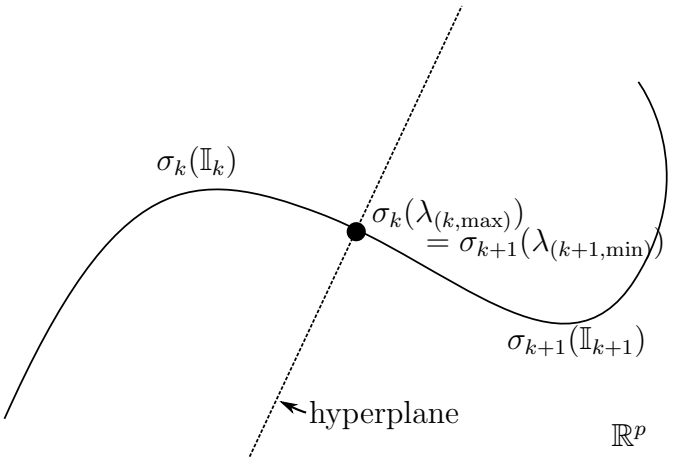

Figure 4.4: Spline k and spline $k+1$, their stitch point $\sigma_k(\lambda_{(k,\max)}) = \sigma_{k+1}(\lambda_{(k+1,\min)})$, and the hyperplane.

and is still continuous at the transition. For closed curves, when k transitions between 1 and n_{spline}, the value of $\eta_{1_k}(x)$ resets between 0 and $\sum_{i=1}^{n_{\text{spline}}} s_i(\lambda_{(k,max)})$, respectively.

For the remaining states, $h(x), J(x)$ are continuous by the class of systems (2.3). The remaining terms are the FS vectors and its derivatives. For a fixed k, the FS vectors $\mathbf{e}_{j_k}(\lambda^*)$ depends on $\sigma^{(j)}(\lambda^*)$ for $j = 1, ..., p$ by (4.8), and thus will be continuous by Assumption 4.1.1. Furthermore, the derivative of the FS vectors are governed by the FS equations for general spaces (4.11) , and depend only on the generalized curvatures $\mathcal{X}_i(\lambda) = \frac{\langle \mathbf{e}_i'(\lambda), \mathbf{e}_{i+1}(\lambda) \rangle}{\|\sigma'(\lambda)\|}$, $i \in \{1, ..., p-1\}$. Thus, the highest derivative that appears again is $\sigma^{(p)}(\lambda^*)$ and so the FS vector derivatives will remain continuous for a fixed spline k. At the transition point, $\mathbf{e}_{j_k}(\lambda_{(k,\max)}) = \mathbf{e}_{j_{k+1}}(\lambda_{(k+1,\min)})$ holds if and only if $\sigma_k^{(j)}(\lambda_{(k,\max)}) = \sigma_{k+1}^{(j)}(\lambda_{(k+1,\min)})$ for $j \in \{1, \ldots, P\}$. This is satisfied by Assumption 4.1.1. To show that $\mathbf{e}'_{j_k}(\lambda_{(k,\max)}) = \mathbf{e}'_{j_{k+1}}(\lambda_{(k+1,\min)})$ for $j \in \{1, \ldots, P\}$, by (4.11) it suffices to show that the generalized curvatures are continuous at the transition, i.e., $\mathcal{X}_{i_k}(\lambda_{(k,\max)}) = \mathcal{X}_{i_{k+1}}(\lambda_{(k+1,\min)})$ for $i \in \{1, \ldots, p-1\}$. By definition of the curvatures, this then simplifies to showing $\mathbf{e}'_{j_k}(\lambda_{(k,max)}) = \mathbf{e}'_{j_{k+1}}(\lambda_{(k+1,min)})$ for $j \in \{1, ..., p-1\}$. These derivatives can be expressed by evaluating the derivative of (4.8), and so the equation will hold if $\sigma_k^{(j)}(\lambda_{(k,\max)}) = \sigma_{k+1}^{(j)}(\lambda_{(k+1,\min)})$ for $j \in \{1, \ldots, P\}$. Again by Assumption 4.1.1 this holds. □

4.2.2 Dynamics and Control

The dynamics in transformed coordinates are

$$\begin{aligned}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= L_f^2 \eta_1(x) + L_g L_f \eta_1(x) u \\
\dot{\xi}_1^1 &= \xi_2^1 \\
\dot{\xi}_2^1 &= L_f^2 \xi_1^1(x) + L_g L_f \xi_1^1(x) u \\
&\vdots \\
\dot{\xi}_1^{p-1} &= \xi_2^{p-1} \\
\dot{\xi}_2^{p-1} &= L_f^2 \xi_1^{p-1}(x) + L_g L_f \xi_1^{p-1}(x) u \\
\dot{\zeta} &= L_f \varphi(x) + L_g \varphi(x) u =: f_\zeta(x, u)
\end{aligned} \tag{4.14}$$

where $x = T^{-1}(\eta, \xi, \zeta)$. The Lie derivatives can be computed given the state maps from Section 4.2.1 and the class of system (2.3) as follows.

Recall

$$\frac{\mathrm{d}\varpi_{k^*}}{\mathrm{d}y} = \frac{\sigma'(\lambda^*)^\top}{\|\sigma'(\lambda^*)\|^2} = \frac{\mathbf{e}_1^\top(\lambda^*)}{\|\sigma'(\lambda^*)\|}\bigg|_{\lambda^* = \varpi \circ h(x)}$$

For the η dynamics, recall:

$$\eta_2 = \mathbf{e}_1(\lambda^*)^\mathsf{T}|_{\lambda^* = \varpi \circ h(x)} J(x_c) x_v$$

By product rule and chain rule,

$$\begin{aligned}
\dot{\eta}_2 &= \mathbf{e}_1'(\lambda^*)^\mathsf{T} \frac{\mathrm{d}\lambda^*}{\mathrm{d}t} J(x_c) x_v + \mathbf{e}_1'(\lambda^*)^\mathsf{T} \left(\frac{\partial J(x_c) x_v}{\partial x} \dot{x} \right) \\
&= \mathbf{e}_1'(\lambda^*)^\mathsf{T} \frac{\mathrm{d}\lambda^*}{\mathrm{d}t} J(x_c) x_v + \mathbf{e}_1'(\lambda^*)^\mathsf{T} \left(\frac{\partial J(x_c) x_v}{\partial x_c} x_v + J(x_c) \left(f_v(x) + g_v(x) u \right) \right) \\
\frac{\mathrm{d}\lambda^*}{\mathrm{d}t} &= \frac{\mathrm{d}\varpi_{k^*}}{\mathrm{d}y} \dot{y} = \frac{\mathbf{e}_1^\top(\lambda^*)}{\|\sigma'(\lambda^*)\|} J(x_c) x_v = \frac{\eta_2}{\|\sigma'(\lambda^*)\|} \\
\Rightarrow \dot{\eta}_2 &= \mathbf{e}_1'(\lambda^*)^\mathsf{T} \frac{\eta_2}{\|\sigma'(\lambda^*)\|} J(x_c) x_v + \mathbf{e}_1'(\lambda^*)^\mathsf{T} \left(\frac{\partial J(x_c) x_v}{\partial x_c} x_v + J(x_c) \left(f_v(x) + g_v(x) u \right) \right) \\
&= \underbrace{\mathbf{e}_1'(\lambda^*)^\mathsf{T} \frac{\eta_2}{\|\sigma'(\lambda^*)\|} J(x_c) x_v + \mathbf{e}_1'(\lambda^*)^\mathsf{T} \left(\frac{\partial J(x_c) x_v}{\partial x_c} x_v + J(x_c) f_v(x) \right)}_{L_f \eta_2(x) = L_f^2 \eta_1(x)} + \underbrace{\mathbf{e}_1'(\lambda^*)^\mathsf{T} J(x_c) g_v(x)}_{L_g \eta_2(x) = L_g L_f \eta_1(x)} u
\end{aligned}$$

For the ξ dynamics, recall:

$$\begin{aligned}\xi_2^{j-1} &= \langle \mathbf{e}_j(\lambda^*), J(x_c)x_v\rangle + \frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|}\left\langle \mathbf{e}_j'(\lambda^*), h(x)-\sigma(\lambda^*)\right\rangle|_{\lambda^*=\varpi\circ h(x)} \\ &= \langle \dot{y},\ \mathbf{e}_j(\lambda^*)\rangle + \left\langle y-\sigma(\lambda^*),\ \mathbf{e}_j'(\lambda^*)\frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|}\right\rangle\end{aligned}$$

Note that

$$\begin{aligned}\frac{\mathrm{d}\mathbf{e}_j(\lambda^*)}{\mathrm{d}t} &= \mathbf{e}_j'(\lambda^*)\frac{\mathrm{d}\lambda^*}{\mathrm{d}t} = \mathbf{e}_j'(\lambda^*)\frac{\eta_2}{\|\sigma'(\lambda^*)\|} \\ \frac{\mathrm{d}\frac{1}{\|\sigma'(\lambda^*)\|}}{\mathrm{d}t} &= \frac{\mathrm{d}\frac{1}{\|\sigma'(\lambda^*)\|}}{\mathrm{d}\lambda^*}\frac{\mathrm{d}\lambda^*}{\mathrm{d}t} = -\frac{\langle\sigma'(\lambda^*),\ \sigma''(\lambda^*)\rangle}{\|\sigma'(\lambda^*)\|^3}\frac{\eta_2}{\|\sigma'(\lambda^*)\|} = -\eta_2\frac{\langle\sigma'(\lambda^*),\ \sigma''(\lambda^*)\rangle}{\|\sigma'(\lambda^*)\|^4}\end{aligned}$$

By product rule and chain rule,

$$
\begin{aligned}
\dot{\xi}_2^{j-1} &= \langle \ddot{y},\ \mathbf{e}_j(\lambda^*)\rangle + \left\langle \dot{y},\ \mathbf{e}_j'(\lambda^*)\frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|}\right\rangle + \\
&\left\langle \dot{y} - \sigma'(\lambda^*)\frac{\eta_2}{\|\sigma'(\lambda^*)\|},\ \mathbf{e}_j'(\lambda^*)\frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|}\right\rangle + \\
&\left\langle y - \sigma(\lambda^*),\ \mathbf{e}_j''(\lambda^*)\left(\frac{\eta_2}{\|\sigma'(\lambda^*)\|}\right)^2 + \mathbf{e}_j'(\lambda^*)\frac{\dot{\eta_2}}{\|\sigma'(\lambda^*)\|} - \mathbf{e}_j'(\lambda^*)\eta_2^2\frac{\langle\sigma'(\lambda^*),\ \sigma''(\lambda^*)\rangle}{\|\sigma'(\lambda^*)\|^4}\right\rangle \\
&= \left\langle \frac{\partial J(x_c)x_v}{\partial x_c}x_v + J(x_c)\left(f_v(x) + g_v(x)u\right),\ \mathbf{e}_j(\lambda^*)\right\rangle + \left\langle J(x_c)x_v,\ \mathbf{e}_j'(\lambda^*)\frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|}\right\rangle + \\
&\left\langle J(x_c)x_v - \mathbf{e}_1(\lambda^*)\eta_2,\ \mathbf{e}_j'(\lambda^*)\frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|}\right\rangle + \\
&\left\langle y - \sigma(\lambda^*),\ \mathbf{e}_j''(\lambda^*)\left(\frac{\eta_2}{\|\sigma'(\lambda^*)\|}\right)^2 + \mathbf{e}_j'(\lambda^*)\frac{1}{\|\sigma'(\lambda^*)\|}\left(L_f^2\eta_1(x) + L_gL_f\eta_1(x)u\right) - \right. \\
&\left.\mathbf{e}_j'(\lambda^*)\eta_2^2\frac{\langle\sigma'(\lambda^*),\ \sigma''(\lambda^*)\rangle}{\|\sigma'(\lambda^*)\|^4}\right\rangle \\
&= \left\langle \frac{\partial J(x_c)x_v}{\partial x_c}x_v + J(x_c)\left(f_v(x) + g_v(x)u\right),\ \mathbf{e}_j(\lambda^*)\right\rangle + \\
&\left\langle 2J(x_c)x_v - \mathbf{e}_1(\lambda^*)\eta_2,\ \mathbf{e}_j'(\lambda^*)\frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|}\right\rangle + \\
&\left\langle y - \sigma(\lambda^*),\ \mathbf{e}_j''(\lambda^*)\left(\frac{\eta_2}{\|\sigma'(\lambda^*)\|}\right)^2 + \mathbf{e}_j'(\lambda^*)\frac{1}{\|\sigma'(\lambda^*)\|}\left(L_f^2\eta_1(x) + L_gL_f\eta_1(x)u\right) - \right. \\
&\left.\mathbf{e}_j'(\lambda^*)\eta_2^2\frac{\langle\sigma'(\lambda^*),\ \sigma''(\lambda^*)\rangle}{\|\sigma'(\lambda^*)\|^4}\right\rangle \\
&= \left\langle \frac{\partial J(x_c)x_v}{\partial x_c}x_v + J(x_c)f_v(x),\ \mathbf{e}_j(\lambda^*)\right\rangle + \\
&\left\langle 2J(x_c)x_v - \mathbf{e}_1(\lambda^*)\eta_2,\ \mathbf{e}_j'(\lambda^*)\frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|}\right\rangle + \\
&\left\langle y - \sigma(\lambda^*),\ \mathbf{e}_j''(\lambda^*)\left(\frac{\eta_2}{\|\sigma'(\lambda^*)\|}\right)^2 + \mathbf{e}_j'(\lambda^*)\frac{1}{\|\sigma'(\lambda^*)\|}L_f^2\eta_1(x) - \mathbf{e}_j'(\lambda^*)\eta_2^2\frac{\langle\sigma'(\lambda^*),\ \sigma''(\lambda^*)\rangle}{\|\sigma'(\lambda^*)\|^4}\right\rangle + \\
&\langle J(x_c)g_v(x)u,\ \mathbf{e}_j(\lambda^*)\rangle + \left\langle y - \sigma(\lambda^*),\ \mathbf{e}_j'(\lambda^*)\frac{1}{\|\sigma'(\lambda^*)\|}L_gL_f\eta_1(x)u\right\rangle
\end{aligned}
$$

Thus,

$$
\begin{aligned}
L_f^2\eta_1(x) &= \frac{\eta_2}{\|\sigma'(\lambda^*)\|}\left\langle \mathbf{e}_1'(\lambda^*), J(x_c)x_v\right\rangle + \left\langle \mathbf{e}_1(\lambda^*), \frac{\partial(J(x_c)x_v)}{\partial x_c}x_v + J(x_c)f_v(x)\right\rangle \\
L_gL_f\eta_1(x) &= \mathbf{e}_1(\lambda^*)^\top J(x_c)g_v(x_c) \\
L_f^2\xi_1^{j-1}(x) &= \left\langle \mathbf{e}_j(\lambda^*), \frac{\partial(J(x_c)x_v)}{\partial x_c}x_v + J(x_c)f_v(x)\right\rangle + \frac{\eta_2}{\|\sigma'(\lambda^*)\|}\left\langle \mathbf{e}_j'(\lambda^*), 2J(x_c)x_v - \mathbf{e}_1\eta_2\right\rangle \\
&\quad + \left\langle h(x)-\sigma(\lambda^*), \mathbf{e}_j''(\lambda^*)\left(\frac{\eta_2}{\|\sigma'(\lambda^*)\|}\right)^2 + \mathbf{e}_j'(\lambda^*)\left(\frac{L_gL_f\eta_1(x)}{\|\sigma'(\lambda^*)\|} - \eta_2^2\frac{\langle\sigma'(\lambda^*),\ \sigma''(\lambda^*)\rangle}{\|\sigma'(\lambda^*)\|^4}\right)\right\rangle \\
L_gL_f\xi_1^{j-1}(x) &= \mathbf{e}_j(\lambda^*)^\top J(x_c)g_v(x_c) + (h(x)-\sigma(\lambda^*))^\top\mathbf{e}_j'(\lambda^*)\frac{L_gL_f\eta_1(x)}{\|\sigma'(\lambda^*)\|}
\end{aligned}
$$

for $j \in \{2,\dots,P\}$ and $\lambda^* = \varpi \circ h(x)$. Physically, see their definitions in Section 4.2.1, the η-subsystem represents the dynamics tangent to the path, and the ξ-subsystem represents the dynamics transversal to the path. Thus, the redundant dynamics f_ζ represent the dynamics of the system (2.3) that do not produce any output motion.

It then follows from Lemma 4.2.2 that the decoupling matrix

$$
\beta(x) \coloneqq \begin{bmatrix} L_gL_f\eta_1(x) \\ L_gL_f\xi_1^1(x) \\ \vdots \\ L_gL_f\xi_1^{p-1}(x)\end{bmatrix} = \begin{bmatrix} \mathbf{e}_1(\lambda^*)^\top \\ \frac{(h(x)-\sigma(\lambda^*))^\top}{\|\sigma'(\lambda^*)\|}\mathbf{e}_2'(\lambda^*)\mathbf{e}_1(\lambda^*)^\top + \mathbf{e}_2(\lambda^*)^\top \\ \vdots \\ \frac{(h(x)-\sigma(\lambda^*))^\top}{\|\sigma'(\lambda^*)\|}\mathbf{e}_p'(\lambda^*)\mathbf{e}_1(\lambda^*)^\top + \mathbf{e}_p(\lambda^*)^\top\end{bmatrix} J(x_c)g_v(x_c)
$$

has full rank for all $x \in U$. This is equivalent to saying that for each $x \in U$, system (2.3) with output $(\eta_1(x), \xi_1^1(x), \dots, \xi_1^{p-1}(x))$ has a vector relative degree of $\{2,\dots,2\}$ and we can perform partial feedback linearization. Let

$$
\alpha(x) \coloneqq \begin{bmatrix} L_f^2\eta_1(x) & L_f^2\xi_1^1(x) & \dots & L_f^2\xi_1^{P-1}(x)\end{bmatrix}^\top \tag{4.15}
$$

and introduce an auxiliary input $v = \begin{bmatrix} v_\eta & v_\xi\end{bmatrix}^\top \in \mathbb{R}\times\mathbb{R}^{P-1}$ such that

$$
v = \beta(x)u + \alpha(x) \tag{4.16}
$$

the dynamics (4.14) become

$$
\dot\zeta = f_\zeta(x,u) \qquad \begin{matrix}\dot\eta_1 = \eta_2 \\ \dot\eta_2 = v_\eta\end{matrix} \qquad \begin{matrix}\dot\xi_1^j = \xi_2^j \\ \dot\xi_2^j = v_{\xi^j}\end{matrix}
$$

where $j \in \{1, \ldots, P-1\}$, $x = T^{-1}(\eta, \xi, \zeta)$, and $f_\zeta(x, u)$ is used for brevity (see Equation (3.7)).

In U the ξ-subsystem is linear and controllable, and can be stabilized to ensure attractiveness and invariance of the path $\mathcal{P}$. One can do this using a simple PD controller $v_{\xi^j} = -K_P^j \xi_1^j - K_D^j \xi_2^j$, $j \in \{1, \ldots, P-1\}$, for positive K_P^j, K_D^j. Another transversal controller is (3.20) which is used on the experimental platform to overcome modelling uncertainties. Doing so will stabilize the largest controlled invariant subset of Γ known as the path following manifold [48], see Chapter 3. The η-subsystem is also linear and controllable, thus a tangential controller can be designed for v_η to track some desired tangential position or velocity profile $\eta^{\text{ref}}(t)$. A block diagram of the entire control system can be seen in Figure 4.1.

Remark 4.2.4. Suppose a desired tangential velocity trajectory is to be tracked, $\eta_2^{\text{ref}}(t)$. In the case that Assumption 4.1.1 fails for $i > 0$, $\eta_2^{\text{ref}}(t)$ can be designed such that it approaches zero at the junction of the splines, then ramping up to, for instance, some desired steady state value just after the junction. This way, the control input remains continuous and the system avoids unwanted transients associated with the differential discontinuities of the path. •

4.2.3 Redundancy Resolution

As we saw in the last chapter, if $N = P$, there is no φ map and thus no redundant dynamics. However, if $N > P$, then the system has internal dynamics. Unlike kinematic redundancy resolution techniques [98], we seek a scheme at the dynamic level that can incorporate actuator constraints while ensuring the internal dynamics of the system are bounded. The approach should also work when the joint-space and task-space dynamics are not necessarily related by a static input conversion map (as is the case for torque-input model of a robot manipulator [58]). Such systems include a combined motor and manipulator model (see Section 4.3, where a dynamic relation exists between motor input voltage and end-effector force), or mobile manipulator systems.

Suppose the ξ-subsystem is stabilized and the η-subsystem is tracking a desired profile $\eta^{\text{ref}}(t)$ by some outer loop controller for v. The zero-dynamics correspond to the redundant dynamics $\dot{\zeta} = f_\zeta(x, u)$ under the transform (4.16), $x = T^{-1}(\eta^{\text{ref}}(t), 0, \zeta)$ and $v_\eta = \dot{\eta}_2^{\text{ref}}(t), v_\xi = 0$. In order to ensure a well-behaved response, we must ensure boundedness of the zero dynamics.

With v generated by some outer loop controller, there is some freedom in the choice of the control input u under the feedback transform (4.16) (recall $v \in \mathbb{R}^p$, $u \in \mathbb{R}^{N=n/2}$ and

$n > 2p$); this freedom can be used to enforce boundedness of the zero dynamics. Since the relation from u to the states is dynamic, one can use dynamic programming to achieve some desired (and bounded) trajectory in the zero dynamics. However, this in general is quite complicated. This is why previous works on path following have relied on the zero dynamics being absent, assumed to be stable, and are analysed in relatively simple cases [28]). To make things simple, consider the static optimization

$$\min_u \left(u - r(x)\right)^\top W \left(u - r(x)\right) \tag{4.17}$$

under the constraint (4.16). It is a static minimization of a quadratic function of u under a linear constraint, for which a closed form solution for u can be solved for using Lagrange multipliers. The matrix $W \in \mathbb{R}^{n\times n}$ is a positive definite weight matrix, and the function $r : \mathbb{R}^n \to \mathbb{R}^n$ is used to bias the control input u to achieve desired behaviour in the zero dynamics. For example, if $r(x) \equiv 0$ and W is the identity matrix, then we are minimizing control effort while staying on the path and following some desired $\eta^{\text{ref}}(t)$.

Most real systems have actuation and configuration limits. Thus, another useful criteria for redundancy resolution is to design u so that the configuration variables x_c are driven away from their limits. For example, consider a joint on a manipulator. If this joint is at its minimum value, setting the control effort corresponding to this joint to be the maximum control effort will *likely* increase the value of the joint, thereby pushing it away from the negative joint limit. The corresponding r function to achieve this behaviour in (4.17) is

$$r(x_c)_i = -\frac{u_{i_{max}} - u_{i_{min}}}{x_{c_{i_{max}}} - x_{c_{i_{min}}}}(x_{c_i} - x_{c_{i_{min}}}) + u_{i_{max}}, \tag{4.18}$$

for $i \in \{1, \ldots, N\}$, or graphically shown below in Figure 4.5.

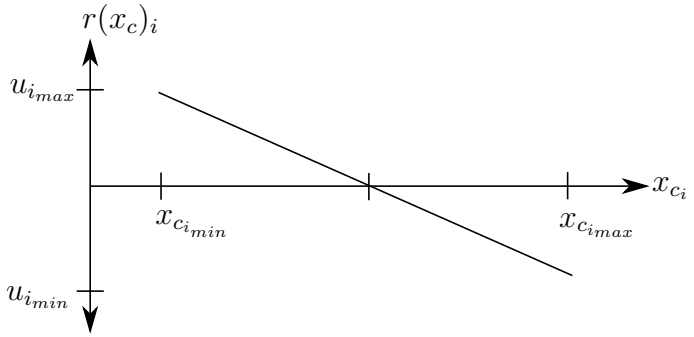

Figure 4.5: The function $r(x_c)$ for avoiding joint and actuation limits.

Using Lagrangian Multipliers, the solution to (4.17) under constraint (4.16) is

$$u = \beta^{\dagger}(x)(v - \alpha(x)) + \left(I_{n\times n} - \beta^{\dagger}(x)\beta(x)\right) r(x) \tag{4.19}$$

where $\beta^{\dagger}(x) = W^{-1}\beta(x)^{\top}\left(\beta(x)W^{-1}\beta(x)^{\top}\right)^{-1}$ and $I_{n\times n}$ is the identity matrix.

Conjecture 4.2.5. *Using (4.19) and (4.18) to generate the control input u results in boundedness of the redundant dynamics $\dot{\zeta} = f_{\zeta}(x, u)$, while achieving path following and maintaining joint limits when possible, in the situation that system (2.1) has inherent damping.*

Conjecture 4.2.5 has been tested and verified on the following two examples 3-4, and on our experimental platform in Section 4.3.

Example 3. (2-DOF linear system) Consider the linear system in Figure 4.6 with output $y \in \mathbb{R}$ taken to be the position of the top-most block. Let m_i, b_i and u_i be the mass, coefficient of friction, and force acting on each respective block. The corresponding

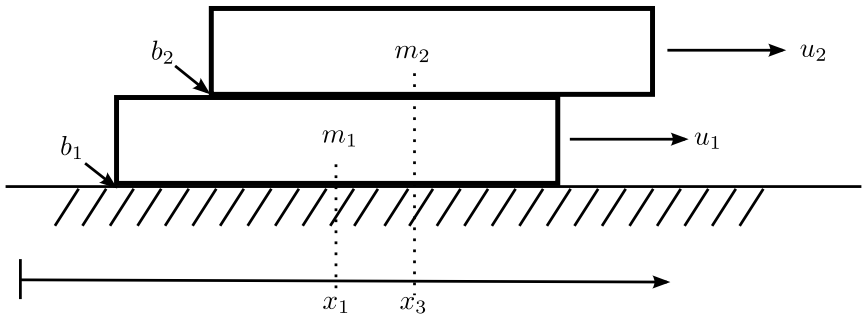

Figure 4.6: A Planar 2-DOF fully-actuated system where $m_1, m_2, b_1, b_2 > 0$.

dynamics are

$$\begin{aligned}
\dot{x} &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{b_1+b_2}{m_1} & 0 & \frac{b_2}{m_1} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{b_2}{m_2} & 0 & -\frac{b_2}{m_2} \end{bmatrix} x + \begin{bmatrix} 0 & 0 \\ \frac{1}{m_1} & 0 \\ 0 & 0 \\ 0 & \frac{1}{m_2} \end{bmatrix} u \\
&=: Ax + Bu \\
y &= \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} x.
\end{aligned}$$

For this particular example, $N = 2, P = 1$. Thus $P-1 = 0$ and there are no transversal dynamics. Therefore, under our path following context, the position of the output along the path is $\eta_1 = x_3$. The remaining function $\varphi : \mathbb{R}^4 \to \mathbb{R}^2$ is chosen to complete a diffeomorphism. For instance

$$\begin{aligned} T : \mathbb{R}^4 &\to \mathbb{R}^4 \\ x &\mapsto (\eta_1(x), \eta_2(x), \zeta_1(x), \zeta_2(x)) = (x_3, x_4, x_1, x_2) \end{aligned}$$

yields the dynamics in the transformed coordinates

$$\begin{aligned} \dot{\eta}_1 &= \eta_2 \\ \dot{\eta}_2 &= \frac{b_2}{m_2}(\zeta_2 - \eta_2) + \frac{1}{m_2}u_2 =: v_\eta \\ \dot{\zeta}_1 &= \zeta_2 \\ \dot{\zeta}_2 &= -\left(\frac{b_1 + b_2}{m_1}\right)\zeta_2 + \frac{b_2}{m_1}\eta_2 + \frac{1}{m_1}u_1. \end{aligned}$$

The corresponding terms in the feedback transform (4.16) are $\alpha(x) = \frac{b_2}{m_2}(x_2 - x_4)$ and $\beta(x) = \begin{bmatrix} 0 & \frac{1}{m_2} \end{bmatrix}$. Now assume the desired tangential reference trajectory is

$$\eta^{\text{ref}}(t) = \begin{bmatrix} \eta_1^{\text{ref}} \\ 0 \end{bmatrix}$$

for some constant η_1^{ref}. The control law $v_\eta = K_P(\eta_1^{\text{ref}} - \eta_1) + K_D(-\eta_2)$ for positive K_P, K_D will force η to converge to η^{ref}. Thus the zero dynamics are the redundant dynamics when $\eta = \eta^{\text{ref}}$ and $\alpha(x) + \beta(x)u = v_\eta = 0$, i.e.

$$\begin{aligned} \dot{\zeta}_1 &= \zeta_2 \\ \dot{\zeta}_2 &= -\left(\frac{b_1 + b_2}{m_1}\right)\zeta_2 + \frac{1}{m_1}u_1. \end{aligned}$$

Using the controller (4.19) yields the zero dynamics

$$\begin{aligned} \dot{\zeta}_1 &= \zeta_2 \\ \dot{\zeta}_2 &= -\left(\frac{b_1 + b_2}{m_1}\right)\zeta_2 + \frac{1}{m_1}r(x)_1. \end{aligned}$$

and thus the zero dynamics can be made globally exponentially stable if, for example, $r(x)_1$ is a linear function of $\zeta_1 = x_1$ with negative slope. This is precisely the case with

our proposed r function (4.18). Furthermore, the equilibrium point will be $\zeta_2 = 0$, and the value of $x = T^{-1}(\eta^{\text{ref}}, \zeta_1, \zeta_2 = 0)$ for which $r(x)_1$ vanishes. In our proposed r function, this occurs when $\zeta_1 = x_1 = \frac{u_{1_{max}}}{u_{1_{max}} - u_{1_{min}}}(x_{1_{max}} - x_{1_{min}}) + x_{1_{min}}$. If $u_{1_{max}} = -u_{1_{min}}$, which usually is the case for practical examples, then the equilibrium point is $\zeta_1 = \frac{1}{2}(x_{1_{min}} + x_{1_{max}})$, the average value of the joint limits. Thus, the path following constraints are met and the zero dynamics are stabilized while achieving the objective of staying within joint limits using (4.18). △

Example 4. (3-DOF planar manipulator) Consider a planar 3-DOF robot manipulator and the desired path of a circle, and the point $\eta_1 = 0$ on the path is to be stabilized, shown in Figure 4.7.

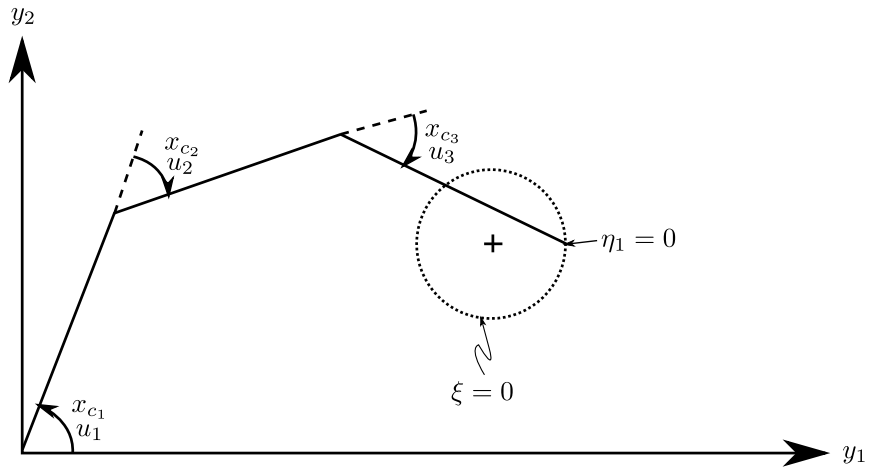

Figure 4.7: A Planar 3-DOF fully-actuated manipulator whose task is to stabilize a point on a circle.

Here $N = 3, P = 2$ and thus the dimension of redundancy is 1. Let's further assume that each link has unit mass and inertia, with the center of mass of each link be at the midpoint of each link. Without loss of generality, we assume gravity is ignored. The dynamics can be derived as per [98] to yield the form (2.1).

Let the redundant state be $\zeta_1 = \zeta_1(x) = x_{c_1} + x_{c_2} + x_{c_3}$, the sum of the joint angles, the angle the end-effector makes with the ground. Thus $\zeta_2 = x_{v_1} + x_{v_2} + x_{v_3}$, and $\dot{\zeta}_2 = \sum_{i=1}^{3} [f(x) + g(x)u]_i \big|_{x=T^{-1}(\eta,\xi,\zeta)}$. The zero dynamics correspond to these redundant dynamics when $\xi = 0, \eta = 0$ and thus $v = 0$. Stability of the zero dynamics depends on the choice of u.

Suppose that u is chosen as in (4.19) with r from (4.18). The parameters for the r function are $u_{i_{max}} = -u_{i_{min}} = 10$ for $i = 1, 2, 3$. The joint limits are chosen so that $x = T^{-1}(\eta = \eta^{\text{ref}}, \zeta_1 = 0, \zeta_2 = 0)$ is the average value of the joint limits. This is done to see if $\zeta = (0, 0)$ will become an equilibrium point for the zero dynamics.

The resulting phase curves for the zero dynamics for the 3-DOF simple robot case with viscous friction can be found in Figure 4.8.

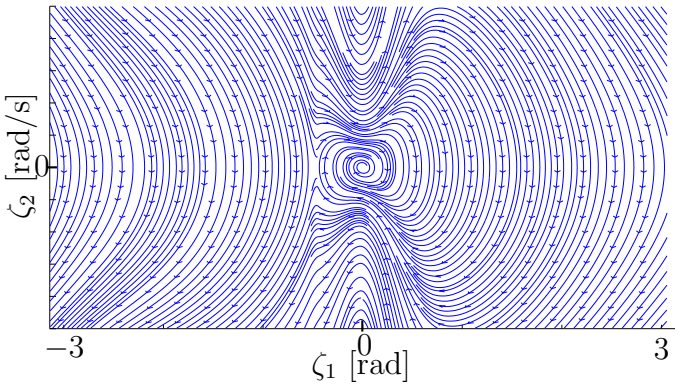

Figure 4.8: Phase portrait of the zero dynamics for the 3-DOF manipulator example with $G = 0$.

Notice there are two equilibrium points in the zero dynamics phase plane: One is unstable and one that is asymptotically stable. The unstable equilibrium point is the singularity configuration of the manipulator on the zero dynamics manifold. The asymptotically stable equilibrium point is located at $\zeta = (0, 0)$, which corresponds to the joint positions being at the midpoint of their joint limits. Furthermore, if the system were to have no viscous friction, $\zeta = (0, 0)$ will no longer be asymptotically stable. △

Remark 4.2.6. The proposed approach assumes that the system is fully actuated. To examine the under-actuated system case, let m be the number of inputs, P be the dimension of the output space, and N be the degrees of freedom of the system. There is a basic requirement ([48]) that $m \geq P - 1$ in order to only follow the path, and $m \geq P$ in order to follow the path and move along it in a desired fashion.

If $N > m = P$, it is still possible to follow the path in a desired fashion, however the internal dynamics would be completey uncontrollable, since all the freedom over the control input will be used to stay on the path and traverse the path. If $N = m > P$, then Conjecture 4.2.5 applies. If $N > m > P$, then there will be some uncontrolled

internal dynamics and the proposed redundancy resolution approach may not work to ensure boundedness of these internal dynamics. •

Remark 4.2.7. In the case that the system hits a singularity ($J(x)$ loses rank), the controller will fail as the decoupling matrix β loses rank, and there will no longer be a solution to the feedback transformation (4.16) or (4.19). The proposed methodology assumes that J is non-singular by the class of systems (2.4). Practically, one just has to ensure there exists non-singular configurations to follow the path, and the redundancy resolution scheme can be used to stay away from these configurations. •

4.2.4 Continuity of control input

The tangential and transversal dynamics using the linearizing control input v become a series of double integrators. Stabilizing such dynamics can be done using linear control techniques. For the path $\mathcal{P}$ it would be nice for the control input u to remain continuous as the output moves from one curve to another within $\mathcal{P}$.

Theorem 4.2.8. *If the auxiliary input v is chosen to be a continuous function of $\bar{T}_k(x)$ with $k = k^*$ selected by Algorithm 1, then the control input u is continuous for the entire splined path $\mathcal{P}$.*

Proof. If v is a continuous function of η and ξ, then v is continuous if and only if the η and ξ states are continuous. This is in fact the case by Proposition 4.2.3, thus v is continuous over the entire path.

The control input u is calculated based on the feedback transform (4.16) for the k'th spline:

$$v = \beta_{k^*}(x)u + \alpha_{k^*}(x)$$

Since v is continuous, u is continuous if and only if $\beta_{k^*}(x)$ and $\alpha_{k^*}(x)$ are continuous (assuming the optimization to solve for u in the redundant case is also smooth). Equivalently, in a similar manner to Proposition 4.2.3, we must show that

- $\beta_k(x)$ and $\alpha_k(x)$ are continuous given k;
- $\beta_k(x)$ and $\alpha_k(x)$ are continuous at the transistion of k:

$$(\forall x \in \{x \in \mathbb{R}^n | x = (\varpi_k \circ h(x))^{-1}(\lambda_{(k,max)})\})\ \beta_k(x) = \beta_{k+1}(x)\ ,\ \alpha_k(x) = \alpha_{k+1}(x)$$

Using the definition for $\beta_k(x)$ from Section 4.2.2, the term $J(x_c)g_v(x_c)$ is continuous over the path (by system assumptions) and is not dependent on the spline k. The remaining terms are all used in $\bar{T}_k(x)$, and thus it follows from Proposition 4.2.3 that $\beta_k(x)$ is continuous.

Similarly, using the definition for $\alpha_k(x)$ from Section 4.2.2, the terms $\mathbf{e}''_{j_k}(\lambda^*)$, $j \in \{1, \ldots, P\}$ are continuous for a given spline k because by using the FS formulae from Proposition 4.2.3, the highest derivative that appears will be $\sigma_k^{(p+1)}(\lambda^*)$; our path is C^{p+1} by Assumption 4.1.1. The remaining terms in α are continuous since they are the same ones in $\bar{T}_k(x)$, which were already shown to be continuous in Proposition 4.2.3.

□

4.3 Experiment

4.3.1 System

The experimental platform for this Chapter is the CPM (2.7). The model falls into the class of systems (2.2)-(2.4) with $f_v(x) = -M^{-1}(x_c)\left(C(x)x_v + G(x_c)\right)$, $g_v(x_c) = M^{-1}(x_c)A(x_c)$, with $N = 4$. The dimension of the output space is $P = 3$ – y_1 and y_2 are the end-effector positions projected on the ground plane, and y_3 is the end-effector's height off the ground. That is,

$$y = h(q) = \begin{bmatrix} \cos(q_1)\left(a_2\cos(q_2) + a_3\cos(q_2+q_3) + a_4(\cos(q_2+q_3+q_4)\right) \\ \sin(q_1)\left(a_2\cos(q_2) + a_3\cos(q_2+q_3) + a_4(\cos(q_2+q_3+q_4)\right) \\ a_2\sin(q_2) + a_3\sin(q_2+x_3) + d_1 \end{bmatrix}$$

Since $N > P$, the CPM system is redundant.

4.3.2 Path Generation

Waypoints in the 3-dimensional workspace were specified *a-priori*, and a standard spline-fitting process using polynomials [35] was used to fit splines through these way points while ensuring fourth-derivative continuity. Thus, quintic polynomial splines are required to fit the waypoints.

A total of 15 way points are specified. The way points were placed to create self-intersections in the path to show the robustness of Algorithm 1. Mathematically, our path

is regular and C^4, satisfies Assumptions 4.1.1 and 4.1.2, and is given as the following after spline interpolation

$$\sigma_k : \mathbb{I}_k \to \mathbb{R}^3, \qquad k = 1, 2, ..., 14$$
$$\lambda \mapsto \begin{bmatrix} \sum_{i=0}^{5} a_i \lambda^i \\ \sum_{i=0}^{5} b_i \lambda^i \\ \sum_{i=0}^{5} c_i \lambda^i \end{bmatrix},$$

where $\mathbb{I}_k = [0, l_k]$, and l_k is the chord length between way point k and $k+1$.

4.3.3 Control Design

The first step in the control design is to define the coordinate transformation. The k^* and λ^* are determined by Algorithm 1. The initial guess for Algorithm 1 is determined a-priori by solving the optimization globally using brute force and given the (known) initial output position $y_0 = h(x_0)$. Then, when the real time control law is run, Algorithm 1 is used to determine the k^* and λ^* at each time step. The coordinate transformation follows from Section 4.2.1. Since $n - 2p = 2$, we need a map $\varphi_k : \mathbb{R}^n \to \mathbb{R}^2$ to complete the diffeomorphism. We let $\varphi_{1_k}(x) = x_{c_2} + x_{c_3} + x_{c_4}$ and $\varphi_{2_k}(x) = x_{v_2} + x_{v_3} + x_{v_4}$ for all k to represent the angle and its rate, respectively, of the end-effector with respect to the ground plane. It can be shown that this completes the diffeomorphism.

The dynamics in the transformed space are then

$$\begin{aligned}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= L_f^2 \eta_{1_k^*}(x) + L_g L_f \eta_{1_k^*}(x) u \\
\dot{\xi}_1^1 &= \xi_2^1 \\
\dot{\xi}_2^1 &= L_f^2 \xi_{1_k^*}^1(x) + L_g L_f \xi_{1_k^*}^1(x) u \\
\dot{\xi}_1^2 &= \xi_{1_k^*}^2 \\
\dot{\xi}_2^2 &= L_f^2 \xi_{1_k^*}^2(x) + L_g L_f \xi_{1_k^*}^2(x) u \\
\dot{\zeta}_1 &= \zeta_2 \\
\dot{\zeta}_2 &= \sum_{i=2}^{4} [f_v(x) + g_v(x) u]_i
\end{aligned}$$

where $x = T_{k^*}^{-1}(\eta, \xi, \zeta)$, k^* is chosen by Algorithm 1, and $\dot{\zeta}$ represent the redundant dynamics for path following.

Setting $v = \beta_{k^*}(x)u + \alpha_{k^*}(x)$ as previously defined in (4.16), the following partially linearised system is obtained:

$$\begin{aligned} \dot{\eta_1} &= \eta_2 & \dot{\xi_1^1} &= \xi_2^1 & \dot{\xi_1^2} &= \xi_2^2 \\ \dot{\eta_2} &= v_\eta & \dot{\xi_2^1} &= v_{\xi^1} & \dot{\xi_2^2} &= v_{\xi^2} \\ \dot{\zeta} &= \bar{f}_\zeta(x, v) \end{aligned}$$

where $x = T_{k^*}^{-1}(\eta, \xi, \zeta)$ and $\bar{f}_\zeta$ represents the redundant dynamics based on our choice of control input u. The desired task is to track a constant tangential velocity profile (i.e., to track some η_2^{ref}) while staying on the path (force ξ to go to zero). Designing a controller for v is a linear control design problem; the one designed in Chapter 3 is used as it was shown to be robust to the modelling inaccuracies. The tangential controller is the PI controller (3.17), and the transversal controller is (3.20).

To solve for u based on the auxiliary control v, the optimization is done as outlined in Section 4.2.3 using (4.18). The actuation limits are $u_{i_{max}} = -u_{i_{min}} = 50\%$ duty cycle for $i = 1, \ldots, 4$. The joint limit values $x_{c_{i_{max}}}, x_{c_{i_{min}}}$ were set to their true limits for joints $i = 1, 2, 3$, and for the wrist joint ($i = 4$) two different runs were done, one with limits set to $0 - 45$ degrees and another to $45 - 90$ degrees. The results are found in the subsequent section.

We implement the path following controller on the CPM via Labview Real-Time Module© to control the motor PWM amplifiers and to read the optical encoders. The loop rate of the controller is 20 ms. The encoders read the linear actuator distances, so an inverse measurement model is used to retrieve the states x_c of the system. The encoder resolution for the waist is $1.3E-5$ rad, and for the linear actuators is $4.2E-5$ inch. The derivative of x_c is numerically computed to retrieve the x_v states using a first-order approximation. The tangential controller (3.17) is the only dynamic part of the control algorithm, and is discretized also using a first-order approximation.

4.3.4 Results

The results of the path following controllers using both joint limit values can be found in Figures 4.9-4.13. The output of Algorithm 1 is also shown for reference, with the number of iterations the gradient descent takes to converge at each time step shown.

The path following controller naturally converges to the closest point on the path due to the explicit stabilization of the path following manifold.

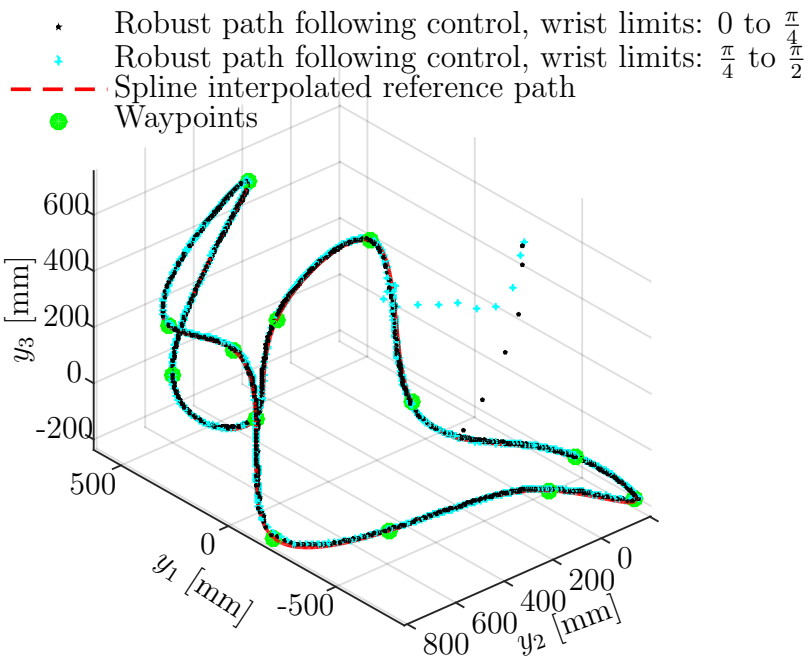

Figure 4.9: Experiment: Output position.

Note that in the Cartesian plot, the responses are very similar, with similar path errors. The transformed state plots also show this as expected: the controllable η and ξ subsystems behave the same in both scenarios, because these subsystems, under the feedback transform, do not depend on the redundant state dynamics (ζ). The redundant state ζ trajectory differs, however, since we are using the redundancy to maintain different joint angle limits using (4.17),(4.18). The plots show well-behaved dynamics and boundedness for ζ as we had postulated for systems with inherent damping.

Closer inspection of the x_c states in Figure 4.12 shows that indeed our controller was able to satisfy the respective joint angle limits. The angle trajectories are also slightly different for the first three joints, because maintaining the output on the path with a different preferred position of the wrist forces the angles for the elbow and shoulder links to adjust accordingly. All this is done automatically by (4.17),(4.18).

Figure 4.13 shows well behaved control action without chattering.

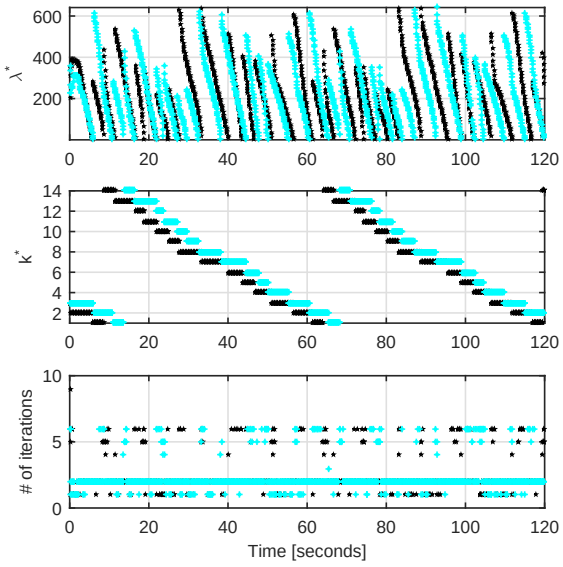

Figure 4.10: Experiment: Algorithm 1 information.

4.4 Summary

This chapter proposes a method for following general paths in the form of sequences of curves; notably spline interpolated paths which can be made to go through arbitrary way points. The path following manifold corresponding to this path is stabilized, thus rendering the path attractive and invariant. A numerical algorithm is used to allow the use of paths which are self-intersecting. This in combination with the robust controller used in Chapter 3 fully achieve **PF1** to **PF3**.

Furthermore, a redundancy resolution scheme is proposed based on a static optimization. A conjecture that this scheme can yield bounded zero-dynamics is made and, in particular, satisfy an objective of staying away from joint limits. This scheme has been validated in two analytical examples, as well as on our experimental platform. A rigorous proof may be direction for future work. This achieves **PF4** for redundant Euler-Lagrange systems. In the next chapter, we will apply the ideas of the redundancy resolution developed for use on a more general, mobile manipulator system.

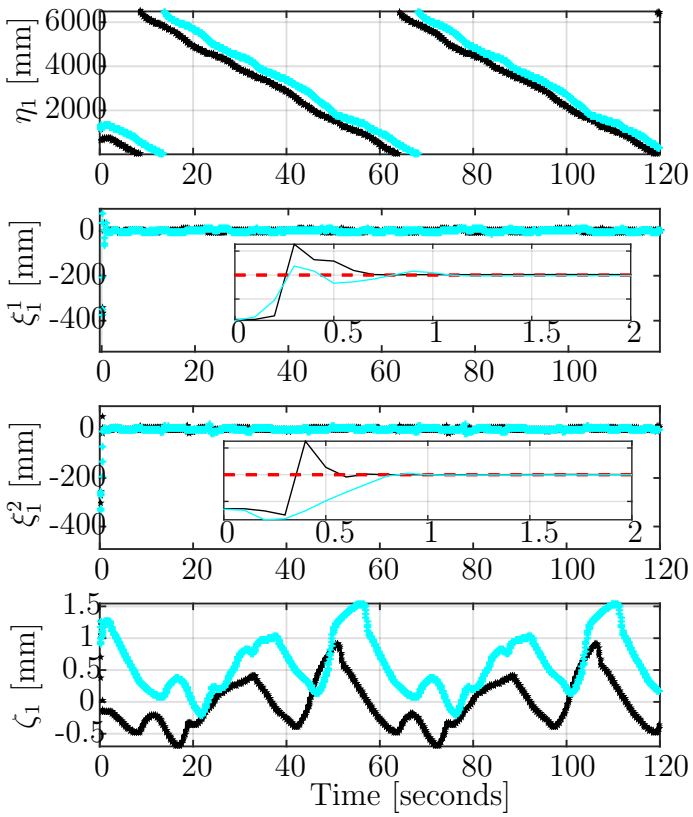

Figure 4.11: Experiment: Transformed state positions.

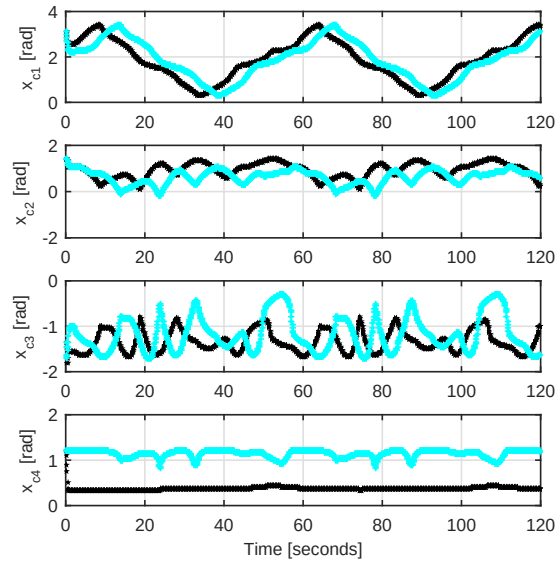

Figure 4.12: Experiment: Joint angles.

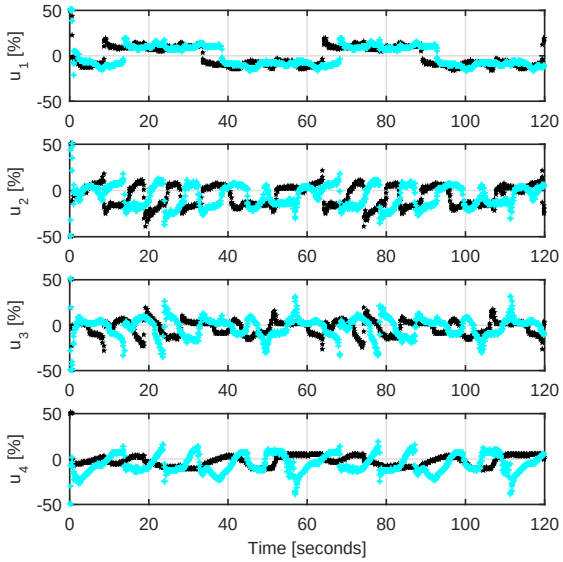

Figure 4.13: Experiment: Control effort.

Chapter 5

Mobile Manipulator Path Following

This chapter presents a framework of path following via set stabilization for mobile manipulator systems. The mobile manipulator is modelled as a single redundant dynamic system. The mobile base considered belongs to a large class of wheeled ground vehicles, including those with nonholonomic constraints, see Chapter 2. The path following framework stems from Chapter 3 in terms of robust control, and Chapter 4 in terms of path following of splines and redundancy resolution. The controller simultaneously controls the manipulator and mobile base. The result is a unified path following controller without any trajectory planning performed on the mobile base. The approach is experimentally verified on a 4-degree-of-freedom (4-DOF) manipulator mounted on a differential drive mobile platform in Section 5.3.

5.1 Problem Formulation

The problem studied in this chapter is to find a continuous feedback control law for u_m, γ_b, γ_s for the mobile manipulator system (2.2)(2.8) with output (2.15), to achieve **PF1** to **PF4**. This will be done via set stabilization [76] as has been done in the previous chapters. Since the mobile manipulator system is redundant, the redundancy is used to avoid (physical or artificial) manipulator and steering wheel limits.

The class of paths considered are the same as in Chapter 4, instead now the output space is of dimension p.

5.2 Path Following Control Design

The control design approach will rely on the output y having a well-defined relative degree of $\{2, ..., 2\}$. That is, the control inputs u_m, γ_b, γ_s all appear in the second time-derivative of y. It can be shown that this is not the case: the control inputs γ_b appear too soon by one derivative. Hence, by delaying the appearance of this input via dynamic extension, we can make y have a well-defined relative degree for the augmented system (Section 5.2.1). Then a virtual output (Section 5.2.2) is constructed that incorporates the desired path (4.1). Through partial feedback linearization, the virtual output partitions the system dynamics into a controllable transversal and tangential sub-systems that govern the motion transversal to the path and along the path, respectively, and a redundant sub-system that governs the motion of the system that does not produce any output motion (Section 5.2.3). A novel redundancy resolution approach is then used to ensure the redundant dynamics remain bounded (Section 5.2.4), while maintaining joint limits of the manipulator. The overall block diagram can be found below in Figure 5.1.

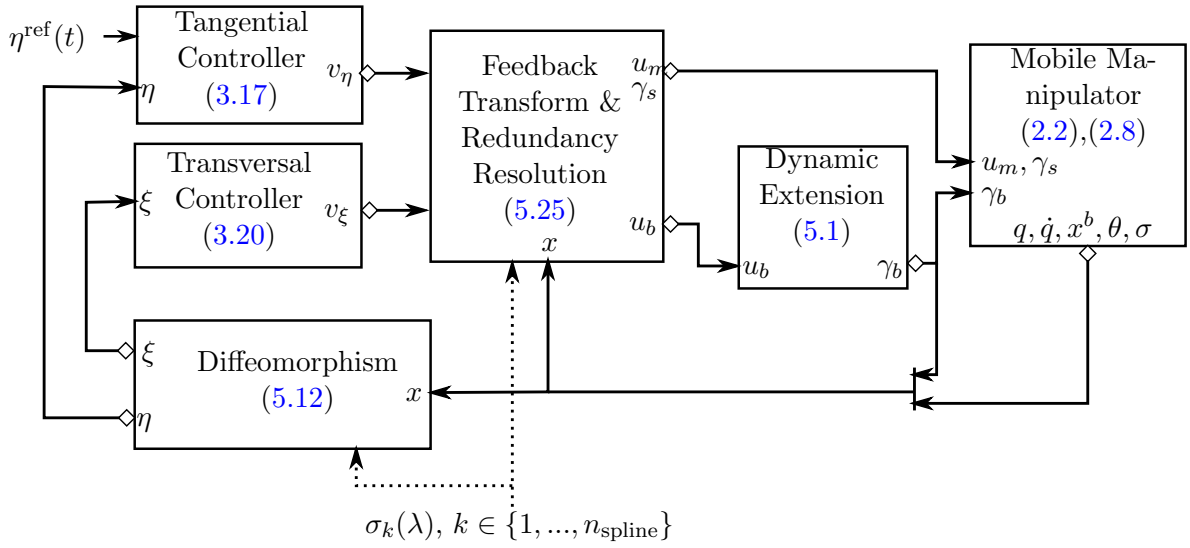

Figure 5.1: Spline path following controller block diagram for mobile manipulator systems.

5.2.1 Dynamic Extension

Dynamic extension amounts to controlling the derivative of the actual control input. Since γ_b appears too soon by one derivative when computing the derivatives of y, we introduce an auxiliary control variable $u_b \in \mathbb{R}^{\delta_b}$ as follows

$$\dot{\gamma}_b = \rho\gamma_b + u_b \tag{5.1}$$

where $\rho < 0$ is a damping coefficient. It is typical with dynamic extension [3] to let $\rho = 0$, however we will see in Section 5.2.4 that setting $\rho = 0$ results in unbounded internal dynamics.

State Space Equations

To facilitate the control design procedure, we write the dynamics (2.2),(2.8),(5.1) with output (2.15) in state-space form. First, let $x_c \coloneqq (q, x^b, \theta) \in \mathbb{R}^N \times \mathbb{R}^2 \times \mathbb{R}$, $x_v \coloneqq (\dot{q}, \gamma_b, \sigma) \in \mathbb{R}^N \times \mathbb{R}^{\delta_b} \times \mathbb{R}^{\delta_s}$. Then,

$$\dot{x}_c = \begin{bmatrix} \dot{q} \\ R(\theta)\Sigma(\sigma)\gamma_b \end{bmatrix} =: F_c(x) \tag{5.2}$$

$$\dot{x}_v = \begin{bmatrix} -M^{-1}(q)\left(C(q,\dot{q})\dot{q} + G(q)\right) \\ \rho\gamma_b \\ 0_{\delta_s \times 1} \end{bmatrix} + \begin{bmatrix} M^{-1}(q)A(q) & 0_{N\times\delta_b} & 0_{N\times\delta_s} \\ 0_{\delta_b\times N} & I_{\delta_b\times\delta_b} & 0_{\delta_s\times\delta_s} \\ 0_{\delta_b\times N} & 0_{\delta_b\times\delta_b} & I_{\delta_s\times\delta_s} \end{bmatrix} \begin{bmatrix} u_m \\ u_b \\ \gamma_s \end{bmatrix} \tag{5.3}$$

$$=: F_v(x) + G_v(x)u \tag{5.4}$$

$$y = \begin{bmatrix} R(\theta) & 0_{3\times(p-3)} \\ 0_{(p-3)\times 3} & I_{(p-3)\times(p-3)} \end{bmatrix} h(q) + \begin{bmatrix} x_1^b \\ x_2^b \\ 0_{(p-2)\times 1} \end{bmatrix} =: H(x_c) \tag{5.5}$$

where $x \coloneqq (x_c, x_v) \in \mathbb{R}^{2N+3+\delta_b+\delta_s}$ is the state of the system, $u \coloneqq (u_m, u_b, \gamma_s) \in \mathbb{R}^{N+\delta_b+\delta_s}$ is the control input to the system, and y is the output. Note that $F_c(x)$ and $F_v(x)$ are smooth vector fields, $G_v(x)$ is smooth and has full rank, and $\frac{\partial H(x_c)}{\partial x_v} = 0$. Note that the actual inputs to control the physical systems (2.2),(2.8) are u_m, γ_b, γ_s.

The state space equations can be compactly written as

$$\dot{x} = \begin{bmatrix} \dot{x}_c \\ \dot{x}_v \end{bmatrix} = \begin{bmatrix} F_c(x) \\ F_v(x) \end{bmatrix} + \begin{bmatrix} 0_{(N+3)\times(N+\delta_b+\delta_s)} \\ G_v(x) \end{bmatrix} u =: f(x) + g(x)u \tag{5.6}$$

$$y = H(x_c) \tag{5.7}$$

The mobile manipulator system is redundant in the sense that $\dim u = N + \delta_b + \delta_s \geq \dim y = p$.

5.2.2 Virtual Output

As we have seen in previous chapters, the virtual output construction is two-fold: the first virtual output is the projected position of the output y along the desired path (Section 5.2.2). The second virtual output are the transversal positions towards the path along each FSF normal vector (Section 5.2.2). Essentially, the results of Chapter 4 apply here but for the system (5.6),(5.7).

Tangential State

The first virtual output, $\eta_1(x)$, is the tangential position of the output y along the desired path. Assume that the closest curve to the output y is known; denote its index as $k^* \in \{1, \ldots, n_{\text{spline}}\}$. Furthermore, denote the parameter of spline k^* that corresponds to the closest point to the output y as $\lambda^* \in \mathbb{I}_{k^*}$

$$\lambda^* \coloneqq \varpi_{k^*}(y) \coloneqq \underset{\lambda \in \mathbb{I}_{k^*}}{\arg\inf} \left\| y - \sigma_{k^*}(\lambda) \right\|. \tag{5.8}$$

The minimization for λ^* and the determination of k^* are done numerically using Algorithm 1 from Chapter 4. Henceforth, the k^* notation will be dropped for brevity.

The first virtual output is the projected, traversed arclength along the spline:

$$\eta_1 = \eta_1(x) \coloneqq s_{k^*} \circ \varpi \circ H(x_c) + \sum_{k=1}^{k^*-1} s_k(\lambda_{(k,max)}) \tag{5.9}$$

where

$$s_k(\lambda) \coloneqq \int_{\lambda_{(k,min)}}^{\lambda} \left\| \frac{\mathrm{d}\sigma_k(\lambda)}{\mathrm{d}\lambda} \right\| \mathrm{d}\lambda.$$

Transversal States

The remaining virtual outputs are selected to represent the cross track error to the path using the remaining FSF normal vectors, known as transversal positions (see Chapter 4).

The transversal positions can be computed by projecting the error to the path, $y - \sigma(\lambda^*)$, onto each of the FS normal vectors:

$$\xi_1^{j-1} = \xi_1^{j-1}(x) \coloneqq \langle \mathbf{e}_j(\lambda^*), H(x_c) - \sigma_{k^*}(\lambda^*) \rangle|_{\lambda^* = \varpi \circ H(x_c)} \tag{5.10}$$

for $j \in \{2, \dots, p\}$.

The tangential and transversal positions are illustrated in Figure 5.2 in the case that $p = 3$.

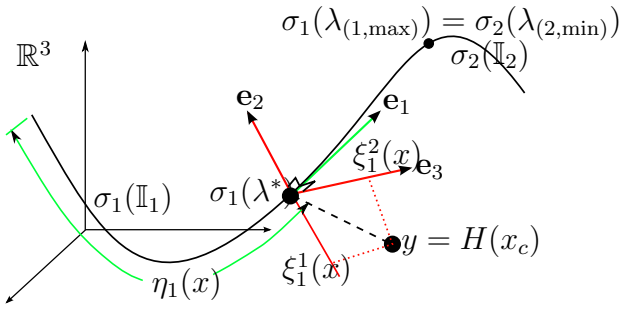

Figure 5.2: The tangential and transversal state positions when $p = 3$.

5.2.3 Dynamics and Control

It will be shown that the virtual output $(\eta_1(x), \xi_1^1(x), \dots, \xi_1^{p-1}(x)) \in \mathbb{R}^p$ has a well-defined vector relative degree of $\{2, 2, \dots, 2\}$ for all $x \in \mathcal{U}$, where

$$\mathcal{U} \coloneqq \left\{ x \in \mathbb{R}^{2N+3+\delta_b+\delta_s} \mid \operatorname{rank}\left(\frac{\partial H}{\partial x_c} \frac{\partial F_c}{\partial x_v} G_v(x) \right) = p \right\} \tag{5.11}$$

(see (5.17)) when (5.8) is solved for numerically by Algorithm 1. We emphasize that (5.8) is solved for numerically by a local search since if the output y is equidistant to multiple points on the path, then (5.8) is not well defined and neither is the relative degree.

Therefore, there exists a local coordinate transformation

$$T : \mathcal{U} \to \mathbb{R}^{2N+3+\delta_b+\delta_s} \tag{5.12}$$

$$x \mapsto (\eta_1(x), L_f\eta_1(x), \xi_1^1(x), L_f\xi_1^1(x), \dots, \xi_1^{p-1}(x), L_f\xi_1^{p-1}(x), \varphi(x)) \tag{5.13}$$

$$= (\eta_1, \eta_2, \xi_1^1, \xi_2^1, \dots, \xi_1^{p-1}, \xi_2^{p-1}, \zeta), \tag{5.14}$$

where $L_f\eta_1(x) = \frac{\partial \eta_1(x)}{\partial x_c} F_c(x)$ and $L_f\xi_1^j(x) = \frac{\partial \xi_1^j(x)}{\partial x_c} F_c(x), j \in \{1, \dots, p-1\}$, is a diffeomorphism onto it's image for a suitable function $\varphi : \mathbb{R}^{2N+3+\delta_b+\delta_s} \to \mathbb{R}^{2N+3+\delta_b+\delta_s-2p}$ [52] (which may limit the domain of T by [52, Proposition 5.1.2], but in practice (see Section 5.3.2),

these functions do not need to be computed unless one wants to visualize the redundant dynamics or study their stability properties). Note that since the virtual output η_1 represents the tangential position of the output y along the path, η_2 represents the tangential velocity of the output along the path. Similarly, since ξ_1^j represents the transversal positions to the path, ξ_2^j represents the transversal velocities, $j \in \{1, \dots, p-1\}$. Note that there will always be an internal state ζ, since by Assumption 2.2.1, $N + \delta_b + \delta_s \geq N + \delta_m \geq p$ and $\delta_b + \delta_s \in \{2, 3\}$ always implies $2N + 3 + \delta_b + \delta_s - 2p > 0$, which is expected as mobile manipulators are highly redundant systems.

The dynamics in the transformed coordinates are

$$\begin{aligned}
\dot{\eta}_1 &= L_f \eta_1(x) = \eta_2 \\
\dot{\eta}_2 &= L_f^2 \eta_1(x) + L_g L_f \eta_1(x) u \\
\dot{\xi}_1^1 &= L_f \xi_1^1(x) = \xi_2^1 \\
\dot{\xi}_2^1 &= L_f^2 \xi_1^1(x) + L_g L_f \xi_1^1(x) u \\
&\vdots \\
\dot{\xi}_1^{p-1} &= L_f \xi_1^{p-1}(x) = \xi_2^{p-1} \\
\dot{\xi}_2^{p-1} &= L_f^2 \xi_1^{p-1}(x) + L_g L_f \xi_1^{p-1}(x) u \\
\dot{\zeta} &= L_f \varphi(x) + L_g \varphi(x) u =: f_\zeta(x, u)
\end{aligned} \tag{5.15}$$

where $x = T^{-1}(\eta, \xi, \zeta)$. The expressions for the Lie derivatives can be found below and are computed in a similar manner as in Chapter 4, applied to the system (5.6):

$$\eta_2 = L_f \eta_1(x) = \frac{\mathrm{d}s}{\mathrm{d}\lambda^*} \frac{\mathrm{d}\varpi}{\mathrm{d}y} \frac{\partial H}{\partial x} \dot{x} = \left\langle \mathbf{e}_1(\lambda^*), \frac{\partial H}{\partial x_c} F_c(x) \right\rangle,$$

$$\xi_2^{j-1} = L_f \xi_1^{j-1} = \frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|} \left\langle \mathbf{e}_j'(\lambda^*), H(x_c) - \sigma(\lambda^*) \right\rangle + \left\langle \mathbf{e}_{j_{k^*}}(\lambda^*), \frac{\partial H(x_c)}{\partial x_c} F_c(x) \right\rangle,$$

$$\begin{aligned}
L_f^2 \eta_1(x) = \frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|} &\left\langle \mathbf{e}_1'(\lambda^*), \frac{\partial H(x_c)}{\partial x_c} F_c(x) \right\rangle \\
&+ \left\langle \mathbf{e}_1(\lambda^*), \frac{\partial \frac{\partial H(x_c)}{\partial x_c} F_c(x)}{\partial x_c} F_c(x) + \frac{\partial H(x_c)}{\partial x_c} \frac{\partial F_c(x)}{\partial x_v} F_v(x) \right\rangle,
\end{aligned}$$

$$L_g L_f \eta_1(x) = \mathbf{e}_1(\lambda^*)^\top \frac{\partial H(x_c)}{\partial x_c} \frac{\partial F_c(x)}{\partial x_v} G_v(x_c),$$

$$L_f^2\xi_1^{j-1}(x) = \left\langle \mathbf{e}_j(\lambda^*), \frac{\partial \frac{\partial H(x_c)}{\partial x_c} F_c(x)}{\partial x_c} F_c(x) + \frac{\partial H(x_c)}{\partial x_c}\frac{\partial F_c(x)}{\partial x_v} F_v(x) \right\rangle + \frac{\eta_2(x)}{\|\sigma'(\lambda^*)\|}\left\langle \mathbf{e}_j'(\lambda^*), 2J\frac{\partial H(x_c)}{\partial x_c}F_c(x) - \mathbf{e}_1\eta_2(x)\right\rangle + \left\langle H(x_c) - \sigma(\lambda^*), \mathbf{e}_j''(\lambda^*)\left(\frac{\eta_2(x)}{\|\sigma'(\lambda^*\|}\right)^2 + \mathbf{e}_j'(\lambda^*)\left(\frac{L_gL_f\eta_1(x)}{\|\sigma'(\lambda^*)\|} - \eta_2(x)^2\frac{\langle \sigma'(\lambda^*),\ \sigma''(\lambda^*)\rangle}{\|\sigma'(\lambda^*)\|^4}\right)\right\rangle,$$

$$L_gL_f\xi_1^{j-1}(x) \;=\; \mathbf{e}_j(\lambda^*)^\top \frac{\partial H(x_c)}{\partial x_c}\frac{\partial F_c(x)}{\partial x_v}G_v(x) \;+\; (H(x_c) \;-\; \sigma(\lambda^*))^\top \mathbf{e}_j'(\lambda^*)\frac{L_gL_f\eta_1(x)}{\|\sigma'(\lambda^*)\|}$$

for $j \in \{2, \ldots, p\}$ and $\lambda^* = \varpi \circ H(x_c)$.

Based on the virtual output construction (Section 5.2.2), the η-subsystem represents the dynamics tangent to the path, and the ξ-subsystem represents the dynamics transversal to the path. Thus, the dynamics f_ζ represent the dynamics of the system (5.6) that do not produce any output motion. These are the redundant dynamics to path following.

Next we perform standard partial feedback linearization for non-square systems on the virtual output $(\eta_1(x), \xi_1^1(x), \ldots, \xi_1^{p-1}(x))$. Let

$$\alpha(x) \coloneqq \begin{bmatrix} L_f^2\eta_1(x) & L_f^2\xi_1^1(x) & \ldots & L_f^2\xi_1^{p-1}(x)\end{bmatrix}^\intercal \in \mathbb{R}^p \tag{5.16}$$

$$\beta(x) \coloneqq \begin{bmatrix} L_gL_f\eta_1(x) \\ L_gL_f\xi_1^1(x) \\ \vdots \\ L_gL_f\xi_1^{p-1}(x)\end{bmatrix} \in \mathbb{R}^{p\times(N+\delta_b+\delta_s)}. \tag{5.17}$$

The decoupling matrix $\beta(x)$ has full row rank p in the set $\mathcal{U}$ (see (5.11)) since each $\mathbf{e}_i$ are orthogonal by FSF construction, thus in U the virtual output has a well-defined relative degree [52].

Remark 5.2.1. A necessary condition for $\beta(x)$ to have rank p is that each matrix $\frac{\partial H}{\partial x_c}$, $\frac{\partial F_c}{\partial x_v}$ and $G_v(x)$ have rank of at least p. The matrix $\frac{\partial H(x_c)}{\partial x_c}$ has the form

$$\frac{\partial H(x_c)}{\partial x_c} = \left[\begin{array}{c:c:c} \begin{bmatrix} R(\theta)_{1:2,1:2} & 0_{2\times(P-2)} \\ 0_{(p-2)\times 2} & I_{(p-2)\times(P-2)}\end{bmatrix} J(q) & \begin{matrix} I_{2\times 2} \\ 0_{(p-2)\times 2}\end{matrix} & \begin{bmatrix} R'(\theta)_{1:2,1:2} & 0_{2\times(P-2)} \\ 0_{(p-2)\times 2} & 0_{(p-2)\times(P-2)}\end{bmatrix} h(q)\end{array}\right] \tag{5.18}$$

and always has rank of at least 2 due to the ground vehicle (thus is always full rank if $p = 2$). For $p > 2$, it has full rank if $J(q)$ is non-singular.

The matrix $\frac{\partial F_c(x)}{\partial x_v}$ has the form

$$\frac{\partial F_c}{\partial x_v} = \begin{bmatrix} I_{N\times N} & 0_{N\times\delta_b} & 0_{N\times\delta_s} \\ 0_{3\times N} & R(\theta)\Sigma(\sigma) & R(\theta)\frac{\partial\Sigma(\sigma)}{\partial\sigma} \end{bmatrix}, \tag{5.19}$$

thus $N + 3 \geq \text{rank}(\frac{\partial F_c}{\partial x_v}) \geq N + \delta_b \geq p$ (by Assumption 2.2.1) and will depend on the particular ground vehicle. For a unicycle (Section 5.3.1), for all x, $\text{rank}(\frac{\partial F_c(x)}{\partial x_v}) = N + \delta_b = N + 2$.

The matrix $G_v(x)$ (see Equation (5.2)) always has full rank of $N + \delta_b + \delta_s \geq p$. •

Next, introduce an auxiliary input $v := (v_\eta, v_\xi) \in \mathbb{R} \times \mathbb{R}^{p-1}$ such that

$$v = \beta(x)u + \alpha(x). \tag{5.20}$$

The dynamics (5.15) then become

$$\dot{\zeta} = f_\zeta(x, u) \qquad \begin{matrix} \dot{\eta}_1 = \eta_2 \\ \dot{\eta}_2 = v_\eta \end{matrix} \qquad \begin{matrix} \dot{\xi}_1^j = \xi_2^j \\ \dot{\xi}_2^j = v_{\xi^j} \end{matrix}$$

where $j \in \{1, \ldots, p-1\}$ and $x = T^{-1}(\eta, \xi, \zeta)$.

In the set $\mathcal{U}$ the ξ-subsystem is linear and controllable, and can be stabilized to ensure attractiveness and invariance of the path $\mathcal{P}$ [48], thereby satisfying **PF1** to **PF2** using the robust controller (3.20). The η-subsystem is also linear and controllable, thus a tangential controller can be designed for v_η to track a desired tangential position or velocity profile $\eta^{\text{ref}}(t)$. On the experimental platform, this is done via a PI controller for robustness with feed-forward (3.17), thereby satisfying **PF3**. Figure 5.1 shows a block diagram of the entire control system.

5.2.4 Redundancy Resolution

Once the auxiliary control signal $v \in \mathbb{R}^p$ is generated to stabilize the η and ξ-subsystems for **PF1** to **PF3**, the feedback transformation (5.20) must be solved to generate the actual control signal $u \in \mathbb{R}^{N+\delta_b+\delta_s}$. When $N + \delta_b + \delta_s = p$, there is a unique solution to (5.20), and one must ensure that the internal dynamics $\dot{\zeta} = f_\zeta(x, u)$ remain bounded.

For most mobile manipulator systems (see Section 5.3 for an example), $N+\delta_b+\delta_s > p$. Thus there is some freedom in the choice of the control input u under the feedback transform (5.20). This freedom can be used to enforce boundedness of the internal dynamics. The approach used here uses the ideas of Chapter 4 but applied to mobile manipulator systems. Consider the static optimization

$$\begin{aligned} &\min_{u} \left(u - r(x)\right)^\top W \left(u - r(x)\right) \\ &\text{s.t.} \quad v = \beta(x)u + \alpha(x) \end{aligned} \tag{5.21}$$

The matrix $W \in \mathbb{R}^{(N+\delta_b+\delta_s)\times(N+\delta_b+\delta_s)}$ is a positive-definite weighting matrix, and the function $r : \mathbb{R}^{N+3} \to \mathbb{R}^{N+\delta_b+\delta_s}$ is used to bias the control input u to achieve desired behaviour in the internal dynamics.

In this application, we want to bias $u = (u_m, \gamma_b, \gamma_s)$ such that the manipulator stays away from joint and actuation limits and the vehicle stays away from steering limits (**PF4**). We apply a similar approach as in Chapter 4 for redundancy resolution: if a joint $q_i, i \in \{1, ..., N\}$ of the manipulator is at its minimum limit $q_i^{\min}$, setting the control effort $u_i, i \in \{1, ..., N\}$ (i.e., u_m) corresponding to this joint to be the maximum control effort $u_i^{\max}$ will *likely* increase the value of the joint, thereby pushing it away from the negative joint limit. These joint limits can be set artificially to bias the manipulator in a preferred position (so as to avoid singular configurations of $J(q)$, see Remark 5.2.1) or set to their true joint limit values. If the vehicle has steering wheels ($\delta_s > 0$), that has steering limits, then a similar logic can be applied to the steering limits $\sigma_i^{\max}, \sigma_i^{\min}$ and steering input limits $\gamma_{s_i}^{\max}, \gamma_{s_i}^{\min}$ for $i \in \{1, \ldots, \delta_s\}$. For γ_b, we'd like to minimize the control effort so that the mobile base moves as little as possible. The corresponding r function to achieve this behaviour in (5.21) is

$$r(x)_i \coloneqq -\frac{u_i^{\max} - u_i^{\min}}{q_i^{\max} - q_i^{\min}}(q_i - q_i^{\min}) + u_i^{\max}, \quad i \in \{1, ..., N\} \tag{5.22}$$

$$r(x)_i \coloneqq 0, \quad i \in \{N+1, N+\delta_b\} \tag{5.23}$$

$$r(x)_i \coloneqq -\frac{u_i^{\max} - u_i^{\min}}{\sigma_j^{\max} - \sigma_j^{\min}}(\sigma_j - \sigma_j^{\min}) + u_i^{\max}, \quad i \in \{N+\delta_b+1, ..., N+\delta_b+\delta_s\}, j = i - (N+\delta_b) \tag{5.24}$$

Using Lagrangian multipliers, the solution to (5.21) is

$$u = \beta^\dagger(x)(v - \alpha(x)) + \left(I_{n\times n} - \beta^\dagger(x)\beta(x)\right) r(x) \tag{5.25}$$

where $\beta^\dagger(x) \coloneqq W^{-1}\beta(x)^\top \left(\beta(x)W^{-1}\beta(x)^\top\right)^{-1}$.

In Chapter 4, Conjecture 4.2.5, we showed via examples and experiments on manipulators that a similar control law (5.25) ensures boundedness of the internal dynamics while maintaining joint limits of a manipulator when each degree of freedom has inherent viscous friction. In this paper, for mobile manipulator systems, boundedness of internal dynamics and maintaining joint limits of the manipulator holds when $\rho < 0$ in the dynamic extension (5.1) (**PF4**). If $\rho = 0$, **PF1** to **PF3** can still be achieved, but at least one of the base position states (x^b, θ) may become unbounded, see Ex. 5. The reason is u_b controls the mobile base acceleration through (5.1). So although (5.22) is chosen such that base acceleration is minimized, ideally keeping $u_b = 0$, then the base velocities will remain at a some constant, not necessarily zero, value, resulting in the base positions (namely θ) to continuously grow. Thus, $\rho < 0$ is used to dampen these dynamics. The same doesn't happen for the steering dynamics of σ since the steering dynamics are governed by a single integrator. So γ_s is minimized, ideally 0 when possible, implies the steering position is constant when possible, thus remaining bounded. A simulation for a ground vehicle with steering can be found in Example 6.

Example 5. (Internal Dynamics) Consider the mobile manipulator system of Section 5.3.1. In this system and in most systems, $u_i^{\max} = -u_i^{\min}$ for use in (5.22). The joint limits are set to their true values except for the waist joint q_1 where $q_1^{\min} := 0°, q_1^{\max} := 20°$. Thus the optimal control effort for the waist $r(x_c)_1 = 0$ when $q_1 = 10°$, so that the desired position of the waist is 10°, that is, the manipulator should face to the right of the vehicle (or the vehicle is to stay towards the left of the desired path). The path is shown in Figure 5.3, where two simulation results are included that show **PF1** to **PF3** are achieved, and there is automatic coordination between the manipulator and the mobile base. When $\rho = 0$, **PF4** is not achieved as there is an internal state that is unbounded. Namely, the yaw state θ of the manipulator continuously increases and the vehicle orbits the end-effector as the end-effector follows the path, see Figure 5.4-5.5.

When $\rho < 0$, **PF4** is achieved as discussed above. At the very start of the run, the base automatically orients itself by rotating, travels backwards until the end-effector is at its preferred position while the end effector is on the path $\mathcal{P}$, then proceeds forward as it achieves **PF3**. Note that the initial position of the base could have been placed further away from the path, and the same controller will automatically move the base to a neighbourhood of the path, see Figure 5.6. Further notice the controller automatically speeds up the mobile base when the end-effector is traversing the higher curvature areas of the path in order to maintain the desired constant tangential velocity profile.

△

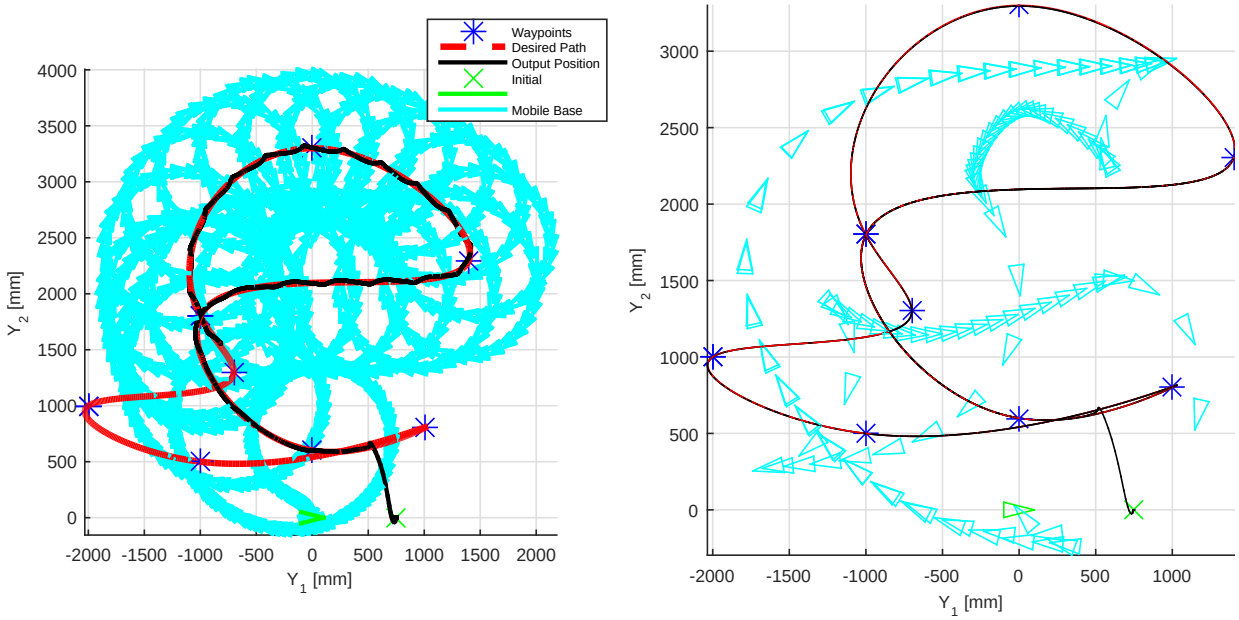

Figure 5.3: Example 5: Simulation of $\rho = 0$ (left) and $\rho = -10$ (right). $\eta_2^{\text{ref}} = 100\text{mm/ sec}$. The preferred position of q_1 is set to 10 degrees. That is, the manipulator should face to the right of the mobile base. In both figures, this is the case. However, when $\rho = 0$, the mobile base orbits the output position $y = H(x_c(t))$ such that it maintains the preferred position and is on the desired path $\mathcal{P}$. Each snapshot of the mobile base is taken every 1.5 seconds.

Example 6. (Car-like ground vehicle) Consider the same 4-DOF CPM manipulator as in Section 5.3.1, but instead of a unicycle ground vehicle we consider a car-like ground vehicle as in Example 1. The car model we consider has $l = 1\text{m}$ and $\sigma^{\max} = -\sigma^{\min} = 60^\circ$ (saturated steering can be seen in Figure 5.11). Applying the path following controller framework of Section 5.2, to the same path in Example 5, with $\rho = -10$, yields the responses in Figures 5.9 to 5.12. The vehicle takes a different trajectory towards the path compared to the unicycle robot in Example 5, due to the fixed turning radius. In Figure 5.10, there are multiple points where the car actually travels backwards while the end-effector progresses forwards along the path in order to achieve **PF3** (at points (0,1500), (750,2900), and (-1500,1000)).

△

Remark 5.2.2. The weighting matrix W plays an important role to ensure the joint limits are satisfied. If the associated weights for the manipulator joint actuation are too low, the

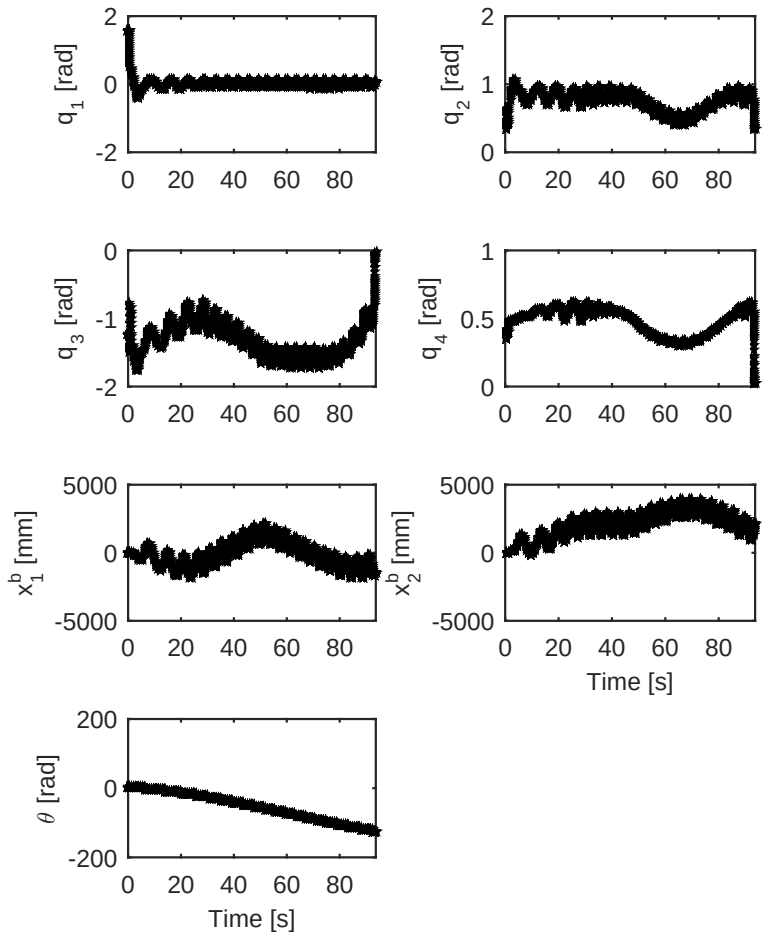

Figure 5.4: Example 5: State position $x_c = (q, x^b, \theta)$ trajectories – $\rho = 0$.

controller may not move the mobile base at all before the manipulator reaches, say, a singular configuration. If the weights are high enough, then the controller will increase control effort to the base so that the base moves while the manipulator satisfies its joint limits. •

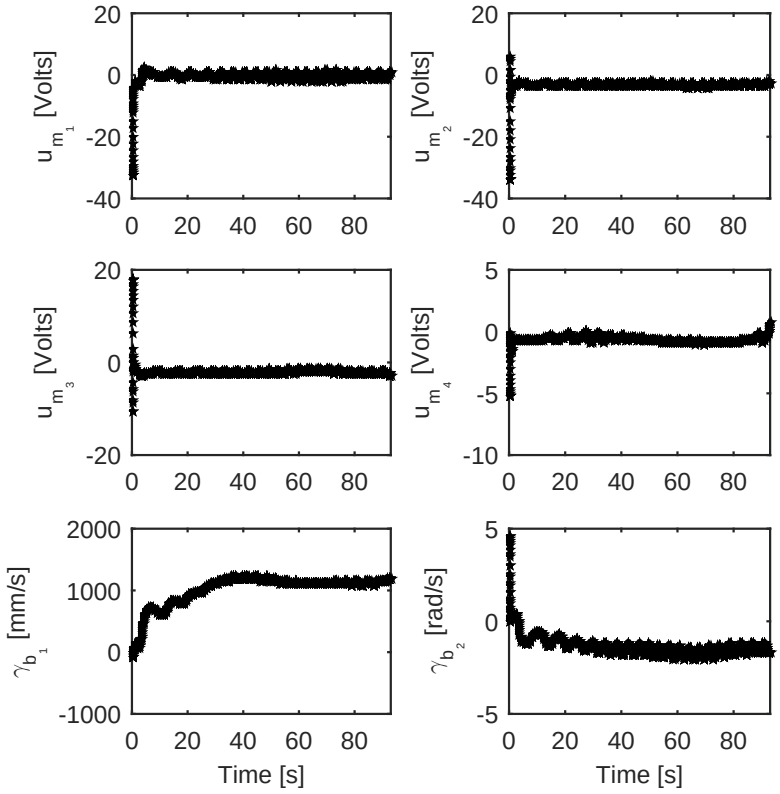

Figure 5.5: Example 5: Control inputs u_m, γ_b – $\rho = 0$.

5.3 Experiment

5.3.1 System Model

The mobile manipulator system can be found in Chapter 2, Section 2.2.3. The general mobile vehicle equations (2.8) reduce to the unicycle equations (2.16). On the Clearpath mobile platform, $\delta_s = 0$ and $\delta_b = 2$. The output space is taken to be the 3-dimensional Euclidean space, that is $p = 3$.

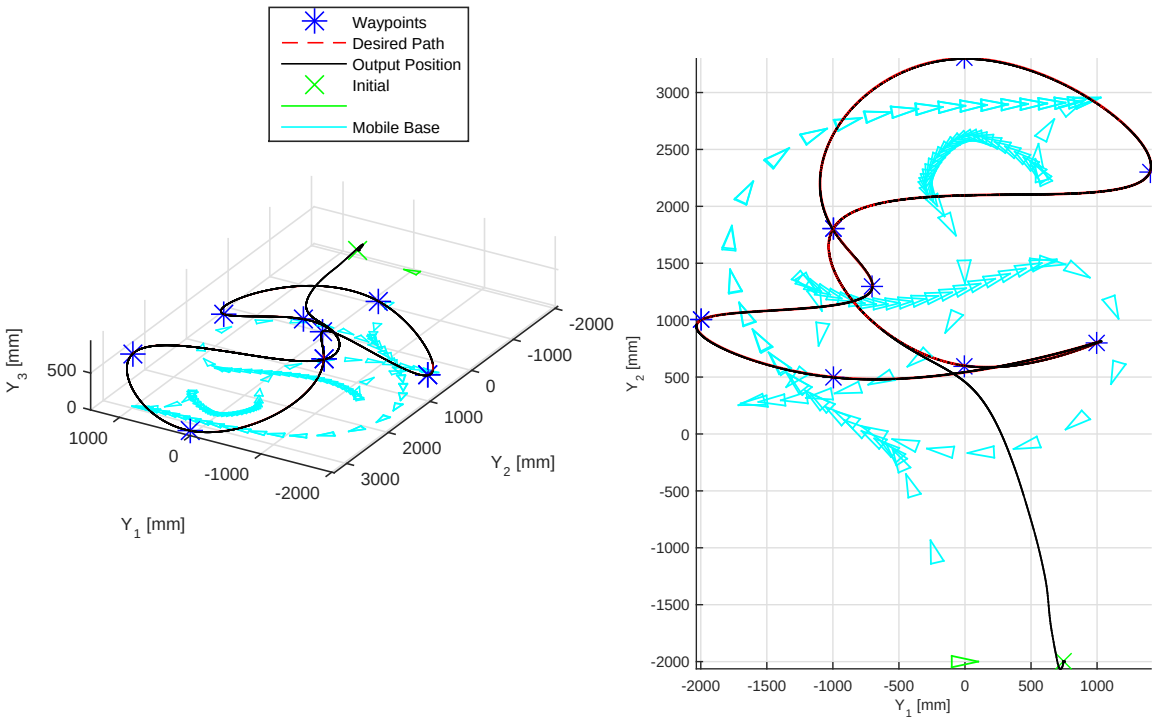

Figure 5.6: Example 5: Simulation with $\rho = -10$, with the initial pose of the mobile manipulator further away from the path. Each snapshot of the mobile base is taken every 1.5 seconds.

5.3.2 Path Following Controller

The desired path is generated by spline-interpolating 3-dimensional waypoints using quintic polynomials [34]. The control design approach then follows from Section 5.2. For this mobile manipulator system, significant modelling uncertainties arise due to inaccurate manipulator modelling (Chapter 2) and the assumption of a perfect kinematic model of the mobile base, as well as ignoring the coupling dynamics between the two. This results in imperfect cancellation of the actual system dynamics using (5.16),(5.17). The robust controller (3.20) will asymptotically stabilize the ξ^j-subsystem in the presence of modelling uncertainty. For this application, these constants are used as tuning parameters as is done in Chapter 3, thereby achieving **PF1** and **PF2**.

The η-subsystem is also linear and controllable, thus a tangential controller can be

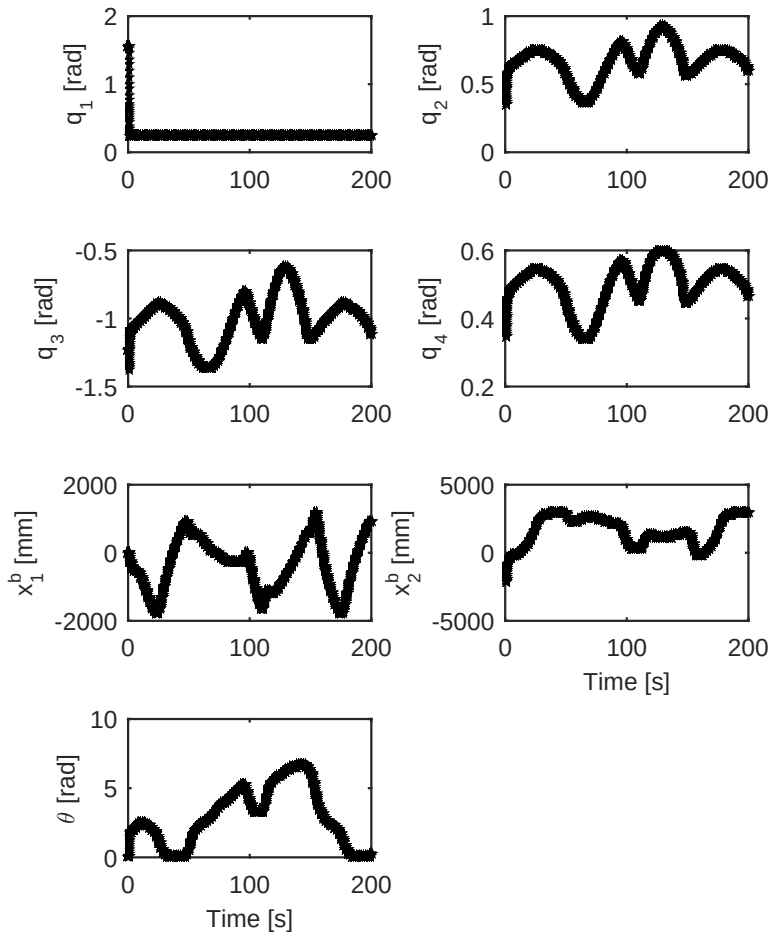

Figure 5.7: Example 5: State position $x_c = (q, x^b, \theta)$ trajectories – $\rho = -10$.

designed for v_η to track some desired tangential position or velocity profile $\eta^{\text{ref}}(t)$. Our goal is to track a desired constant velocity profile η_2^{ref}. This can be done using the PI controller (3.17), thereby achieving **PF3**. A block diagram of the entire control system can be found in Figure 5.1.

The manipulator has a Labview Real-Time Module®which is used to read the linear actuator distances using optical encoders and to control the motor PWM amplifiers. The

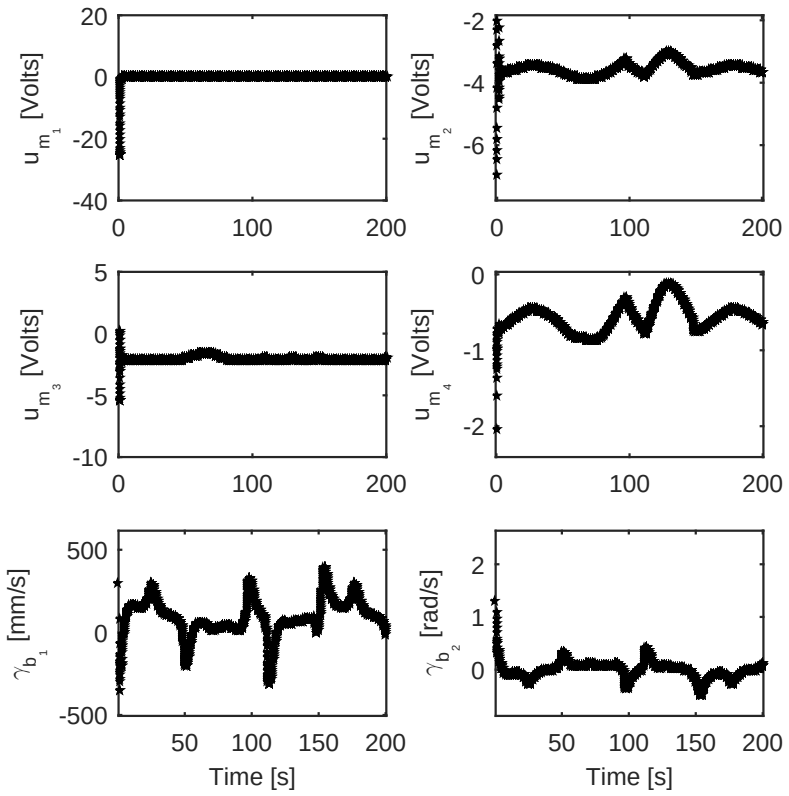

Figure 5.8: Example 5: Control inputs u_m, γ_b – $\rho = -10$.

encoder readings are converted to joint angles q and numerically differentiated to approximate $\dot{q}$. This module communicates with ROS via the Rosbridge package. The Husky is a ROS-enabled robot which takes in γ_b to control the robot, and gives out (x^b, θ) data based on a sensor fusion of wheel odometry and an on-board IMU. The path following controller is implemented using MATLAB Robotics System Toolbox.

5.3.3 Results

The joint limits in (5.22) are set so that in Figure 5.13, the waist of the manipulator prefers 90 degrees (the manipulator is ahead of the vehicle), and again in another run Figure 5.18, so that the waist prefers 160 degrees (the manipulator is to the left of the vehicle). A

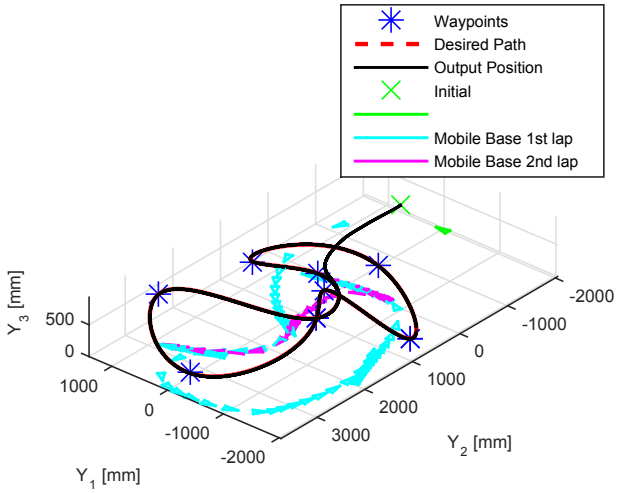

Figure 5.9: Example 6: 3D response.

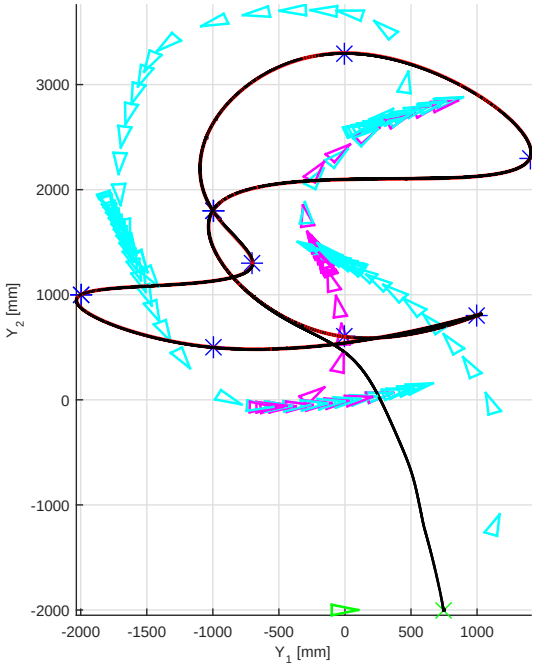

Figure 5.10: Example 6: 3D response. Each snapshot of the mobile base is taken every 1.5 seconds.

simulation of the case when the manipulator waist prefers 10 degrees (the manipulator is to the right of the vehicle) can be found in Figure 5.3. In all cases, $\eta_2^{\text{ref}} = 100\text{mm}$.

As shown in the 3D and 2D plots, the proposed path following controller successfully satisfies the goals **PF1** and **PF2**. The output y automatically goes towards the closes point on the path $\mathcal{P}$ due to the coordinate transformation employed. A key advantage to the proposed approach is the automatic coordination of the manipulator and mobile base. Based on the preferred position of the manipulator via (5.22), the mobile base automatically orients itself. In the second lap of the path (indicated in magenta), the mobile base actually moves backwards until $x^b \approx (-1000, 500)\text{mm}$, at which point the base turns and moves forward in order to keep the manipulator at its preferred position and to traverse the path at the desired rate η_2^{ref}, without any explicit trajectory planning and tracking for the mobile base.

We see that in the state trajectory Figure 5.20, the manipulator positions q quickly converge to their preferred positions when possible, in particular, the waist q_1 approaches 90°. The virtual output trajectories can be found in Figure 5.21 and show that the system

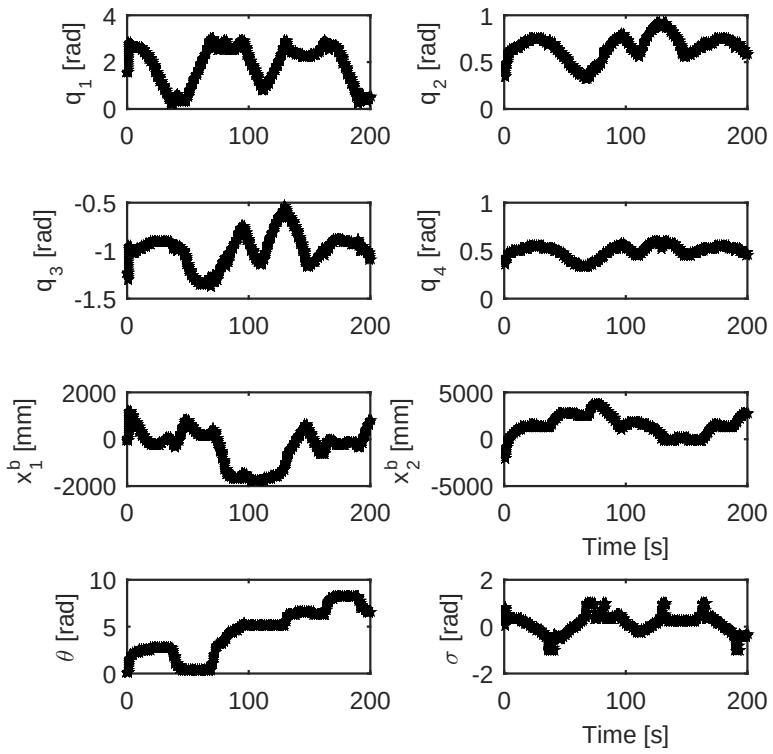

Figure 5.11: Example 6: State position $x_c = (q, x^b, \theta)$ and steering angle σ trajectories.

is traversing the path at a constant (desired) rate (**PF3**), while the transversal errors quickly approach 0 resulting in convergence to the desired path $\mathcal{P}$ (**PF1** and **PF2**). At steady state, the cross-track errors are less than ≈ 15mm.

When the preferred waist position is adjusted, the mobile base automatically takes another route for the same desired path $\mathcal{P}$. In Figure 5.19, it can be seen that the mobile base tries to stay to the right of the path in order to respect the artificial preferred position of the waist of the manipulator. Note that at the high curvature areas, the mobile base speeds up significantly (apparent by the decreased density of the mobile base snapshots) in order to respect **PF3**.

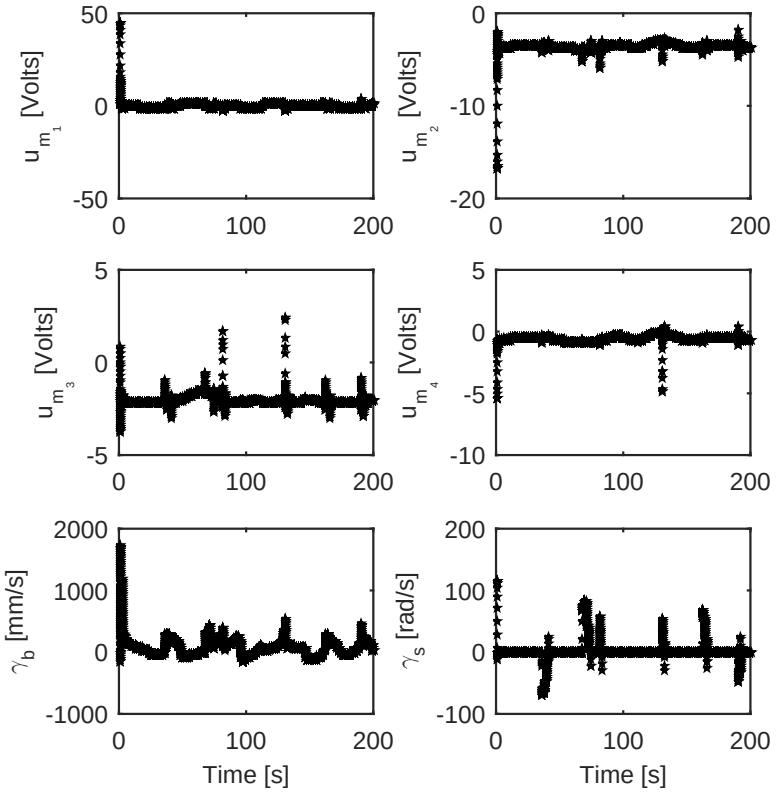

Figure 5.12: Example 6: Control inputs u_m, γ_b, γ_s.

5.4 Summary

This chapter proposes a unified path following controller for mobile manipulator systems. The controller automatically moves the mobile base and the manipulator such that the end-effector traverses a path in the output space at a desired position or velocity profile. There is no explicit trajectory required for the mobile base or the end-effector to follow. The desired path is rendered invariant and attractive, and the redundancy resolution scheme employed allows for the manipulator and vehicle steering wheels to stay away from their respective limits.

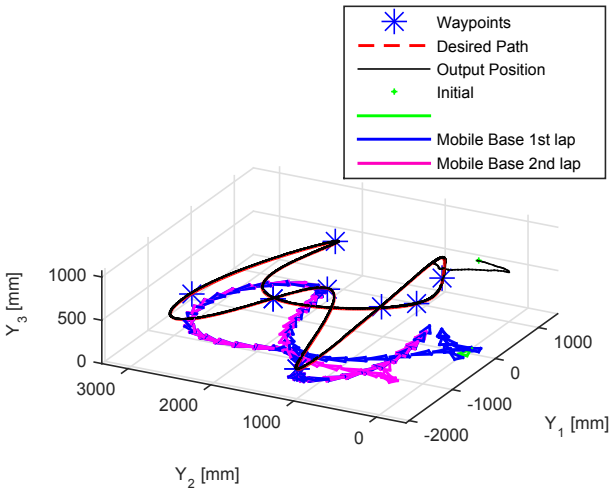

Figure 5.13: Experiment 3D response – preferred $q_1 = 90°$.

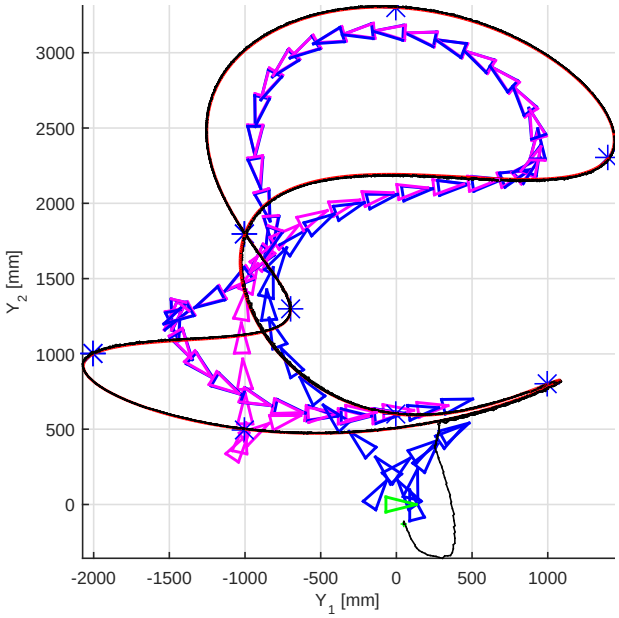

Figure 5.14: Experiment response, top view – preferred $q_1 = 90°$. Each snapshot of the mobile base is taken every 4 seconds.

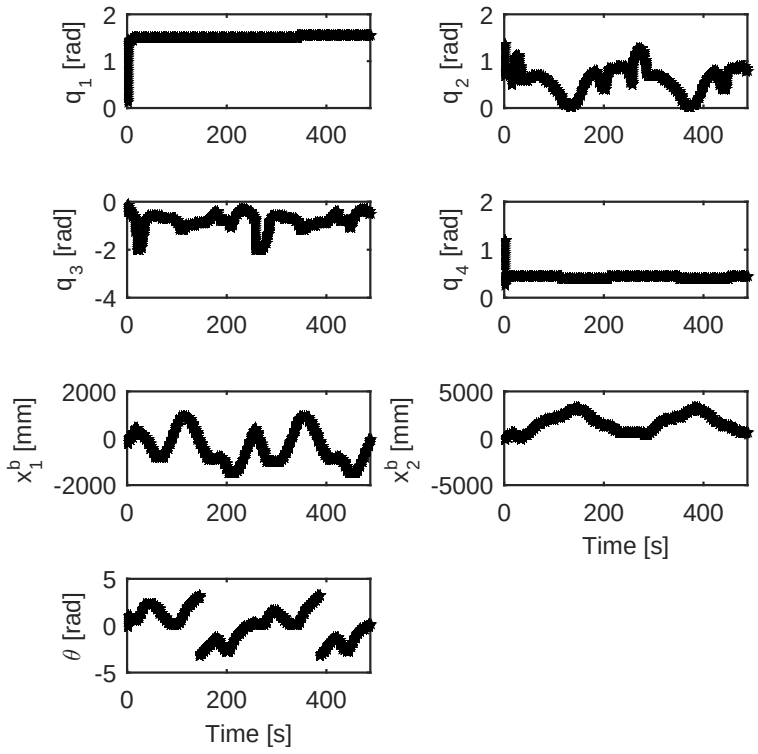

Figure 5.15: Experiment: Preferred $q_1 = 90°$. State position x_c trajectories.

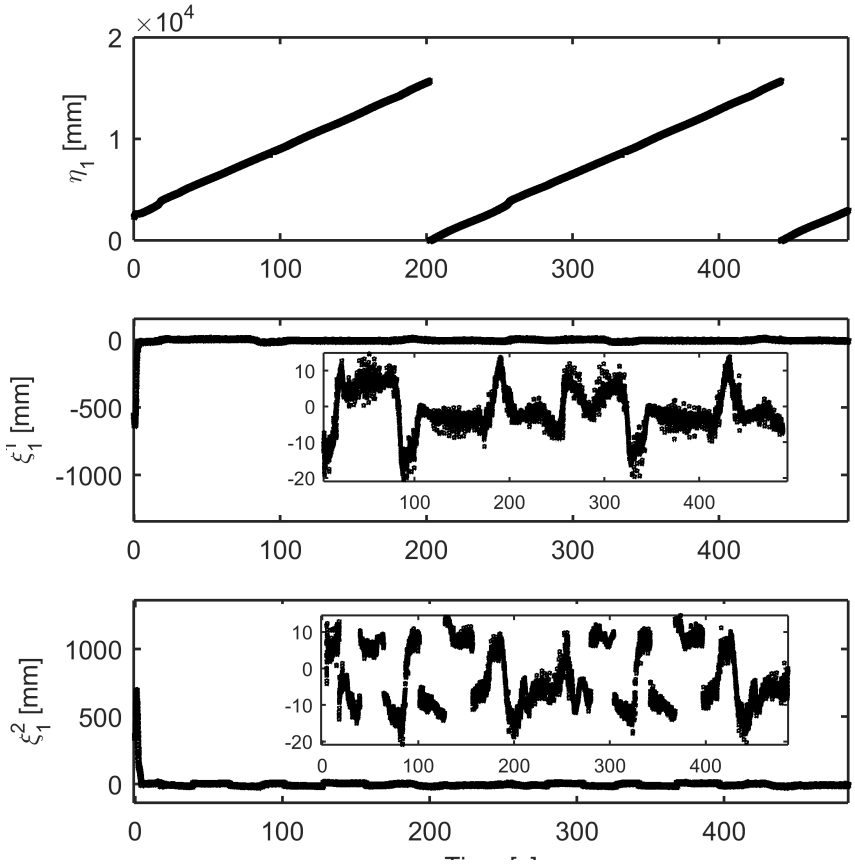

Figure 5.16: Experiment: Preferred $q_1 = 90°$. Virtual output trajectories. Notice from η_1 just over 2 laps are completed.

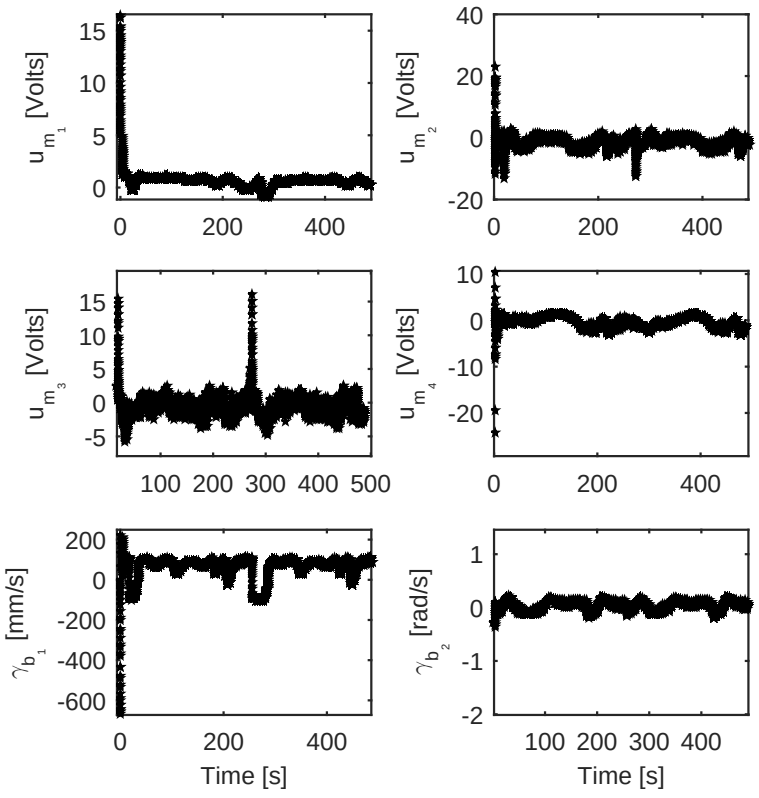

Figure 5.17: Control input u_m, γ_b – preferred $q_1 = 90°$.

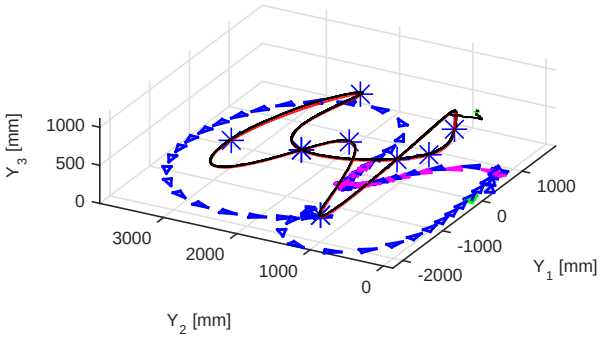

Figure 5.18: Experiment: 3D response – preferred $q_1 = 160°$.

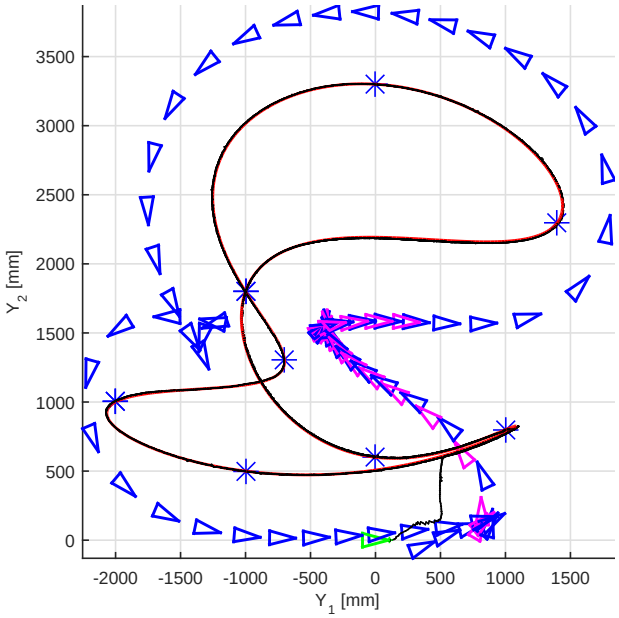

Figure 5.19: Experiment response, top view – preferred $q_1 = 160°$. Each snapshot of the mobile base is taken every 4 seconds.

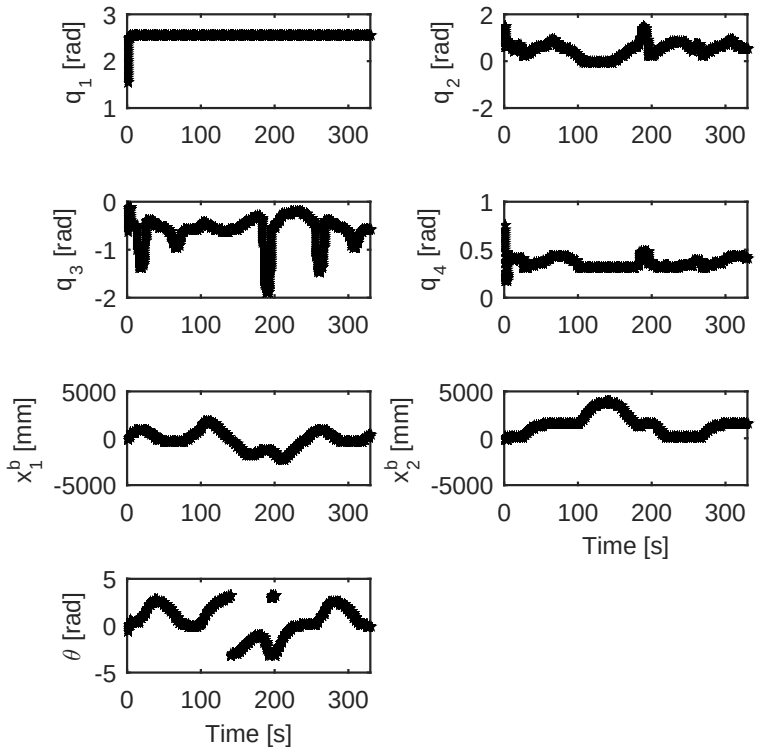

Figure 5.20: Experiment: Preferred $q_1 = 160°$. State position x_c trajectories.

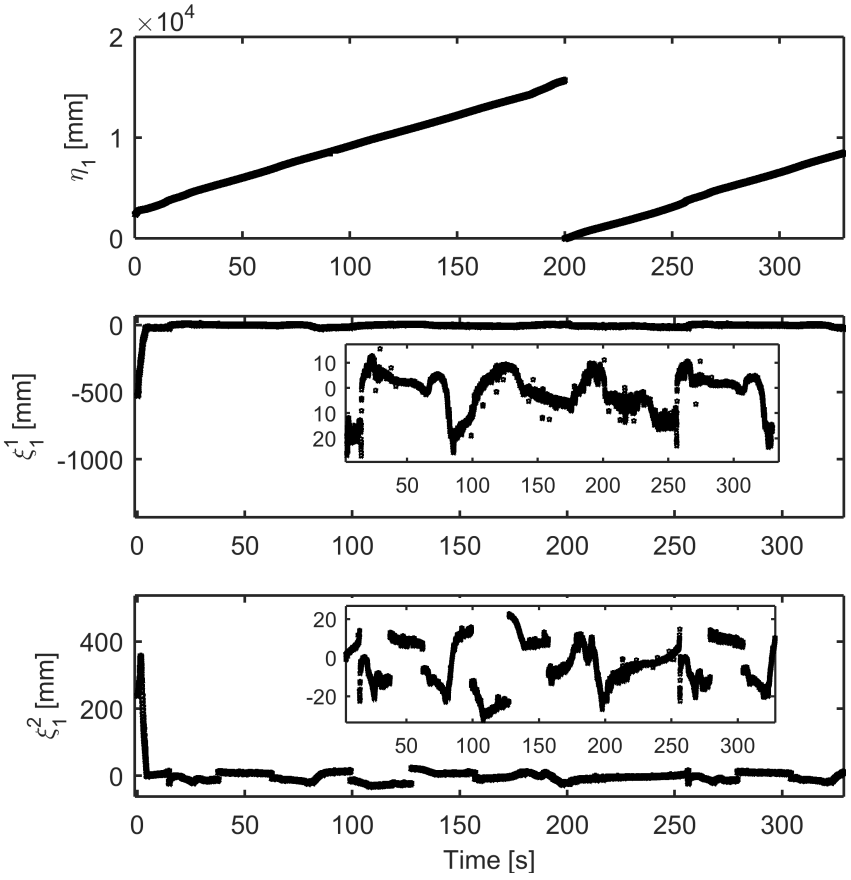

Figure 5.21: Experiment: Preferred $q_1 = 160°$. Virtual output trajectories.

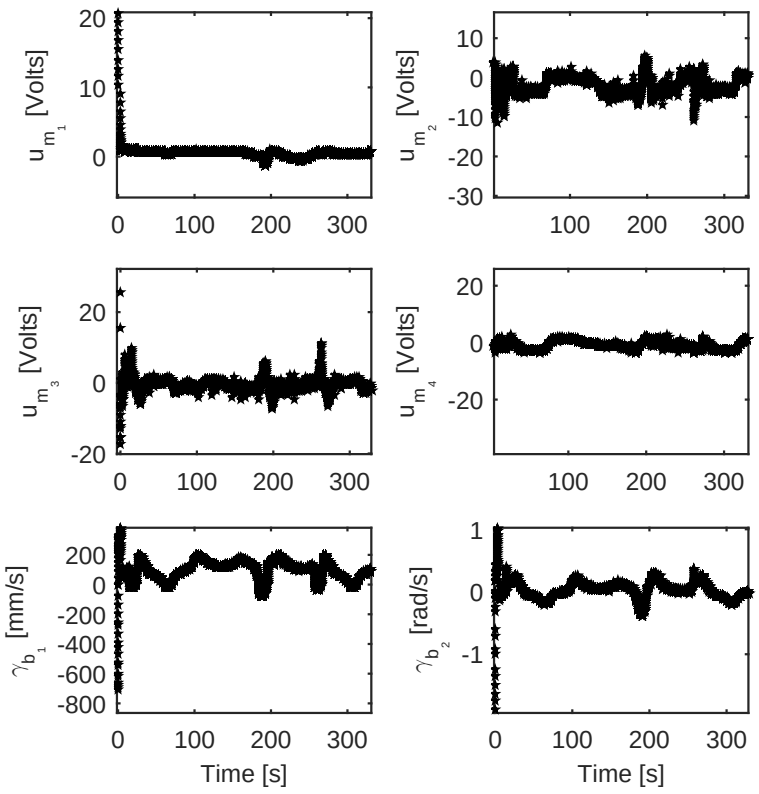

Figure 5.22: Control input u_m, γ_b – preferred $q_1 = 160°$.

Chapter 6

Conclusions and Future Work

In this thesis, we design a control framework of path following for spline paths and systems with kinematic redundancies and modelling uncertainties. In the surveyed literature, in the context of path following control, the class of paths are restrictive. In Chapter 4, these restrictions are alleviated by designing a numerical algorithm and a systematic approach to generate the zero-level set representation of splines. The robust controller developed in Chapter 3 is used to address the modelling uncertainties of most practical implementations. These contributions have key practical implications, as splines are a very powerful class of paths that allow robots to traverse any set of waypoints, and most systems have modelling errors.

In Chapter 4, a novel redundancy resolution approach was also developed that allows a redundant Euler-Lagrange system to achieve robust path following of splines while maintaining a preferred pose. In Chapter 5, this idea is applied to general mobile manipulators to yield a novel approach to unified path following for such systems. Dynamic extension with damping was used so that the virtual output has a full relative degree and to ensure boundedness of the internal dynamics. The approach automatically controls the manipulator and the ground vehicle in a parallel manner. This was shown in simulation and experiments with a 4-DOF manipulator on a car-like robot and a differential drive robot, respectively.

As the chapters progressively build up on one another, one could look at Chapter 5 as the general case of robust path following of splines for redundant mechanical systems, namely, mobile manipulators. Then, Chapter 4 shows the special case in which there is no ground vehicle. Both chapters rely on the robust controller in Chapter 3 for the practical implementation.

Future Work

The redundancy resolution approach was shown to work on examples and on the experimental platforms, and relied on a solving a constrained static optimization. Generalizing the optimization function and providing conditions for which it yields stable internal dynamics is a direction for future work. Furthermore, the static function used in the optimization can be used to avoid obstacles in the workspace by adjusting the preferred pose in real-time, which may also be direction for future work. The static optimization, however, doesn't take into account dynamic effects, such as gravity. It may be interesting to extend the redundancy resolution technique through dynamic programming.

Analyzing when the virtual output constructed in the mobile manipulator path following controller loses full relative degree is another direction for future work. This will help determine which configurations of the mobile manipulator with respect to the path following should be avoided. Dynamic programming may also be used instead of a static optimization, in order to incorporate the dynamics of the system and to respect acceleration and jerk constraints.

References

[1] A. P. Aguiar, J. P. Hespanha, and P. V. Kokotovic. Path-following for nonminimum phase systems removes performance limitations. *IEEE Transactions on Automatic Control*, 50(2):234–239, 2005. 2

[2] A. Akhtar. Dynamic path following controllers for planar mobile robots. *University of Waterloo Master's Thesis*, 2011.

[3] A. Akhtar, C. Nielsen, and S.L. Waslander. Path following using dynamic transverse feedback linearization for car-like robots. *IEEE Transactions on Robotics*, 31(2):269–279, April 2015. 3, 10, 17, 67

[4] A. Akhtar, S.L. Waslander, and C. Nielsen. Path following for a quadrotor using dynamic extension and transverse feedback linearization. In *IEEE 51st Annual Conference on Decision and Control*, pages 3551–3556, Dec 2012.

[5] M. A. Akivis and B. A. Rosenfeld. *Elie Cartan (1869-1951)*, volume 123. American Mathematical Soc., 2011. 42

[6] G. Antonelli and S. Chiaverini. Task-priority redundancy resolution for underwater vehicle-manipulator systems. In *IEEE International Conference on Robotics and Automation, 1998.*, volume 1, pages 768–773 vol.1, May 1998. 8

[7] C. Bailin, G.I. Doods, and G.W. Irwin. Time-optimal and smooth constrained path planning for robot manipulators. In *IEEE International Conference on Robotics and Automation*, pages 1853–1858 vol.3, May 1994. 5

[8] A. Banaszuk and J. Hauser. Feedback linearization of transverse dynamics for periodic orbits. *Systems & control letters*, 26(2):95–105, 1995. 7

[9] A. Banaszuk and J. Hauser. Feedback linearization of transverse dynamics for periodic orbits in r^3 with points of transverse controllability loss. *Systems & control letters*, 26(3):185–193, 1995. 7

[10] E. A. Barbashin. The method of sections in the theory of dynamical systems. *Matematicheskii Sbornik*, 71(2):233–280, 1951. 7

[11] J. T. Betts and W. P. Huffman. Path-constrained trajectory optimization using sparse sequential quadratic programming. *Journal of Guidance, Control, and Dynamics*, 16(1):59–68, 1993. 6

[12] R. Bischoff. HERMES - A Humanoid Mobile Manipulator for Service Tasks. In *Field and Service Robotics*, 1997. 9

[13] J. E. Bobrow, S. Dubowsky, and J. S. Gibson. Time-optimal control of robotic manipulators along specified paths. *The International Journal of Robotics Research*, 4(3):3–17, 1985. 2, 6

[14] J. F. Bonnans, J. C. Gilbert, and Claudia A. Lemaréchal, C. S. *Numerical optimization: theoretical and practical aspects*. Springer, 2006. 39

[15] J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4(1):25–30, 1965. 5

[16] O. Brock and O. Khatib. Elastic strips: A framework for integrated planning and execution. In *Experimental Robotics VI*, volume 250 of *Lecture Notes in Control and Information Sciences*, pages 329–338. Springer London, 2000.

[17] J. M. Cameron, D. C. MacKenzie, K. R. Ward, R. C. Arkin, and W. J. Book. Reactive control for mobile manipulation. In *IEEE International Conference on Robotics and Automation*, pages 228–235 vol.3, May 1993. 9

[18] M. F. M. Campos, G. A. S. Pereira, S. R. C. Vale, A. Q. Bracarense, G. A. Pinheiro, and M. P. Oliveira. A mobile manipulator for installation and removal of aircraft warning spheres on aerial power transmission lines. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3559–3564 vol.4, 2002. 9

[19] C. Canudas-de Wit. On the concept of virtual constraints as a tool for walking robot control and balancing. *Annual Reviews in Control*, 28(2):157–166, 2004.

[20] C. Canudas-de Wit, B. Siciliano, and B. Bastin. *Theory of Robot Control*. Springer, New York, 3^{rd}*edition*, 1997. 33

[21] M. Y. Cheng and C. C. Lee. Motion controller design for contour-following tasks based on real-time contour error estimation. *IEEE Transactions on Industrial Electronics*, 54(3):1686–1695, June 2007. 5

[22] C. Chevallereau, E. R. Westervelt, and J. W. Grizzle. Asymptotic stabilization of a five-link, four-actuator, planar bipedal runner. In *IEEE Conference on Decision and Control*, volume 1, pages 303–310. IEEE, 2004. 8

[23] G. C. Chiu and M. Tomizuka. Contouring control of machine tool feed drive systems: a task coordinate frame approach. *IEEE Transactions on Control Systems Technology*, 9(1):130–139, 2001. 5

[24] H. Y. Chuang and C. H. Liu. Cross-coupled adaptive feedrate control for multiaxis machine tools. *Journal of dynamic systems, measurement, and control*, 113(3):451–457, 1991. 5

[25] J. H. Chung and S.A. Velinsky. Modeling and control of a mobile manipulator. *Robotica*, 16(06):607–613, 1998. 16

[26] J. Connors and G. Elkaim. Analysis of a spline based, obstacle avoiding path planning algorithm. In *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*, pages 2565–2569, April 2007. 5

[27] J. Connors and G. Elkaim. Manipulating b-spline based paths for obstacle avoidance in autonomous ground vehicles. In *Proceedings of the ION National Technical Meeting*, 2007. 5

[28] L. Consolini, M. Maggiore, C. Nielsen, and M. Tosques. Path following for the pvtol aircraft. *Automatica*, 46(8):1284–1296, 2010. 21, 36, 52

[29] D. Costantinescu and E.A. Croft. Smooth and time-optimal trajectory planning for industrial manipulators along specified paths. *Journal of robotic systems*, 17(5):233–249, 2000. 6

[30] C.C. De Wit, G. Bastin, and B. Siciliano. *Theory of robot control*. Springer-Verlag New York, Inc., 1996. 16, 17, 18

[31] S. Dubowsky and E.E. Vance. Planning mobile manipulator motions considering vehicle dynamic stability constraints. In *IEEE International Conference on Robotics and Automation*, pages 1271–1276 vol.3, May 1989. 9

[32] G. Dullerud and F. Paganini. *Course in Robust Control Theory*. Springer-Verlag New York, 2000. 26

[33] M. I. El-Hawwary and M. Maggiore. Global path following for the unicycle and other results. In *American Control Conference, 2008*, pages 3500–3505. IEEE, 2008. 8

[34] K. Erkorkmaz and Y. Altintas. High speed {CNC} system design. part i: jerk limited trajectory generation and quintic spline interpolation. *International Journal of Machine Tools and Manufacture*, 41(9):1323 – 1345, 2001. 2, 3, 4, 78

[35] K. Erkorkmaz and Y. Altintas. High speed {CNC} system design. part i: jerk limited trajectory generation and quintic spline interpolation. *International Journal of Machine Tools and Manufacture*, 41(9):1323 – 1345, 2001. 33, 58

[36] K. Erkorkmaz and Y. Altintas. High speed cnc system design. part iii: high speed tracking and contouring control of feed drives. *International Journal of Machine Tools and Manufacture*, 41(11):1637–1658, 2001. 4

[37] E. Farzad and H. Khalil. Output feedback stabilization of fully linearizable systems. *International Journal of Control*, 56(5):1007–1037, 1992. 26, 27

[38] M. Fliess, J. Lvine, and P. Rouchon. Flatness and defect of nonlinear systems: Introductory theory and examples. *International Journal of Control*, 61:1327–1361, 1995. 3, 10

[39] B.A. Francis and W.M. Wonham. The internal model principle of control theory. *Automatica*, 12(5):457 – 465, 1976. 2

[40] L. B. Freidovich, U. Mettin, A. S. Shiriaev, and M. W. Spong. A passive 2-dof walker: hunting for gaits using virtual holonomic constraints. *Robotics, IEEE Transactions on*, 25(5):1202–1208, 2009.

[41] E. G. Gilbert and D. W. Johnson. Distance functions and their application to robot path planning in the presence of obstacles. *Robotics and Automation, IEEE Journal of*, 1(1):21–30, Mar 1985. 5

[42] R. Gill, D. Kulic, and C. Nielsen. Robust path following for robot manipulators. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 3412–3418. IEEE, 2013. 20

[43] R. Gill, D. Kulic, and C. Nielsen. Spline path following for redundant mechanical systems. Submitted March 2015. 32

[44] B. Hamner, S. Koterba, J. Shi, R. Simmons, and S. Singh. An autonomous mobile manipulator for assembly tasks. *Autonomous Robots*, 28(1):131–149, 2010. 9

[45] K. Harbick, J.F. Montgomery, and G. Sukhatme. Planar spline trajectory following for an autonomous helicopter. In *Computational Intelligence in Robotics and Automation, 2001. Proceedings 2001 IEEE International Symposium on*, pages 408–413, 2001. 2, 6

[46] J. Hauser and R. Hindman. Maneuver regulation from trajectory tracking: Feedback linearizable systems. In *Proc. IFAC Symp. Nonlinear Control Systems Design*, pages 595–600, 1995. 1, 2, 7

[47] J. Hauser and R. Hindman. Aggressive flight maneuvers. In *Decision and Control, 1997., Proceedings of the 36th IEEE Conference on*, volume 5, pages 4186–4191. IEEE, 1997. 7

[48] A. Hladio, C. Nielsen, and D. Wang. Path following for a class of mechanical systems. *IEEE Transactions on Control Systems Technology*, 21(6):2380–2390, 2013. 2, 8, 10, 20, 21, 22, 34, 36, 40, 43, 51, 56, 72

[49] J. M. Hollerbach. Dynamic scaling of manipulator trajectories. *Journal of Dynamic Systems, Measurement, and Control*, 106(1):102–106, 1984. 2, 6

[50] J.M. Hollerbach and Ki S. Redundancy resolution of manipulators through torque optimization. *Robotics and Automation, IEEE Journal of*, 3(4):308–316, August 1987. 9

[51] F. Inoue, T. Muralami, and K. Ihnishi. A motion control of mobile manipulator with external force. *Mechatronics, IEEE/ASME Transactions on*, 6(2):137–142, Jun 2001. 9

[52] A. Isidori. *Nonlinear control systems II*, volume 2. Springer Verlag, 1999. 3, 7, 10, 21, 44, 69, 71

[53] A. Isidori and C.I. Byrnes. Output regulation of nonlinear systems. *Automatic Control, IEEE Transactions on*, 35(2):131–140, Feb 1990. 2

[54] K. B. Judd and T. W. McLain. Spline based path planning for unmanned air vehicles. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 6–9. Montreal, Canada, 2001. 5

[55] V. Jurdjevic. *Geometric Control Theory*. Cambridge University Press, New York, 1997.

[56] K. Kant and S. W. Zucker. Toward efficient trajectory planning: The path-velocity decomposition. *The International Journal of Robotics Research*, 5(3):72–89, 1986. 2, 7

[57] H. K. Khalil. *Nonlinear Systems 3rd edition*. Prentice Hall, 2002.

[58] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *Robotics and Automation, IEEE Journal of*, 3(1):43–53, 1987. 9, 10, 51

[59] O. Khatib. Mobile manipulation: The robotic assistant. *Robotics and Autonomous Systems*, 26(23):175 – 183, 1999. Field and Service Robotics. 9, 16

[60] O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, and A. Casal. Vehicle/arm coordination and multiple mobile manipulator decentralized cooperation. In *Intelligent Robots and Systems' 96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on*, volume 2, pages 546–553. IEEE, 1996.

[61] Y. Koren. Cross-coupled biaxial computer control for manufacturing systems. *Journal of Dynamic Systems, Measurement, and Control*, 102(4):265–272, 1980. 4, 5

[62] Y. Koren and C. C. Lo. Advanced controllers for feed drives. *CIRP Annals-Manufacturing Technology*, 41(2):689–698, 1992. 2, 4, 5

[63] E. Kreyszig. *Differential Geometry*. Differential Geometry. Dover Publications, 1991.

[64] B. C. Kuo and M. F. Golnaraghi. *Automatic control systems*, volume 4. John Wiley & Sons, 2003. 26

[65] L. Lapierre and D. Soetanto. Nonlinear path-following control of an auv. *Ocean engineering*, 34(11):1734–1744, 2007. 2

[66] M. Lepetic, G. Klancar, I. Skrjanc, D. Matko, and B. Potocnik. Time optimal path planning considering acceleration limits. *Robotics and Autonomous Systems*, 45(34):199 – 210, 2003. 5

[67] M. Lepetic, G. Klancar, I. Skrjanc, D. Matko, and B. Potocnik. Path optimisation considering dynamic constraints. 3276:574–585, 2005. 5

[68] Y. Lin, E. D. Sontag, and Y. Wang. Input to state stabilizability for parametrized families of systems. *International Journal of Robust and Nonlinear Control*, 5(3):187–205, 1995. 7

[69] C. C. Lo. A tool-path control scheme for five-axis machine tools. *International Journal of Machine Tools and Manufacture*, 42(1):79 – 88, 2002. 2, 4

[70] T. Lozano-Pérez and M. A. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570, 1979. 2, 6

[71] U. Mettin, P. X. La Hera, D. O. Morales, A. S. Shiriaev, L. B. Freidovich, and S. Westerberg. Path-constrained trajectory planning and time-independent motion control: Application to a forestry crane. In *14th International Conference on Advanced Robotics (ICAR), Proceedings*, 2009.

[72] 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. 2

[73] Keiji N. and S. Yuta. Designing strategy and implementation of mobile manipulator control system for opening door. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 2828–2834 vol.3, Apr 1996. 9

[74] K. Nagatani, T. Hirayama, A. Gofuku, and Y. Tanaka. Motion planning for mobile manipulator with keeping manipulability. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1663–1668 vol.2, 2002.

[75] C. Nielsen, C. Fulford, and M. Maggiore. Path following using transverse feedback linearization: Application to a maglev positioning system. *Automatica*, 46(3):585–590, 2010. 22

[76] C. Nielsen and M. Maggiore. Output stabilization and maneuver regulation: A geometric approach. *Systems & control letters*, 55(5):418–427, 2006. 8, 65

[77] C. Nielsen and M. Maggiore. On local transverse feedback linearization. *SIAM Journal on Control and Optimization*, 47(5):2227–2250, 2008. 7

[78] H. Nijmeijer and A. Van der Schaft. *Nonlinear dynamical control systems*. Springer, 1990. 7

[79] R. L. Norton. *Machine design: an integrated approach*. Prentice-Hall New York, 1996. 15

[80] G. R. Pennock and D. J. Kassner. The workspace of a general geometry planar three-degree-of-freedom platform-type manipulator. *Journal of Mechanical Design*, 115(2):269–276, 1993. 16

[81] L. Peterson, D. Austin, and D. Kragic. High-level control of a mobile manipulator for door opening. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2333–2338 vol.3, 2000. 9

[82] H. Poincaré and R. Magini. Les méthodes nouvelles de la mécanique céleste. *Il Nuovo Cimento (1895-1900)*, 10(1):128–130, 1899. 7

[83] A. N. Poo, J. G. Bollinger, and G. W. Younkin. Dynamic errors in type 1 contouring systems. *IEEE Transactions on Industry Applications*, (4):477–484, 1972. 5

[84] T. Pu-Sheng, L.S. Wang, C. Fan-Ren, and W. Ter-Fen. Systematic backstepping design for b-spline trajectory tracking control of the mobile robot in hierarchical model. In *IEEE International Conference on Networking, Sensing and Control*, volume 2, pages 713–718 Vol.2, 2004. 2, 6

[85] S. Quinlan and O. Khatib. Elastic bands: connecting path planning and control. In *IEEE International Conference on Robotics and Automation*, pages 802–807 vol.2, May 1993.

[86] R. Ramesh, M. A. Mannan, and A. N. Poo. Tracking and contour error control in cnc servo systems. *International Journal of Machine Tools and Manufacture*, 45(3):301–326, 2005. 2, 4

[87] N. Rouche, P. Habets, M. Laloy, and A. M. Ljapunov. *Stability theory by Liapunov's direct method*. Springer, 1977. 7

[88] E. Roxin. Axiomatic foundation of the theory of control systems. In *Proceedings 2nd Int. Conf. IFAC, Basel*, pages 640–644, 1963. 7

[89] M. Saska, M. Macas, L. Preucil, and L. Lhotska. Robot path planning using particle swarm optimization of ferguson splines. In *IEEE Conference on Emerging Technologies and Factory Automation*, pages 833–839. IEEE, 2006. 5, 6

[90] M. Scheint, M. Sobotka, and M. Buss. Virtual holonomic constraint approach for planar bipedal walking robots extended to double support. In *IEEE Conference on Decision and Control*, pages 8180–8185. IEEE, 2009.

[91] B. Sencer and Y. Altintas. Modeling and control of contouring errors for five-axis machine toolspart ii: Precision contour controller design. *Journal of manufacturing science and engineering*, 131(3):031007, 2009. 4, 5

[92] H. Seraji. A unified approach to motion control of mobile manipulators. *The International Journal of Robotics Research*, 17(2):107–118, 1998. 9

[93] A. Shiriaev, J. W. Perram, and C. Canudas-de Wit. Constructive tool for orbital stabilization of underactuated nonlinear systems: Virtual constraints approach. *IEEE Transactions on Automatic Control*, 50(8):1164–1176, 2005. 8

[94] A. S. Shiriaev, L. B. Freidovich, A. Robertsson, R. Johansson, and A. Sandberg. Virtual-holonomic-constraints-based design of stable oscillations of furuta pendulum: Theory and experiments. *IEEE Transactions on Robotics*, 23(4):827–832, 2007. 8

[95] P. Sicard and M. D. Levine. An approach to an expert robot welding system. *IEEE Transactions on Systems, Man and Cybernetics*, 18(2):204–222, 1988.

[96] R. Skjetne, T. I. Fossen, and P. V. Kokotović. Robust output maneuvering for a class of nonlinear systems. *Automatica*, 40(3):373–383, 2004. 2, 7

[97] M. Song, T. Tarn, and N. Xi. Integration of task scheduling, action planning, and control in robotic manufacturing systems. *Proceedings of the IEEE*, 88(7):1097–1107, 2000. 2, 6

[98] M. W. Spong, S. Hutchinson, and M. Vidyasagar. *Robot modeling and control*. John Wiley & Sons Hoboken^ eNJ NJ, 2006. 13, 28, 29, 51, 55

[99] K. Srinivasan. Machine tool feed drives and their controla survey of the state of the art. *Urbana*, 51:61801, 1997. 5

[100] T. J. Tarn, N. Xi, and A. K. Bejczy. Path-based approach to integrated planning and control for robotic systems: A path-based approach provides an analytical integration of robot planning and control. as a result, the task level control can be achieved, and the system is capable of coping with unexpected or uncertain events. *Automatica*, 32(12):1675 – 1687, 1996. 2, 6

[101] M. Tomizuka. Zero phase error tracking algorithm for digital control. *Journal of Dynamic Systems, Measurement, and Control*, 109(1):65–68, 1987. 2, 4

[102] M. Toussaint. Some notes on gradient descent. *Machine Learning & Robotics Lab, FU Berlin*, 2012. 39, 40

[103] V. I. Utkin. *Sliding modes in control and optimization*, volume 116. Springer, 1992. 8

[104] B. Van der Pol. Lxxxviii. on relaxation-oscillations. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):978–992, 1926. 7

[105] F.-C. Wang and D.C.H. Yang. Nearly arc-length parameterized quintic-spline interpolation for precision machining. *Computer-Aided Design*, 25(5):281 – 288, 1993. 2, 4

[106] S. Westerberg, U. Mettin, A. S. Shiriaev, L. B. Freidovich, and Y. Orlov. Motion planning and control of a simplified helicopter model based on virtual holonomic constraints. In *International Conference on Advanced Robotics*, pages 1–6. IEEE, 2009. 8

[107] Y. Xu, H.B. Jr. Brown, M. Friedman, and T. Kanade. Control system of the self-mobile space manipulator. *IEEE Transactions on Control Systems Technology*, 2(3):207–219, Sep 1994. 9

[108] Y. Yamamoto and X. Yun. Coordinating locomotion and manipulation of a mobile manipulator. In *IEEE Conference on Decision and Control*, pages 2643–2648. IEEE, 1992. 9

[109] V. I. Zubov and L. F. Boron. *Methods of AM Lyapunov and their Application*. Noordhoff Groningen, 1964. 7