Stabilization of Brachiation Locomotion in a Monkey Robot

Stian Hjellvik Askeland

Master of Science in Engineering Cybernetics

Submission date:	June 2012
Supervisor:	Anton Shiriaev, ITK
Co-supervisor:	Uwe Mettin, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Problem Description

Consider a 24-degrees-of-freedom monkey robot that is supposed to perform brachiating locomotion, i.e. swinging from one row of a horizontal ladder to the next one using the arms. The robot hand is constructed as a planar hook so that the contact point about which the robot swings is a passive hinge and the mechanical system is underactuated in single-support phase. Assume that a suitable hybrid dynamical model of the robot exists and that feasible coordination patterns (virtual constraints) and periodic trajectories of the joints are known so that the dynamic constraints associated with the passive degree of freedom are satisfied. The task is to design a controller that is able to stabilize such a feasible brachiating motion. As a step in doing so a toolbox aiding the process of controller design should be developed, tested and verified.

Abstract

Achieving robotic locomotion is in general a difficult task. When the system of concern is underactuated, i.e. it has more degrees of freedom than the number of control inputs available, dynamic constraints are imposed, further complicating the task. This is the case for the brachiation motion observed in the lesser apes, i.e. gibbons and siamangs, as the gait involves periods of time at which the ape is suspended by one arm with limited torques available to influence the rotation about the handhold. Earlier work has been concerned with modeling of a 24-degrees-of-freedom monkey robot and the design of a brachiation gait. In this thesis we develop a toolbox to facilitate the design of a controller based on transverse linearization for this brachiation gait. The main focus is to stabilize the single-support part of the gait, i.e. the part that is subjected to dynamic constraints due to the lack of torque about the handhold, as traditional control theory is unable to stabilize the desired motion in this case. The developed toolbox is used in designing a controller that orbitally stabilizes an inverted pendulum system. As an initial step in achieving orbital stabilization of the brachiating gait, asymptotic convergence to the virtual holonomic constraints is demonstrated for a simplified model of the 24 degrees-of-freedom monkey robot.

Sammendrag

Å realisere en ønsket bevegelse i en robot er generelt en vanskelig oppgave. Når systemet i tillegg er underaktuert, dvs. at det har flere frihetsgrader enn antall tilgjengelige aktuatorer, blir systemet påtvinget dynamiske begrensninger og oppgaven kompliseres ytterligere. Dette er tilfellet for fremkomstmetoden som er observert når de langarmede apene slenger seg etter armene. Denne fremkomstmetoden innebærer perioder hvor apen henger etter en arm med begrenset dreiemoment tilgjengelig rundt punktet den griper. Tidligere arbeid har utviklet en forenklet modell for en ape-robot med 24 frihetsgrader og koordinatbaner er funnet slik at en bevegelse inspirert av disse langarmede apene kan realiseres. I denne avhandlingen er målet å utvikle en kontroller som stabiliserer disse planlagte koordinatbanene. For å lette utviklingen av en slik kontroller utvikles også en verktøykasse for å utføre noen av de tunge beregningene som kreves i prosessen. Denne verktøykassen demonstreres så på et system bestående av en underaktuert invertert pendel, og konvergens til de ønskede periodiske banene oppnås. Som et første skritt mot å oppnå stabilisering av den ønskede fremkomstmetoden for aperoboten demonstreres det at de aktuerte frihetsgradene relativt enkelt kan stabiliseres til de ønskede koordinatbanene.

Contents

Chapter 1

Introduction

Brachiation is the locomotion observed in the lesser apes (i.e. siamangs and gibbons), and is described as a specialized form of arboreal locomotion in which movement is accomplished by swinging from one hold to another by the arms [7]. These primates are capable of energy efficient locomotion by swinging from one arm to the next, allowing them to travel at high speeds through the treetops.

The great apes, (chimpanzees, gorillas, orangutans and humans), are also capable of brachiation. However, in the great apes, brachiation is not the primary form of locomotion, but one that might be used alongside other locomotion styles, such as bipedal and quadrupedal locomotion. This allows great versatility as a change of locomotion style can be made for better adaption to a change in the environments. For instance might bipedal walking be used in tight spaces, quadrupedal walking on rough terrain and brachiation to overcome obstacles.

Anthropomorphic (human-like) robots often aim to replace people in dangerous environments. The versatility of the great apes would clearly be an advantage in this case. A multi-locomotion robot has been developed in an attempt to mimic these charactersistics of the greater apes [5]. The proposed multi-locomotion robot aims to achieve biped locomotion, quadruped locomotion and brachiation while being able to switch between locomotion styles to best adapt to the environment.

The problem of achieving brachiation in a physical robot can be considered dividable into three main parts: modeling, trajectory design and controller design. Earlier work [1] has been concerned with modeling and trajectory design for this brachiating robot, and a robust and energy-efficient brachiating gait was found. In this thesis we aim to design a controller that stabilizes this brachiation gait. As an aid in doing so a MATLAB toolbox is developed to perform parts of the computations that need to be done before arriving at a stabilizing controller.

Chapter 2

Problem Formulation

Earlier work [1] has been concerned with modeling and planning a feasible trajectory for the gorilla robot and both models and feasible trajectories were found. However, in order to achieve the planned brachiating motion in an actual physical gorilla robot, a controller is needed. Several controllers have been implemented on various types of brachiating robots (e.g. [11], [6], [10], [5]) with fairly good results. In this work an attempt is made to use the concept of transverse linearization in order to find a controller that stabilizes the desired trajectory. This concept is described in e.g. [14] and [15], and demonstrations of the concept to a variety of systems are made. Transverse linearization has, to the authors knowledge, not yet been used to stabilize a brachiation locomotion. This is the primary goal of this work; to demonstrate successful stabilization of a brachiating motion using transverse linearization.

2.1 System Description

The robot in question is a humanoid robot with 24 degrees of freedom built at Professor Fukuda's lab, Nagoya University, Japan. A schematic of the robot is shown in Figure 2.1.

This robot can be modeled as an Euler-Lagrange system with n states and m inputs.

$$\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial \mathcal{L}(q,\dot{q})}{\partial \dot{q}}\right) - \frac{\partial \mathcal{L}(q,\dot{q})}{\partial q} = B(q)u \tag{2.1}$$

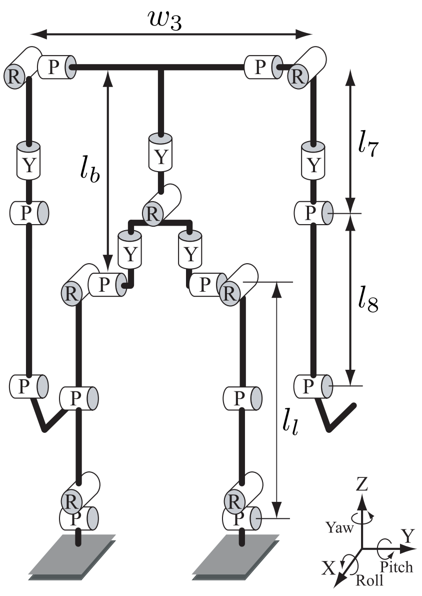


Figure 2.1: Schematic of the 24 DOF robot built at Professor Fukuda's Lab, Nagoya University, Japan

where $q \in \mathbb{R}^n$ is a vector of generalized coordinates and $u \in \mathbb{R}^m$ is a vector of control inputs. During the swing-part of the brachiating motion, the system is underactuated, i.e. the number of independent control inputs is smaller than the number of generalized coordinates. This can be written as

$$n = m + p \tag{2.2}$$

where $p > 0$ is the degree of underactuation.

It is well-known that this system can be rewritten as [16]

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = B(q)u \tag{2.3}$$

Where M is a symmetric, positive definite inertia matrix, C contains the centrifugal and Coriolis terms, G contains the partial derivative of the potential energy of the system with respect to each generalized coordinates. B is a matrix function of constant rank m describing how the control inputs influence the dynamics.

Due to the complexity of modeling and designing a trajectory for all 24 degrees of freedom of the robot, only the 6 degrees of freedom believed to be the most relevant for the brachiating gait is considered when a model is derived. Of these, 5 corresponds to actual joint angles on the robot, while one is the angle about the handhold from which the robot is suspended. The gripping hand is modeled as a planar hook, and there is no torque available about the handhold. This coordinate is therefore considered unactuated and the system as a whole is underactuated of degree one, i.e. we have $n = 6$ generalized coordinates, and $m = 5$ control inputs.

2.2 Brachiation

The brachiation motion consists of successive single-support and double-support phases. The single-support phase refers to the parts of the gait in which the robot is suspended from one arm, while the double-support phase refers to the part where both arms are firmly gripping individual handholds. Because of the pendulum-like motion of the center of mass observed during the single-support phase, this will also be referred to as the swing-phase. During the double-support phase, gaits observed in nature involve moving the center of mass backwards [17]. For this reason the double-support phase will also be referred to as the swing-back-phase or the loop-phase.

Based on analysis of the actual brachiation motion observed in nature [17], a brachiation motion consisting of the following phases is proposed [1].

Initial swing phase The robot is assumed to be initialized in a position where it is holding on to two successive handholds with zero joint velocities. From this initial position, an initial swing phase is entered. This phase is similar to the single-support phase, except that it is initialized with zero joint velocities (opposed to the single-support phase which is initialized with non-zero joint velocities).

Double-support phase This phase starts when the robot grips the front handhold with the free hand. The goal of this phase is to move the center of mass backward and align it with the desired initial conditions for the swing phase. During this phase the system is fully actuated, as the available actuators are able to move the gorilla to any desired configuration, given that enough torque is available. The proposed double-support phase does indeed keep the required torques within reasonable levels.

Single-support phase This phase starts when the robot releases its rear handhold and starts moving the rear hand forward to reach the front handhold. During this phase the system is underactuated due to the lack of torque about the handhold from which the gorilla is suspended.

Snapshots of the proposed gait is shown in Figure 2.2.

The gait is designed in such a way that transitions between the various phases does not result in any impact forces, i.e. the velocities of each joint at the end of one phase equals the initial velocities at the start of the next phase. This ensures minimal collision loss and allows the impact map to be written as simply a relabeling of the coordinates [1].

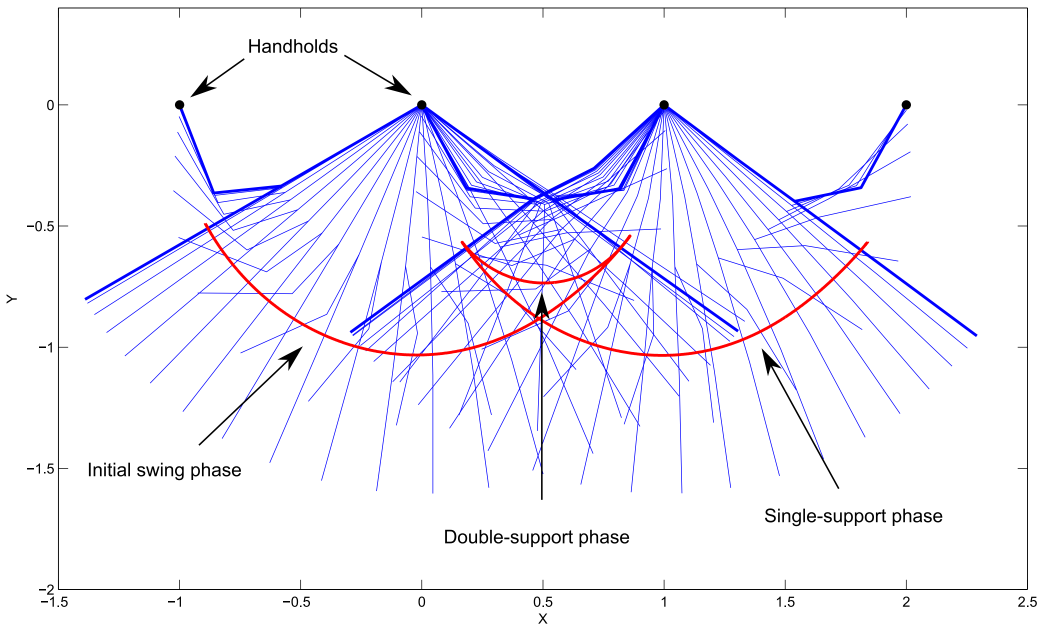


Figure 2.2: Snapshots of the time-evolution of the full gait. The red line is the trajectory traced by the center of mass. The emphasized configurations shows where phase transitions occur.

2.3 Why Controlling the Monkey Robot is Difficult

Achieving brachiation is in general a difficult task. This is because the brachiation motion is naturally under-actuated. From the moment the monkey (or robot) releases the rear handhold, the single-support phase is initiated. Due to the limited torque available for gripping, the angle about the front handhold is considered passive, and the single-support phase is governed by passive, pendulum-like dynamics. These passive dynamics impose dynamic constraints on the motion, complicating gait synthesis and control. Because of the highly limited torque available about the handhold this motion one can not simply apply the required torque about the handhold in order to swing the body forward. Locomotion must be achieved in some other manner, namely by applying the torques that are available about the actuated joints in such a way that the body of the robot does indeed end up in some desired position in front of the handhold allowing the free hand to grip this handhold

2.4 Orbital Stabilization

The brachiation gait consists of two continuous motions that are connected by discrete jumps. The gait is designed to be periodic such that the state at the end of one period equals the initial state for the next period. The motion thus constitutes closed periodic hybrid orbits, consisting of two continuous parts and two jumps. The problem of stabilizing a brachiating gait is thus equivalent to orbital stabilization of the desired periodic hybrid orbits. The concept of transverse linearization described in e.g. [14] [15] [9], can be used to asymptotically stabilize such periodic orbits. This is the approach taken in this thesis; to stabilize the desired periodic orbits using transverse linearization.

Using transverse linearization for orbital stabilization requires numerous complex computations. To facilitate controller synthesis we therefore aim to develop a MATLAB toolbox aiding the process. Such a toolbox could take as input a model of the system as well as some description of the periodic orbit to be stabilized, and then automate parts of the controller synthesis process. To the authors knowledge, no such toolbox exist at present date and the development of this is one of the main contributions of this work.

As the monkey robot is a fairly complex system, even when the system is reduced to the 6 most relevant degrees of freedom, testing and verifying the various parts of the controller design may prove challenging. For this reason, the inverted pendulum system will be used for this purpose. Initially performing the controller synthesis process on the inverted pendulum system first will allow manual verification of the various parts of the controller, significantly simplifying the detection of possible implementation errors.

Chapter 3

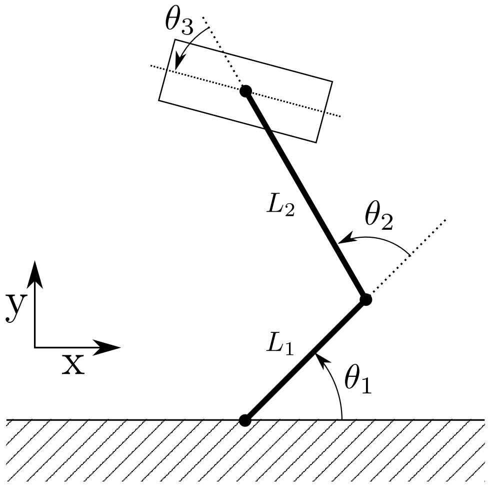
Theory

In this chapter we will introduce some underlying theoretical concepts that are needed when designing a controller for the brachiating motion using transverse linearization.

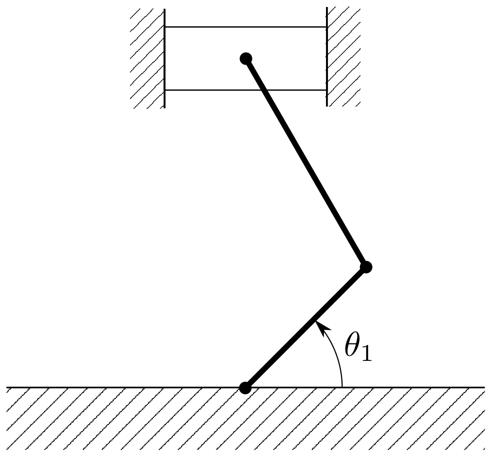
3.1 Virtual Holonomic Constraints

Both the gorilla robot and the inverted pendulum are underactuated systems. This means that not all generalized coordinates can be controlled with respect to time; one or more coordinates have dynamics which are not directly controllable through the control inputs. Controllers used on fully actuated systems typically aims to stabilize some pre-defined reference trajectory with respect to time. When the system in question is underactuated, this can no longer be done. It is certainly possible to define the desired trajectory as a function of time, but attempting to stabilize this would cause the controller to play ”catch-up” with time if the system states were to lag behind the trajectory reference for some reason [4]. For this reason it is desirable to express the desired trajectory not as a function of time, but as a function of the states. This is what virtual holonomic constraints accomplish. The concept of virtual holonomic constraints were first introduced as an aid in achieving underactuated legged locomotion in the dynamic walking testbed, RABBIT [4].

To introduce virtual holonomic constraints, consider the two-link piston system depicted in Figure 3.1a. The position of the piston is in this case given



(a) Unconstrained.



(b) Physically constrained.

Figure 3.1: Double-linked pistons.

by the angles θ_1 through θ_3 and the lengths of the links, L_1 and L_2. Consider the physically constrained piston depicted in Figure 3.1b. In this case the x-coordinate of the piston center is fixed[1] to $x = 0$ and the following relations are imposed by the physical constraints

$$\begin{aligned} L_1 \cos(\theta_1) + L_2 \cos(\theta_1 + \theta_2) &= 0 \\ \theta_1 + \theta_2 + \theta_3 - \pi &= 0 \end{aligned}$$

These relations hold as a result of d'Alambert's principle, i.e. any virtual displacement of the piston would cause a force to be exerted from the physical constraint such that the displacement remain zero.

Suppose that the two joints θ_2 and θ_3 are actuated, while the crank angle, θ_1 is unactuated. It becomes evident that the same constraints can be imposed asymptotically by zeroing the two error functions

$$y_1 = L_1 \cos(\theta_1) + L_2 \cos(\theta_1 + \theta_2) \tag{3.1a}$$

$$y_2 = \theta_1 + \theta_2 + \theta_3 - \pi \tag{3.1b}$$

If this is done by some control law rather than by some physical construction, the constraints are called virtual holonomic constraints.

[1]We could choose to constrain the piston to an arbitrary value $x = a$ as long as $|a| \leq L1 + L2$, but for simplicity we assume $a = 0$ in this example.

The equations (3.1) can be rewritten by first solving (3.1a) for θ_2 and θ_3, and then inserting into (3.1), leading to the alternate output functions

$$y_1 = \theta_2 - \left(\pi - \theta_1 - \arccos\left(\frac{L_1}{L_2}\cos(\theta_1)\right)\right) \tag{3.2a}$$

$$y_2 = \theta_3 - \arccos\left(\frac{L_1}{L_2}\cos(\theta_1)\right) \tag{3.2b}$$

Rewriting to vector functions this can be written as [2]

$$y = h_0(q) - \Phi(\theta(q)) \tag{3.3}$$

where

$$\begin{aligned} h_0(q) &= \begin{bmatrix} \theta_2 \\ \theta_3 \end{bmatrix} \\ \Phi(\theta) &= \begin{bmatrix} \pi - \theta - \arccos\left(\frac{L_1}{L_2}\cos(\theta)\right) \\ \arccos\left(\frac{L_1}{L_2}\cos(\theta)\right) \end{bmatrix} \\ \theta(q) &= \theta_1 \end{aligned} \tag{3.4}$$

In general $h_0(q)$ is some mapping from the generalized coordinates to independent quantities to be controlled, $\theta(q)$ is a scalar function of the coordinates that is monotonically increasing along the desired trajectory and $\Phi(\theta)$ specifies the virtual holonomic constraints.

Throughout the rest of this work it is assumed that $h_0(q) = q$, i.e. the quantities to be controlled are the generalized coordinates themselves.

3.1.1 Trajectory Design

By using the virtual holonomic constraints, it is possible to describe the desired trajectory of the coordinates q as a function of the monotonically increasing scalar function θ. We do this by specifying that the quantities to be controlled are the coordinates themselves[3], and assuming that the output function (3.3) is kept at zero, i.e.

$$h_0(q) = q \tag{3.5}$$

$$y = 0 \tag{3.6}$$

[2]In [4] the notation $h_d(\theta)$ is used for the virtual holonomic constraints. In this thesis a notation that is consistent with the one in e.g. [14] is used.

[3]This is not necessarily the case, but is true for the systems considered in this thesis

so that (3.3) can be rewritten to express the desired trajectories of q

$$q = \Phi(\theta) \tag{3.7}$$

Differentiating this leads to

$$\begin{aligned}\dot{q} &= \Phi'(\theta)\dot{\theta}\\ \ddot{q} &= \Phi''(\theta)\dot{\theta}^2 + \Phi'(\theta)\ddot{\theta}\end{aligned}$$

This eliminates the explicit time-dependency in the desired trajectories and by designing a controller that asymptotically zeros the difference between the generalized coordinates and the virtual holonomic constraints, the problem where the controller plays ”catch-up” with the time-varying trajectory reference is avoided. However, time is not completely eliminated from the system as the unactuated coordinate, θ is still a function of time. Consider the case where the virtual holonomic constraints are exactly satisfied . Then the system (2.3) can be rewritten as [4]

$$M(\Phi)\left(\Phi''\dot{\theta}^2 + \Phi'\ddot{\theta}\right) + C(\Phi,\Phi')\Phi'\dot{\theta}^2 + G(\Phi) = B(\Phi)u \tag{3.8}$$

Here the fact that $C(q,\dot{q})$ is linear in $\dot{q}$ is used. This can be seen from the derivations of $C(q,\dot{q})$ shown in e.g. [16].

Because the system is underactuated, the rank of the $n \times (n-1)$ matrix function $B(q)$ is $(n-1)$. Hence there exists a $1 \times n$ matrix function $B_\perp(q)$ such that $B_\perp(q)B(q) = 0,\ \forall\, q$ [14]. Premultiplying (3.8) by $B_\perp(\Phi(\theta))$ then eliminates the right hand side such that

$$\begin{aligned}0 &= B_\perp(\Phi)M(\Phi)\left(\Phi''\dot{\theta}^2 + \Phi'\ddot{\theta}\right) + B_\perp(\Phi)C(\Phi,\Phi')\Phi\dot{\theta}^2 + B_\perp(\Phi)G(\Phi)\\ &= B_\perp(\Phi)M(\Phi)\Phi'\ddot{\theta} + B_\perp(\Phi)\Big(M(\Phi)\Phi'' + C(\Phi,\Phi')\Phi\Big)\dot{\theta}^2 + B_\perp(\Phi)G(\Phi)\end{aligned} \tag{3.9}$$

by introducing the scalar functions

$$\alpha(\theta) = B_\perp\big(\Phi(\theta)\big)M\big(\Phi(\theta)\big)\Phi'(\theta) \tag{3.10a}$$

$$\beta(\theta) = B_\perp\big(\Phi(\theta)\big)\Big[C\big(\Phi(\theta),\Phi'(\theta)\big)\Phi'(\theta) + M\big(\Phi(\theta)\big)\Phi''(\theta)\Big] \tag{3.10b}$$

$$\gamma(\theta) = B_\perp\big(\Phi(\theta)\big)G\big(\Phi(\theta)\big) \tag{3.10c}$$

[4]Arguments to the synchronization functions have been omitted for readability

the equation (3.9) can be written compactly as

$$\alpha(\theta)\ddot{\theta} + \beta(\theta)\dot{\theta}^2 + \gamma(\theta) = 0 \tag{3.11}$$

This equation is referred to as the *zero dynamics* of the system, as it constitutes the time-evolution of the system given that the virtual holonomic constraints are satisfied, i.e. when the errors are zeroed.

This means that the unactuated part of the system cannot be controlled directly with respect to time, but rather evolves according to the zero dynamics (3.11).

3.2 Transverse Linearization

Having the desired trajectory described in the form of the virtual holonomic constraints

$$q_\star = \Phi(\theta_\star) \tag{3.12}$$

allows us to describe the closed periodic orbit of the desired trajectory as [15]

$$\mathcal{O}_\star(q_\star) = \{[q; \dot{q}] \in \mathbb{R}^{2n} : q = q_\star(\tau), \dot{q} = \dot{q}_\star(\tau), \tau \in [0, T]\} \tag{3.13}$$

It is also useful to define the tubular neighborhood of the desired orbit as the set of all points a distance no larger than ε from the desired orbit, i.e.

$$\mathcal{O}_\varepsilon(q_\star) = \{[q; \dot{q}] \in \mathbb{R}^{2n} : \min_{\tau\in[0,T]} \|[q - q_\star(\tau); \dot{q} - \dot{q}_\star(\tau)]\| \leq \varepsilon\} \tag{3.14}$$

A moving Poincaré surface assisiated with the solution $q_\star(t), t \in [0, T]$ is defined as a family of $(2n-1)$-dimensional C^1-smooth surfaces, $\{S(t), t \in [0, T]\}$ satisfying the following criteria [15]

1. The surfaces $S(\cdot)$ are locally disjoint, i.e. $\exists\, \varepsilon > 0$:

$$S(\tau_1) \cap S(\tau_2) \cap \mathcal{O}_\varepsilon(q_\star) = \emptyset$$

2. Each of the surfaces $S(\cdot)$ locally intersects the orbit only in one point, i.e. for each $\tau \in [0, T]$, $\exists\, \varepsilon > 0$:

$$S(\tau) \cap \{[q_\star(t); \dot{q}_\star(t)], |t - \tau| < \varepsilon\} \cap \mathcal{O}_\varepsilon(q_\star) = \{[q_\star(t); \dot{q}_\star(t)]\}$$

3. The surfaces $S(\cdot)$ are smoothly parametrized by time, i.e. $\exists\, f_s \in C^1(\mathbb{R}^n, \mathbb{R}^n, \mathbb{R})$:

$$S(t) \cap \mathcal{O}_\varepsilon(q_\star) = \{[q; \dot{q}] \in \mathbb{R}^{2n} : f_s(q, \dot{q}, t) = 0\} \cap \mathcal{O}_\varepsilon(q_\star)$$

4. The surfaces $S(\cdot)$ are transversal to the orbit i.e. $\forall t \in [0, T]$:

$$\left[\frac{\partial f_s}{\partial q}\bigg|_{\substack{q=q_\star(t)\\ \dot{q}=\dot{q}_\star(t)}} \right]^T \dot{q}_\star(t) + \left[\frac{\partial f_s}{\partial \dot{q}}\bigg|_{\substack{q=q_\star(t)\\ \dot{q}=\dot{q}_\star(t)}} \right]^T \ddot{q}_\star(t) \neq 0$$

As this family of moving Poincaré surfaces is generally difficult to compute, it is convenient to define the family of tangential planes along the vector field of the desired orbit [9]:

$$TS(t) := \left\{ [q(t); \dot{q}(t)] \in \mathbb{R}^{2n} : \begin{bmatrix} q - q_\star(t) \\ \dot{q} - \dot{q}_\star(t) \end{bmatrix}^T \begin{bmatrix} \dot{q}_\star(t) \\ \ddot{q}_\star(t) \end{bmatrix} = 0 \right\} \tag{3.15}$$

The problem of controlling an underactuated system boils down to having the curve traced by the system in phase space, $[q, \dot{q}]$ converge to the orbit of the desired trajectory, $\mathcal{O}_\star$. The state coordinates, $[q, \dot{q}]$ can be locally changed into the coordinates $[\psi, x_\perp]$, where ψ is a scalar variable that parametrizes the position along the orbit of the desired trajectory, while $x_\perp$ is a $(2n-1)$-dimensional vector defining the location on a particular leaf of the moving Poincaré surface parametrized by ψ. The vector $x_\perp$ is known as the *transverse coordinates*, while the dynamics of $x_\perp$ are called *transverse* [15]. As a step in controlling the states such that the trajectory converges to the desired orbit, the transverse coordinates are linearized along the desired solution, $q_\star(t)$. This is called *transverse linearization*. The procedure for computing the transverse linearization is described in the remaining of this section.

3.2.1 Coordinate Transformation

In general the virtual holonomic constraints are not perfectly satisfied and we can introduce n new coordinates as the difference between the generalized coordinates and the virtual holonomic constraints.

$$y_i = q_i - \phi_i(\theta) \quad \text{for } i = 1, \ldots, n \tag{3.16}$$

Together with the monotonic variable θ this constitutes a set of excessive generalized coordinates. As there are now $n+1$ equations describing the motion of n coordinates, the excessive generalized coordinates are not linearly independent, and one may be expressed as a function of the others [14]. Without loss of generality, assume that this is the case for y_n. q_n can then be written as

$$q_n = \phi_n(\theta) + h(y, \theta) \tag{3.17}$$

The new generalized coordinates are then given by

$$y = [y_1, \ldots, y_{n-1}] \in \mathbb{R}^{n-1} \,\text{and}\, \theta \tag{3.18}$$

The previous generalized coordinates, as well as its first and second derivatives, can now be written as functions of (θ, y) and their derivatives.

$$q = \begin{bmatrix} y \\ h(y, \theta) \end{bmatrix} + \Phi(\theta) \tag{3.19a}$$

$$\dot{q} = L(\theta, y) \begin{bmatrix} \dot{y} \\ \dot{\theta} \end{bmatrix} \tag{3.19b}$$

$$\ddot{q} = L(\theta, y) \begin{bmatrix} \ddot{y} \\ \ddot{\theta} \end{bmatrix} + \dot{L}(\theta, y) \begin{bmatrix} \dot{y} \\ \dot{\theta} \end{bmatrix} \tag{3.19c}$$

where

$$L(\theta, y) = \begin{bmatrix} I_{(n-1)} & 0_{(n-1)\times 1} \\ \frac{\partial h}{\partial y} & \frac{\partial h}{\partial \theta} \end{bmatrix} + \begin{bmatrix} & \phi_1'(\theta) \\ 0_{n\times(n-1)} & \vdots \\ & \phi_n'(\theta) \end{bmatrix} \tag{3.20}$$

The zero dynamics described in Section 3.1.1 are only valid when the virtual holonomic constraints are exactly satisfied. This is generally not the case and when the generalized coordinates are described as in (3.19) then some contribution from y, $\dot{y}$ and $\ddot{y}$ show up in (3.11). In this case the dynamics of θ can be written as [15]

$$\alpha(\theta)\ddot{\theta} + \beta(\theta)\dot{\theta}^2 + \gamma(\theta) = g(\theta, \dot{\theta}, \ddot{\theta}, y, \dot{y}, v) \tag{3.21}$$

where $g(\theta, \dot{\theta}, \ddot{\theta}, y, \dot{y}, v)$ is a smooth function that evaluates to zero along the desired solution, i.e.

$$g(\theta_\star, \dot{\theta}_\star, \ddot{\theta}_\star, 0, 0, 0) = 0 \tag{3.22}$$

3.2.2 Partial Feedback Linearization

As the system in question is underactuated it is not feedback linearizable . It is, however, possible to use feedback linearization of parts of the system. To do this a change of coordinates is needed.

Assuming that $M(q)$ is invertible allows (2.3) to be solved for $\ddot{q}$

$$\ddot{q} = M^{-1}(q)\Big(B(q)u - C(q,\dot{q})\dot{q} - G(q)\Big) \tag{3.23}$$

Inserting (3.19c) and assuming that $L(\theta, y)$ is invertible, an expression for $\ddot{y}$ is obtained as follows

$$\begin{aligned}
L(\theta,y)\begin{bmatrix}\ddot{y}\\ \ddot{\theta}\end{bmatrix} + \dot{L}(\theta,y)\begin{bmatrix}\dot{y}\\ \dot{\theta}\end{bmatrix} &= M^{-1}(q)\Big[-C(\dot{q},q)\dot{q} - G(q) + B(q)u\Big] \\
\begin{bmatrix}\ddot{y}\\ \ddot{\theta}\end{bmatrix} &= L^{-1}(\theta,y)\left(M^{-1}(q)\left[-C(\dot{q},q)\dot{q} - G(q) + B(q)u\right] - \dot{L}(\theta,y)\begin{bmatrix}\dot{y}\\ \dot{\theta}\end{bmatrix}\right) \\
\ddot{y} &= \Big[I_{(n-1)}, 0_{(n-1)\times 1}\Big]\, L^{-1}(\theta,y) \\
&\quad \times \left(M^{-1}(q)\left[-C(\dot{q},q)\dot{q} - G(q) + B(q)u\right] - \dot{L}(\theta,y)\begin{bmatrix}\dot{y}\\ \dot{\theta}\end{bmatrix}\right) \\
\ddot{y} &= R(\theta,\dot{\theta},y,\dot{y}) + N(\theta,y)u
\end{aligned} \tag{3.24}$$

where

$$N(\theta,y) = \Big[I_{(n-1)}, 0_{(n-1)\times 1}\Big]\, L^{-1}(\theta,y)M^{-1}(q)B(q) \tag{3.25}$$

$$\begin{aligned}
R(\theta,\dot{\theta},y,\dot{y}) &= \Big[I_{(n-1)}, 0_{(n-1)\times 1}\Big]\, L^{-1}(\theta,y) \\
&\quad \times \left(M^{-1}(q)[-C(\dot{q},q)\dot{q} - G(q)] - \dot{L}(\theta,y)\begin{bmatrix}\dot{y}\\ \dot{\theta}\end{bmatrix}\right)
\end{aligned} \tag{3.26}$$

where q and $\dot{q}$ are defined in (3.19a) and (3.19b)

By defining the control transform from u to v as follows

$$u = v + U(\theta,\dot{\theta},y,\dot{y}) \tag{3.27}$$

where $U(\cdot)$ is a smooth function that coincides with the nominal input $u_\star(t)\,\forall\, t \in [0,T]$, i.e.

$$U(\theta_\star(t),\dot{\theta}_\star(t),0,0) \equiv u_\star(t) \tag{3.28}$$

Then (3.27) can be inserted into (3.24) to yield

$$\begin{aligned} \ddot{y} &= F(\theta,\dot{\theta},y,\dot{y}) + N(\theta,y)v \\ F(\theta,\dot{\theta},y,\dot{y}) &= R(\theta,\dot{\theta},y,\dot{y}) + N(\theta,y)U(\theta,\dot{\theta},y,\dot{y}) \end{aligned} \tag{3.29}$$

If the system in question is underactuated of degree one, in general $N(\theta,y)$ is invertible. By defining the control transformation

$$u = N^{-1}(\theta,y)\left(v - R(\theta,\dot{\theta},y,\dot{y})\right) \tag{3.30}$$

and inserting into (3.24) this reduces to

$$\begin{aligned} \ddot{y} &= R(\theta,\dot{\theta},y,\dot{y}) + N(\theta,y)u \\ &= R(\theta,\dot{\theta},y,\dot{y}) + N(\theta,y)N^{-1}(\theta,y)\left(v - R(\theta,\dot{\theta},y,\dot{y})\right) \\ &= v \end{aligned} \tag{3.31}$$

Under this control transformation the dynamics of the error-states can be described as the linear time-invariant system [5]

$$\begin{bmatrix}\dot{y}\\ \ddot{y}\end{bmatrix} = \begin{bmatrix}0 & I\\ 0 & 0\end{bmatrix}\begin{bmatrix}y\\ y\end{bmatrix} + \begin{bmatrix}0\\ I\end{bmatrix}v \tag{3.32}$$

This system only describes the dynamics of the $2n-2$ states associated with the errors of the virtual holonomic constraints, i.e. the errors of the actuated coordinates. The dynamics of the unactuated coordinate is still non-linear and will be treated later in this text.

3.2.3 Solving the Zero Dynamics

When the system satisfies the virtual holonomic constraints perfectly, it has been shown that the dynamics of the scalar variable, θ can be written as

$$\alpha(\theta)\ddot{\theta} + \beta(\theta)\dot{\theta} + \gamma(\theta) = 0 \tag{3.33}$$

However, as the pair $[\theta;\dot{\theta}]$ is not controllable, it is desirable to rewrite these $[\theta;\dot{\theta}]$ into a controllable and an uncontrollable part, i.e. partitioning it into one scalar variable determining the position along the desired trajectory

[5]This is only the case if the degree of underactuation is one; if the degree of underactuation is greater than one, some non-linear terms would show up.

$[\theta_\star; \dot{\theta}_\star]$ and another scalar variable describing the distance from the desired trajectory. As a step in achieving this, the scalar dynamics (3.33) is solved for $[\theta; \dot{\theta}]$. This is shown in Appendix A.1 and the result is that if a solution from the initial conditions $[\theta_0, \dot{\theta}_0]$ exist, then the function

$$\begin{aligned} I(\theta, \dot{\theta}, \theta_0, \dot{\theta}_0) &= \dot{\theta}^2 - \Psi_h(\theta, \theta_0)\dot{\theta}_0^2 + \Psi_p(\theta, \theta_0) \\ \Psi_h(\theta, \theta_0) &= \exp\left\{-2\int_{\theta_0}^{\theta} \frac{\beta(\tau)}{\alpha(\tau)} d\tau\right\} \\ \Psi_p(\theta, \theta_0) &= \int_{\theta_0}^{\theta} \Psi_h(\theta, s)\frac{2\gamma(s)}{\alpha(s)} ds \end{aligned} \tag{3.34}$$

preserves its value along the solution [13]. Further, $I(\theta, \dot{\theta}, \theta_0, \dot{\theta}_0)$ is identically zero along the solution. The function $I(\theta, \dot{\theta}, \theta_0, \dot{\theta}_0)$ can then be seen as the distance from the measured $[\theta; \dot{\theta}]$ to the desired trajectory $[\theta_\star; \dot{\theta}_\star]$. As the function I arises form integrating the zero dynamics, it will be referred to as the *integral function* henceforth.

Differentiating the integral function I with respect to time along the solution of the system (3.21) yields

$$\frac{\mathrm{d}}{\mathrm{dt}} I(\theta, \dot{\theta}, \theta_0, \dot{\theta}_0) = \frac{2\dot{\theta}}{\alpha(\theta)}\left[g(\theta, \dot{\theta}, \ddot{\theta}, y, \dot{y}, v) - \beta(\theta)I(\theta, \dot{\theta}, \theta_0, \dot{\theta}_0)\right] \tag{3.35}$$

where $g(\theta, \dot{\theta}, \ddot{\theta}, y, \dot{y}, v)$ is the smooth function from (3.21). $g(\theta, \dot{\theta}, \ddot{\theta}, y, \dot{y}, v)$ can be found from.

$$\begin{aligned} g(\theta, \dot{\theta}, \ddot{\theta}, y, \dot{y}, v) = B_\perp(q)\left[M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q)\right] \\ - \alpha(\theta)\ddot{\theta} - \beta(\theta)\dot{\theta}^2 - \gamma(\theta) \end{aligned} \tag{3.36}$$

In order to control the integral function it is first noted that by linearizing (3.35) in the direction transverse to the desired trajectory one obtains

$$\frac{d}{d\tau} I_\bullet = \kappa_1(\tau)I_\bullet + \kappa_2(\tau)Y_{1\bullet} + \kappa_3(\tau)Y_{2\bullet} + p(\tau)V_\bullet \tag{3.37}$$

where the coefficient functions are defined as

$$\kappa_1(\tau) = \frac{2\dot{\theta}_\star(\tau)}{\alpha(\theta_\star(\tau))}\Big(g_I\Big(\theta_\star(\tau), \dot{\theta}_\star(\tau), \ddot{\theta}_\star(\tau), 0, 0\Big) - \beta\Big(\theta_\star(\tau)\Big)\Big) \tag{3.38a}$$

$$\kappa_2(\tau) = \frac{2\dot{\theta}_\star(\tau)}{\alpha(\theta_\star(\tau))}\Big(g_y\Big(\theta_\star(\tau), \dot{\theta}_\star(\tau), \ddot{\theta}_\star(\tau), 0, 0\Big)\Big) \tag{3.38b}$$

$$\kappa_3(\tau) = \frac{2\dot{\theta}_\star(\tau)}{\alpha(\theta_\star(\tau))}\Big(g_{\dot{y}}\Big(\theta_\star(\tau), \dot{\theta}_\star(\tau), \ddot{\theta}_\star(\tau), 0, 0\Big)\Big) \tag{3.38c}$$

$$p(\tau) = \frac{2\dot{\theta}_\star(\tau)}{\alpha(\theta_\star(\tau))}\Big(g_v\Big(\theta_\star(\tau), \dot{\theta}_\star(\tau), \ddot{\theta}_\star(\tau), 0, 0\Big)\Big) \tag{3.38d}$$

the g-functions corresponds to the first-order terms in the Taylor expansion of $g(\theta, \dot{\theta}, \ddot{\theta}, y, \dot{y}, v)$ [9], i.e.

$$\Big[g_I, g_y, g_{\dot{y}}, g_v\Big] = \left[\frac{\dot{\theta}\frac{\partial g}{\partial \dot{\theta}} - \ddot{\theta}\frac{\partial g}{\partial \theta}}{2\Big(\dot{\theta}^2 + \ddot{\theta}^2\Big)}, \frac{\partial g}{\partial y}, \frac{\partial g}{\partial \dot{y}}, \frac{\partial g}{\partial v}\right]_{\substack{\theta=\theta_\star(\tau)\\ \dot{\theta}=\dot{\theta}_\star(\tau)\\ \ddot{\theta}=\ddot{\theta}_\star(\tau)\\ y=\dot{y}=v=0}} \tag{3.39}$$

Combining this with the dynamics of $[y; \dot{y}]$, which are already linearized through the partial feedback linearization, and defining $z := \Big[I_\bullet, Y_{1\bullet}, Y_{2\bullet}\Big]$ the linearized error transversal to the orbit can be written compactly as

$$\frac{d}{d\tau}z = A(\tau)z(\tau) + B(\tau)V_\bullet(\tau) \tag{3.40}$$

where

$$A(\tau) = \begin{bmatrix} \kappa_1(\tau) & \kappa_2(\tau) & \kappa_3(\tau) \\ 0_{(n-1)\times 1} & 0_{(n-1)\times(n-1)} & I_{(n-1)\times(n-1)} \\ 0_{(n-1)\times 1} & 0_{(n-1)\times(n-1)} & 0_{(n-1)\times(n-1)} \end{bmatrix} \tag{3.41a}$$

$$B(\tau) = \begin{bmatrix} p(\tau) \\ 0_{(n-1)\times 1} \\ I_{(n-1)\times 1} \end{bmatrix} \tag{3.41b}$$

This means that designing a controller that asymptotically satisfies the virtual holonomic constraints and also converges to the desired scalar dynamics $[\theta; \dot{\theta}]$ is equivalent to finding a control law $V(\tau)$ that stabilizes the origin of the linear time-varying system (3.40).

3.2.4 Controlling the Transverse Error

The problem of finding a controller that stabilizes the transverse linearized errors reduces to finding a control gain matrix such that the control law

$$V_\bullet = -K(\tau)z, \qquad K(\tau) = K(\tau + t) \tag{3.42}$$

stabilizes the origin of the linear time-varying system (3.40).

In [14] it is further shown that a possible choice of control law that stabilizes the origin of (3.40) is

$$V_\bullet = -\Gamma^{-1}b^T(\tau)R(\tau)z(\tau) \tag{3.43}$$

where $R(\tau)$ is a $(2n-1)\times(2n-1)$ matrix-function satisfying $R(\tau) = R(\tau+T)$, $R(\tau) = R(\tau)^T$ and the Riccati equation

$$\dot{R}(\tau) + A^T(\tau)R(\tau) + R(\tau)A(\tau) + G = R(\tau)b(\tau)\Gamma^{-1}b^T(\tau)R(\tau) \tag{3.44}$$

$\forall\tau \in [0, T]$, successfully stabilizes the origin of (3.40). However, any choice of $K(\tau)$ that stabilizes the origin of (3.40) could be used to stabilize the desired periodic solution. As computing a periodic solution to the Riccati equations is in itself a challenging task (see e.g. [2,12]), in this work a different approach is taken to stabilizing the system (3.40).

3.3 Linear Time-Varying Systems

A linear time-varying (LTV) system can be written as [3] [6]

$$\dot{x}(t) = A(t)x(t) + B(t)u(t), \qquad x(t_0) = x_0 \tag{3.45}$$

The solution, $x(t)$ to this system can be written as

$$x(t) = \Phi(t, t_0)x_0 + \int_{t_0}^{t} \Phi(t, \tau)B(\tau)u(\tau)d\tau \tag{3.46}$$

where the state transition matrix, $\Phi(t, t_0)$ is defined as the unique solution of

$$\frac{\partial}{\partial t}\Phi(t, t_0) = A(t)\Phi(t, t_0) \tag{3.47}$$

with initial condition $\Phi(t_0, t_0) = I$.

[6] Only the case where all states are measured is considered, i.e. y(t)=x(t)

Now, consider the case where the system is subjected to the feedback law

$$u(t) = -K(t)x(t) \tag{3.48}$$

Then the system can be written as

$$\dot{x}(t) = \Big(A(t) - B(t)K(t)\Big)x(t) \tag{3.49}$$

and a new state transition matrix can be defined as the solution of

$$\frac{\partial}{\partial t}\tilde{\Phi}(t,t_0) = \Big(A(t) - B(t)K(t)\Big)\tilde{\Phi}(t,t_0) \tag{3.50}$$

and the solution of (3.49) can be written as

$$x(t) = \tilde{\Phi}(t,t_0)x_0$$

Using the definition of the induced p-norm of a matrix A (from e.g. [8])

$$\|A\|_p = \sup_{x\neq 0}\frac{\|Ax\|_p}{\|x\|_p} \tag{3.51}$$

an upper bound on the p-norm of the state vector $x(t)$ is found to be

$$\|x(t)\| = \|\Phi(t,t_0)x_0\| \leq \|\Phi(t,t_0)\|\|x_0\| \tag{3.52}$$

Using this, (3.45) can be shown to be marginally stable if $\forall\, t$ and t_0 with $t \geq t_0$ there exist a finite constant M such that [3]

$$\|\Phi(t,t_0)\| \leq M < \infty. \tag{3.53}$$

If, in addition

$$\|\Phi(t,t_0)\| \to 0 \qquad \text{as } t \to \infty \tag{3.54}$$

then the system (3.45) is asymptotically stable.

A more strict stability condition can also be found; it can be shown that the system (3.45) is asymptotic exponentially stable if [8]

$$\|\Phi(t,t_0)\| \leq ke^{-\lambda(t-t_0)}, \qquad \forall\, t \geq t_0 \geq 0 \tag{3.55}$$

for some positive constants k and λ.

This will be utilized when searching for a controller to stabilize the transverse dynamics.

Chapter 4

Steps in Orbital Stabilization

4.1 Illustration of Concepts on the Inverted Pendulum

This paper aims to stabilize a pre-planned periodic brachiating trajectory in a 24-degrees-of-freedom humanoid robot. The model is greatly simplified, and only the 6 degrees of freedom believed to be the most important for a brachiating motion are modeled. However, the system is still fairly complex and the system matrices are too large for any calculations to be verified by hand. For this reason the inverted pendulum system is introduced. Both in order to allow for verification of the concepts described in this text, and to serve as a reference when discussing the various concepts.

4.1.1 Modeling

The inverted pendulum consists of a point-mass attached to a rod which is again attached to a moving cart through a revolute joint. The cart is assumed to be able to move frictionless in the horizontal direction and is driven by a force, $\vec{F}$ applied in the horizontal direction. The cart-pendulum system is depicted in Figure 4.1, where M is the mass of the cart, m is the mass of the point-mass at the end of the rod and L is the length of the rod. For simplicity, we assume that $L = 1$ in the following.

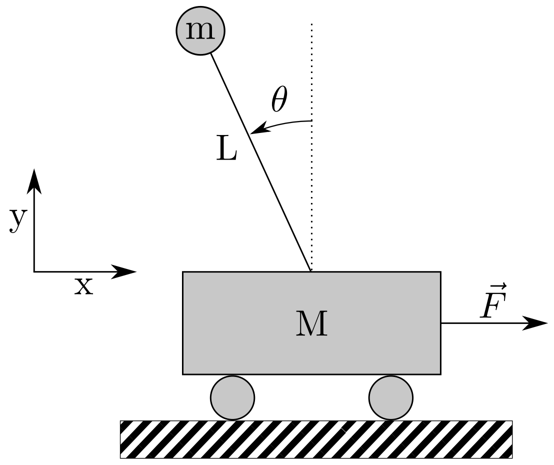


Figure 4.1: Inverted pendulum

For this system the generalized coordinates are chosen as

$$q = \begin{bmatrix} x \\ \theta \end{bmatrix} \tag{4.1}$$

The cartesian coordinates of the two masses are given by

$$\begin{aligned} p_m &= \begin{bmatrix} x - \sin(\theta) \\ \cos(\theta) \end{bmatrix} \\ p_M &= \begin{bmatrix} x \\ 0 \end{bmatrix} \end{aligned} \tag{4.2}$$

and the velocities

$$\begin{aligned} v_m &= \begin{bmatrix} \dot{x} - \cos(\theta)\dot{\theta} \\ -\sin(\theta)\dot{\theta} \end{bmatrix} \\ v_M &= \begin{bmatrix} \dot{x} \\ 0 \end{bmatrix} \end{aligned} \tag{4.3}$$

The kinetic energy is then given by

$$\begin{aligned}
T &= \frac{1}{2}\Big(m v_m^T v_m + M v_M^T v_M\Big) \\
&= \frac{1}{2}\Big(m\dot{x}^2 - 2m\dot{x}\dot{\theta}\cos(\theta) + (\cos^2(\theta) + \sin^2(\theta))\dot{\theta}^2 + M\dot{x}^2\Big) \\
&= \frac{1}{2}(m+M)\dot{x}^2 - m\dot{x}\dot{\theta}\cos(\theta) + \frac{1}{2}m^2\dot{\theta}^2 \qquad (4.4)
\end{aligned}$$

The potential energy of the system can be seen to be

$$V = mg\cos(\theta) \qquad (4.5)$$

The Lagrangian is then

$$\begin{aligned}
\mathcal{L} &= T - V \\
&= \frac{1}{2}(m+M)\dot{x}^2 - m\dot{x}\dot{\theta}\cos(\theta) + \frac{1}{2}m^2\dot{\theta}^2 - mg\cos(\theta) \qquad (4.6)
\end{aligned}$$

The two equations of motion can now be found from

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial\mathcal{L}}{\partial\dot{x}}\right) - \frac{\partial\mathcal{L}}{\partial x} &= F \\
\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial\mathcal{L}}{\partial\dot{\theta}}\right) - \frac{\partial\mathcal{L}}{\partial \theta} &= 0 \qquad (4.7)
\end{aligned}$$

Inserting (4.6) leads to

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial\mathcal{L}}{\partial\dot{x}}\right) &= \frac{\mathrm{d}}{\mathrm{dt}}\Big((m+M)\dot{x} - m\dot{\theta}\cos(\theta)\Big) \\
&= (m+M)\ddot{x} - m\ddot{\theta}\cos(\theta) + m\dot{\theta}^2\sin(\theta) \\
\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial\mathcal{L}}{\partial\dot{\theta}}\right) &= \frac{\mathrm{d}}{\mathrm{dt}}\Big(-m\dot{x}\cos(\theta) + m\dot{\theta}\Big) \\
&= -\cos(\theta)\ddot{x} + L\ddot{\theta} - g\sin(\theta) \\
\frac{\partial\mathcal{L}}{\partial x} &= 0 \\
\frac{\partial\mathcal{L}}{\partial \theta} &= m\dot{x}\dot{\theta}\sin(\theta) + mg\sin(\theta) \qquad (4.8)
\end{aligned}$$

By rewriting this to matrix form, the system in matrix form is obtained

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = B(q)F \qquad (4.9)$$

where

$$M(q) = \begin{bmatrix} M+m & -m\cos\theta \\ -\cos\theta & 1 \end{bmatrix} \tag{4.10a}$$

$$C(q,\dot{q}) = \begin{bmatrix} 0 & m\dot{\theta}\sin\theta \\ 0 & 0 \end{bmatrix} \tag{4.10b}$$

$$G(q) = \begin{bmatrix} 0 \\ -g\sin\theta \end{bmatrix} \tag{4.10c}$$

$$B(q) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{4.10d}$$

4.1.2 Trajectory Planning

The inverted pendulum system has one unstable equilibrium at the upright position ($\theta = 0$). Say we would like a periodic solution such that a point a distance L along the rod always moves in a vertical line, i.e. we have the virtual holonomic constraint

$$x = a + L\sin\theta \tag{4.11}$$

where a is some point along the x-axis. The syncronization functions $\Phi(\theta)$ can then be written as

$$\Phi(\theta) = \begin{bmatrix} a + L\sin\theta \\ \theta \end{bmatrix} \tag{4.12}$$

The first and second derivatives of the synchronization functions are computed as

$$\Phi'(\theta) = \begin{bmatrix} L\cos\theta \\ 1 \end{bmatrix}$$
$$\Phi''(\theta) = \begin{bmatrix} -L\sin\theta \\ 0 \end{bmatrix}$$

The desired trajectory can now be written as

$$q = \Phi(\theta) \tag{4.13a}$$

$$\dot{q} = \Phi'(\theta)\dot{\theta} \tag{4.13b}$$

$$\ddot{q} = \Phi''(\theta)\dot{\theta}^2 + \Phi'(\theta)\ddot{\theta} \tag{4.13c}$$

Inserting this into (4.9) and premultiplying by $B_\perp = [0, 1]$, the zero dynamics in the form given in (3.11) is obtained

$$\alpha(\theta)\ddot{\theta} + \beta(\theta)\dot{\theta}^2 + \gamma(\theta) = 0 \tag{4.14}$$

where $\alpha(\theta)$, $\beta(\theta)$ and $\gamma(\theta)$ are computed according to (3.10)

$$\begin{aligned}
\alpha(\theta) &= B_\perp M\Big(\Phi(\theta)\Big)\Phi'(\theta) \\
&= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} M+m & -m\cos\theta \\ -\cos\theta & 1 \end{bmatrix} \begin{bmatrix} L\cos\theta \\ 1 \end{bmatrix} \\
&= 1 - L\cos^2(\theta) && \text{(4.15a)} \\
\beta(\theta) &= B_\perp\Big[C\Big(\Phi(\theta), \Phi'(\theta)\Big)\Phi'(\theta) + M\Big(\Phi(\theta)\Big)\Phi''(\theta)\Big] \\
&= \begin{bmatrix} 0 & 1 \end{bmatrix} \left(\begin{bmatrix} 0 & m\dot{\theta}\sin\theta \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -L\sin\theta \\ 0 \end{bmatrix} + \begin{bmatrix} M+m & -m\cos\theta \\ -\cos\theta & 1 \end{bmatrix} \begin{bmatrix} -L\sin\theta \\ 0 \end{bmatrix} \right) \\
&= L\sin(\theta)\cos(\theta) && \text{(4.15b)} \\
\gamma(\theta) &= B_\perp G\Big(\Phi(\theta)\Big) \\
&= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ -g\sin\theta \end{bmatrix} \\
&= -g\sin\theta && \text{(4.15c)}
\end{aligned}$$

This means that if the virtual holonomic constraints are perfectly satisfied, i.e. (4.13) holds, then the time-evolution of $\theta(t)$ and $\dot{\theta}(t)$ are uniquely defined given the initial conditions $\theta(0) = \theta_0$ and $\dot{\theta}(0) = \dot{\theta}_0$. This can be used to define the desired trajectory $\theta_\star(t)$. Say we would like to stabilize the periodic orbit in which the pendulum swings periodic between $\theta_{min} = -0.1$ and $\theta_{max} = 0.1$. This trajectory can then be defined by providing the synchronization functions $\Phi(\theta)$ and initial conditions for (4.14). In this case initial conditions that lead to the desired motion is $\theta_\star(0) = \theta_{min} = -0.1$ and $\dot{\theta}_\star(0) = 0$, and the desired trajectory is given as the solution of (4.14) given these initial conditions.

4.1.3 Feedback Linearization

The error for the x-coordinate can be written as

$$y = x - a - L\sin\theta.$$

Hence, the generalized coordinates, q, can now be written as the sum of their desired value, $\Phi(\theta)$ and the error[1].

$$q = \begin{bmatrix} x \\ \theta \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix} + \underbrace{\begin{bmatrix} a + L\sin\theta \\ \theta \end{bmatrix}}_{\Phi(\theta)}. \tag{4.16}$$

and it's first and second derivatives

$$\begin{aligned}
\dot{q} &= \underbrace{\begin{bmatrix} 1 & L\cos\theta \\ 0 & 1 \end{bmatrix}}_{L(\theta,y)} \begin{bmatrix} \dot{y} \\ \dot{\theta} \end{bmatrix} \\
&= \Phi'(\theta)\dot{\theta} + \begin{bmatrix} \dot{y} \\ 0 \end{bmatrix} \\
\ddot{q} &= \underbrace{\begin{bmatrix} 1 & L\cos\theta \\ 0 & 1 \end{bmatrix}}_{L(\theta,y)} \begin{bmatrix} \ddot{y} \\ \ddot{\theta} \end{bmatrix} + \underbrace{\begin{bmatrix} -L\dot{\theta}^2\sin\theta \\ 0 \end{bmatrix}}_{N(\theta,\dot{\theta},y,\dot{y})} \\
&= \Phi'(\theta)\ddot{\theta} + \Phi''(\theta)\dot{\theta}^2 + \begin{bmatrix} \ddot{y} \\ 0 \end{bmatrix}. \qquad (4.17)
\end{aligned}$$

Using the theory from Section 3.2.2 we then have that a control law of the form

$$u = N^{-1}(\theta, y)\left(v - R(\theta, \dot{\theta}, y, \dot{y})\right) \tag{4.18}$$

where $N(\theta, y)$ and $R(\theta, \dot{\theta}, y, \dot{y})$ are computed as follows

$$\begin{aligned}
N(\theta, y) &= \left[I_{(n-1)}, 0_{(n-1)\times 1}\right] L^{-1}(\theta, y) M^{-1}(q) B(q) && (4.19)\\
&= \left[I_{(n-1)}, 0_{(n-1)\times 1}\right] L^{-1}(\theta, y) M^{-1}(q) B(q) && (4.20)\\
R(\theta, \dot{\theta}, y, \dot{y}) &= \left[I_{(n-1)}, 0_{(n-1)\times 1}\right] L^{-1}(\theta, y) \\
&\quad \times \left(M^{-1}(q)[-C(\dot{q}, q)\dot{q} - G(q)] - \dot{L}(\theta, y) \begin{bmatrix} \dot{y} \\ \dot{\theta} \end{bmatrix} \right) && (4.21)
\end{aligned}$$

will cancel the non-linear terms of the y-dynamics, such that we can write

$$\ddot{y} = v. \tag{4.22}$$

The matrix functions $N(\theta, y)$ and $R(\theta, \dot{\theta}, y, \dot{y})$ are implemented as functions calling the appropriate system matrix functions, and the explicit forms of these matrices are therefore omitted.

[1]The error in the θ-coordinate is identically zero as θ is the quantity that parametrizes the motion.

4.1.4 Scalar dynamics

Premultiplying the equations of motion by $B^\perp$ and inserting the above relations yields

$$
\begin{aligned}
&B^\perp(q)\left[M(q)\left(\Phi'(\theta)\ddot{\theta}+\Phi''(\theta)\dot{\theta}^2+\begin{bmatrix}v\\0\end{bmatrix}\right)+C(q,\dot{q})\left(\Phi(\theta)\dot{\theta}+\begin{bmatrix}\dot{y}\\0\end{bmatrix}\right)+G(q)\right]=0\\
&B^\perp(q)M(q)\Phi'(\theta)\ddot{\theta}+B^\perp(q)\left(M(q)\Phi''(\theta)+C(q,\dot{q})\Phi(\theta)\right)\dot{\theta}^2+G(q)\\
&\qquad=-B^\perp(q)M(q)\begin{bmatrix}v\\0\end{bmatrix}-B^\perp(q)C(q,\dot{q})\begin{bmatrix}\dot{y}\\0\end{bmatrix}
\end{aligned}
\tag{4.23}
$$

Noting that the system matrices are in fact only functions of θ, we can write

$$
\begin{aligned}
M(q)&=M(\Phi(\theta))\\
C(q,\dot{q})&=C(\Phi(\theta),\Phi'(\theta))\dot{\theta}\\
G(q)&=G(\Phi(\theta))
\end{aligned}
$$

Hence, (4.23) can be written as

$$
\alpha(\theta)\ddot{\theta}+\beta(\theta)\dot{\theta}^2+\gamma(\theta)=g(\theta,\dot{\theta},\ddot{\theta},y,\dot{y},v) \tag{4.24}
$$

where α, β and γ are given in (4.15), and g is the right hand side of the equation (4.23), i.e.

$$
\begin{aligned}
g(\theta,\dot{\theta},\ddot{\theta},y,\dot{y},v)&=-B^\perp(q)\left(M(q)\begin{bmatrix}v\\0\end{bmatrix}+C(q,\dot{q})\begin{bmatrix}\dot{y}\\0\end{bmatrix}\right)\\
&=-\begin{bmatrix}0&1\end{bmatrix}\left(\begin{bmatrix}M+m&-mL\cos\theta\\-\cos\theta&L\end{bmatrix}\begin{bmatrix}v\\0\end{bmatrix}+\begin{bmatrix}0&mL\dot{\theta}\sin\theta\\0&0\end{bmatrix}\begin{bmatrix}\dot{y}\\0\end{bmatrix}\right)\\
&=v\cos(\theta)
\end{aligned}
\tag{4.25}
$$

The partial derivatives of g are

$$
\begin{aligned}
\frac{\partial g}{\partial\theta}&=-v\sin(\theta)\\
\frac{\partial g}{\partial v}&=\cos(\theta)\\
\frac{\partial g}{\partial\dot{\theta}}&=\frac{\partial g}{\partial\ddot{\theta}}=\frac{\partial g}{\partial y}=\frac{\partial g}{\partial\dot{y}}=0.
\end{aligned}
$$

From (3.40) we have that the linearized transverse dynamics are given by

$$
\frac{d}{d\tau}z=A(\tau)z(\tau)+B(\tau)V_\bullet(\tau)
$$

where

$$A(\tau) = \begin{bmatrix} \kappa_1(\tau) & \kappa_2(\tau) & \kappa_3(\tau) \\ 0_{(n-1)\times 1} & 0_{(n-1)\times(n-1)} & I_{(n-1)\times(n-1)} \\ 0_{(n-1)\times 1} & 0_{(n-1)\times(n-1)} & 0_{(n-1)\times(n-1)} \end{bmatrix}$$

$$B(\tau) = \begin{bmatrix} p(\tau) \\ 0_{(n-1)\times 1} \\ I_{(n-1)\times 1} \end{bmatrix}$$

and $\kappa_1(\tau)$, $\kappa_2(\tau)$, $\kappa_3(\tau)$ and $p(\tau)$ are defined in (3.38). Inserting the partial derivatives of g into (3.39) we obtain

$$\begin{aligned} \Big[g_I, g_y, g_{\dot y}, g_v\Big] &= \left[\frac{\dot\theta\frac{\partial g}{\partial\dot\theta} - \ddot\theta\frac{\partial g}{\partial\theta}}{2\big(\dot\theta^2+\ddot\theta^2\big)}, \frac{\partial g}{\partial y}, \frac{\partial g}{\partial \dot y}, \frac{\partial g}{\partial v}\right]_{\substack{\theta=\theta_\star(\tau)\\ \dot\theta=\dot\theta_\star(\tau)\\ \ddot\theta=\ddot\theta_\star(\tau)\\ y=\dot y=v=0}} \\ &= \Big[0, 0, 0, \cos\left(\theta_\star(\tau)\right)\Big] \end{aligned}$$

inserting this into the equations (3.38), along with the explicit expressions for $\alpha(\theta)$ and $\beta(\theta)$ from (4.15) we obtain explicit expressions for the elements of the time-varying system matrices $A(\tau)$ and $B(\tau)$.

$$\begin{aligned} \kappa_1(\tau) &= \frac{2\dot\theta_\star(\tau)}{\alpha(\theta_\star(\tau))}\Big(g_I\Big(\theta_\star(\tau), \dot\theta_\star(\tau), \ddot\theta_\star(\tau), 0, 0\Big) - \beta\Big(\theta_\star(\tau)\Big)\Big) \\ &= \frac{-2L\dot\theta_\star(\tau)\sin\left(\theta_\star(\tau)\right)\cos\left(\theta_\star(\tau)\right)}{1 - L\cos^2\left(\theta_\star(\tau)\right)} \\ \kappa_2(\tau) &= \frac{2\dot\theta_\star(\tau)}{\alpha(\theta_\star(\tau))} g_y\Big(\theta_\star(\tau), \dot\theta_\star(\tau), \ddot\theta_\star(\tau), 0, 0\Big) \\ &= 0 \\ \kappa_3(\tau) &= \frac{2\dot\theta_\star(\tau)}{\alpha(\theta_\star(\tau))} g_{\dot y}\Big(\theta_\star(\tau), \dot\theta_\star(\tau), \ddot\theta_\star(\tau), 0, 0\Big) \\ &= 0 \\ p(\tau) &= \frac{2\dot\theta_\star(\tau)}{\alpha(\theta_\star(\tau))} g_v\Big(\theta_\star(\tau), \dot\theta_\star(\tau), \ddot\theta_\star(\tau), 0, 0\Big) \\ &= \frac{2\dot\theta_\star(\tau)\cos\left(\theta_\star(\tau)\right)}{1 - L\cos^2\left(\theta_\star(\tau)\right)} \end{aligned}$$

This constitutes the linearized transverse dynamics, and a controller for this system can now be found. The search for a stabilizing controller is performed by the MATLAB toolbox described shortly and details of this is omitted from this section.

4.2 Implementation

A controller based on the previously discussed concepts has been designed and implemented using a combination of MATLAB and Simulink. The Simulink diagram of the complete feedback-system is depicted in Figure 4.2 and consists of the following two main components

System is a module taking the control variable u as input, and providing measurements of the states, $[q; \dot{q}]$ as outputs. In this thesis the system block is simply a simulation of the system dynamics, based on the derived equations of motion. Rewriting the equations of motion, the double derivative of the coordinates, q, can be written as

$$\ddot{q} = M^{-1}(q)\Big(B(q)u - C(q,\dot{q})\dot{q} - G(q)\Big) \tag{4.26}$$

The system block then integrates this twice to obtain the state vector $[q; \dot{q}]$. When implemented on a physical robot, the system module would be replaced by some I/O-module communicating with the robot.

Control implements the controller, i.e. computes the desired control input u, given some measurement of the states $[q; \dot{q}]$. This module is further divided into submodules implementing the various parts of the control problem. A more thorough description of how this is done will follow in the remaining of this chapter.

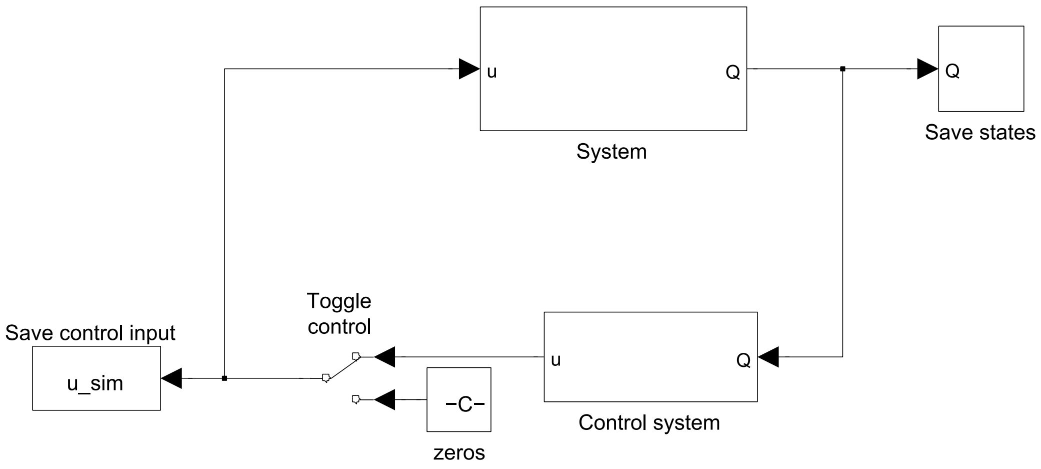


Figure 4.2: Simulink model of the system

The controller module is shown in Figure 4.3 and consists of the four submodules

Transverse Error Calculation calculates the error in the transverse direction, $x_\perp$.

Indexing the Moving Poincaré Section computes the scalar variable ψ that indexes the distance along the desired orbit.

Controlling the Transverse Error computes the auxilary control input, v.

Partial Feedback Linearization computes the full control input based on partial feedback linearization and the auxilary control input.

Each of these modules will be described in more detail in the following sections.

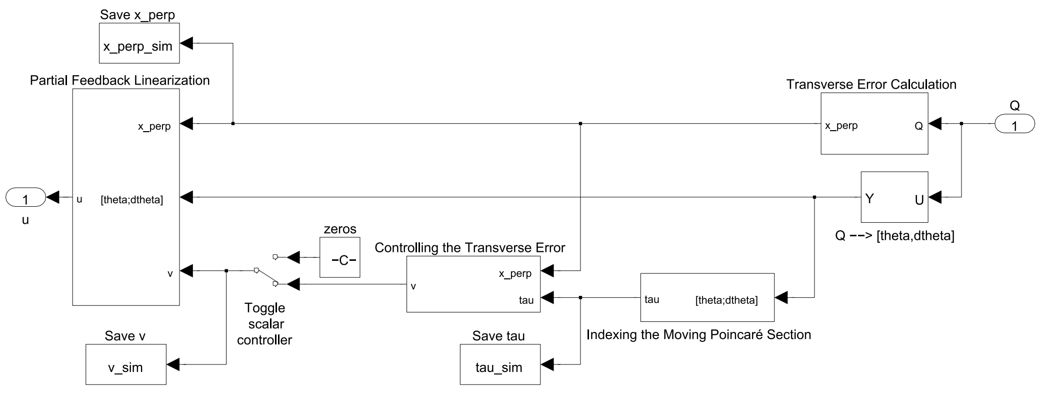


Figure 4.3: Simulink model of the control system

4.2.1 Transverse Error Calculation

The transverse error can be written as

$$x_\perp = \begin{bmatrix} I \\ y \\ \dot{y} \end{bmatrix} \tag{4.27}$$

where I is the integral function, and y and $\dot{y}$ are the errors in the actuated coordinates and their derivatives, respectively. These can be computed according to

$$\begin{aligned} I &= \dot{\theta}^2 - \Psi_h(\theta, \theta_0)\dot{\theta}_0^2 + \Psi_p(\theta, \theta_0) \\ y &= \begin{bmatrix} I & 0 \end{bmatrix} \left(q - \Phi(\theta) \right) \\ \dot{y} &= \begin{bmatrix} I & 0 \end{bmatrix} \left(\dot{q} - \Phi'(\theta)\dot{\theta} \right) \end{aligned}$$

the y and $\dot{y}$ variables are easily computed on-line. However, computing $I(\theta, \dot{\theta}, \theta_0, \dot{\theta}_0)$ requires solving a double-integral where the integration variable of the outer integral is the lower limit of the inner integral. This becomes evident when studying (3.34).

$$\begin{aligned} I(\theta, \dot{\theta}, \theta_0, \dot{\theta}_0) &= \dot{\theta}^2 - \Psi_h(\theta, \theta_0)\dot{\theta}_0^2 + \Psi_p(\theta, \theta_0) \\ \Psi_h(\theta, \theta_0) &= \exp\left\{ -2 \int_{\theta_0}^{\theta} \frac{\beta(\tau)}{\alpha(\tau)} d\tau \right\} \\ \Psi_p(\theta, \theta_0) &= \int_{\theta_0}^{\theta} \Psi_h(\theta, s) \frac{2\gamma(s)}{\alpha(s)} ds \end{aligned} \tag{4.28}$$

Solving this on-line is therefore hard to do. However, as I can be written as

$$I(\theta, \dot{\theta}) = \dot{\theta}^2 + \Psi(\theta) \tag{4.29}$$

where

$$\Psi(\theta) = \Psi_h(\theta)\dot{\theta}_0^2 - \Psi_p(\theta) \tag{4.30}$$

and the desired trajectory $[\theta_\star; \dot{\theta}_\star]$ is known offline from solving

$$\alpha(\theta_\star)\ddot{\theta}_\star + \beta(\theta_\star)\dot{\theta}_\star^2 + \gamma(\theta_\star) = 0$$

from initial conditions $\theta_\star(0) = \theta_{\star 0}$, $\dot{\theta}_\star(0) = \dot{\theta}_{\star 0}$, it is possible to compute the double integral off-line for a range of values for θ that slightly exceeds that of $\theta_\star$, say

$$\theta \in [\theta_\star(0) - \delta_1,\, \theta_\star(T) + \delta_2] \tag{4.31}$$

where δ_1 and δ_2 are some positive constants. A simple look-up-table can then be used to obtain the value for the function $\Psi(\theta)$ given a specific measurement of θ. The remainder of $I(\theta, \dot{\theta})$ is then easily computed on-line by squaring the measured $\dot{\theta}$ and adding the value of $\Psi(\theta)$ obtained from the look-up-table. The Simulink implementation of this is shown in Figure 4.4.

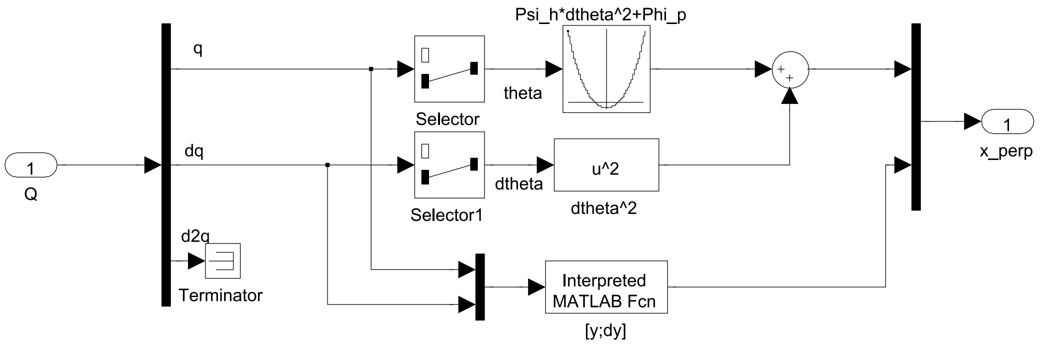


Figure 4.4: Simulink model of the error calculation module

4.2.2 Indexing the Moving Poincaré Section

From e.g. [15] we have that if a control law, of the form

$$V_\bullet = K(\tau)\left[I_\bullet; Y_{1\bullet}; Y_{2\bullet}\right], \qquad K(\tau) = K(\tau + t) \tag{4.32}$$

stabilizes the system of linearized transverse errors, (3.40), then there exist a control law, $v(t)$ of the form

$$v(t) = K(\tau)x_\perp(t) \tag{4.33}$$

that renders the T-periodic solution of the system expononentially orbitally stable. This problem can thus be split into two parts:

1. Finding a matrix of control gains $K(\tau)$ such that $V_\bullet$ stabilizes the transverse linearized dynamics, (3.40).
2. Finding τ such that $[q(t); \dot{q}(t)] \in S(\tau) \cap \mathcal{O}_\varepsilon(q_\star)$ where $\mathcal{O}_\varepsilon$ is defined in (3.14).

Searching for control gains $K(\tau)$ that stabilizes the transverse linearized errors can be done off-line. The method that were used to find this K in this work will be described shortly. However, finding $\tau = \{s : [q(t); \dot{q}(t)] \in S(s) \cap \mathcal{O}_\varepsilon(q_\star)\}$ must be done on-line, and is in general a difficult task as it involves a non-linear optimization problem [9]. However, this problem can be simplified by utilizing the family of planes that are tangential to the flow of a solution $[q; \dot{q}]$.

$$TS(t) = \left\{[q(t); \dot{q}(t)] \in \mathbb{R}^{2n} : \begin{bmatrix} q - q_\star(t) \\ \dot{q} - \dot{q}_\star(t) \end{bmatrix}^T \begin{bmatrix} \dot{q}_\star(t) \\ \ddot{q}_\star(t) \end{bmatrix} = 0\right\} \tag{4.34}$$

such that τ can be written as

$$\tau = \{s : [q(t); \dot{q}(t)] \in TS(s) \cap \mathcal{O}_\varepsilon(q_\star)\} \tag{4.35}$$

Finding this is considerably easier, as it no longer involves explicit knowing the generally Poincaré surface $S(t)$.

Further, if $\theta = \theta_\star(t)$ is a monotonic function, then the index $\tau = \psi(\theta, \dot{\theta})$ can be found as the inverse of $\psi(\theta)$ for $\theta_\star(t)$, such that $\psi(\theta_\star(t)) = t$. This is the approach taken when considering both the gorilla system and the inverted pendulum system. In the gorilla case, $\theta = \theta_\star(t)$ is indeed a monotonic function, and τ can then be found simply as $\tau = \{s : \theta_\star(s) = \theta\}$. In the case of the inverted pendulum, $\theta = \theta_\star(t)$ is not a monotonic function. However, a function that is monotonic over a period is the two-argument arctan-function (atan2 in MATLAB) $\psi(\theta, \dot{\theta}) = \arctan 2(\dot{\theta}, \theta)$. This can therefore be used to find τ in the inverted pendulum case.

4.2.3 Controlling the Transverse Error

This module aims to find a matrix of T-periodic gains, $K(\tau)$, that stabilizes the transverse linearized dynamics, (3.40). As mentioned in Chapter 3, one way of doing this is to compute the solution to the periodic Riccati equation

$$\dot{R}(\tau) + A^T(\tau)R(\tau) + R(\tau)A(\tau) + G = R(\tau)b(\tau)\Gamma^{-1}b^T(\tau)R(\tau) \tag{4.36}$$

and then choose the matrix of control gains as

$$K(\tau) = -\Gamma^{-1}b^T(\tau)R(\tau) \tag{4.37}$$

In this work, a different approach is taken. As any periodic time-varying matrix function of control gains $K(\tau)$ that stabilizes the transverse linearized error dynamics, we simply search for matrix functions doing this. To do this, we first reformulate the problem as an optimization problem. This can be done by first parametrizing the control gains by some vector of optimization coefficients $c = [c_1, c_2, \dots, c_p]$ where p is the number of optimization coefficients. The gain matrix function can now be written as

$$K = K(\tau, c) \tag{4.38}$$

Next, a cost function f depending on the optimization coefficients must be defined. From the discussion on stability of linear time-varying systems in

Section 3.3 we observe that the criterion for marginal stability, asymptotic stability and asymptotic exponential stability, all depend on the norm of the state transition matrix $\Phi(t, t_0)$. For this reason we propose a cost function that also depend on the norm of $\Phi(t, t_0)$, namely the cost function

$$f(c) = \int_{t_0}^{T} w(\tau)\|\Phi(\tau, t_0)\| d\tau \tag{4.39}$$

where $w(t)$ is some window function and $\Phi(\tau, t_0)$ is the state transition matrix for the closed-loop linear time-varying system (3.40)

$$\frac{d}{d\tau} z = \Big(A(\tau) - B(\tau)K(\tau, c)\Big) z(\tau) \tag{4.40}$$

where $A(\tau)$ and $B(\tau)$ are defined in (3.41).

We note that this choice of cost function is highly configurable due to the presence of the weight function $w(t)$. Placing high weights on values on $\|\Phi(t, t_0)\|$ when t is close to T, will cause the optimization to favor control gains that lead to aggressive behavior, i.e. the control gains will allow large errors initially in order to achieve a slightly smaller value for $\|\Psi(t, t_0)\|$ towards the end of the period. Equivalently, placing high weights on values when t is close to 0 will cause large errors to be possible at the end of the period. The weight function should favor the norm towards the end of the period, but without allowing too large errors at the start. A weight function that lead to good results in simulations is

$$w(\tau) = \frac{e^{\tau} - e^{t_0}}{e^{T} - e^{t_0}} \tag{4.41}$$

This is an exponentially increasing weight function, starting at 0 and ending up at 1 the end of the period.

4.2.4 Partial Feedback Linearization

The partial feedback linearization module is a straightforward implementation of the concepts described in Section 3.2.2. The output of the module, u is computed according to (3.30), i.e.

$$u = N^{-1}(\theta, y)\Big(v - R(\theta, \dot{\theta}, y, \dot{y})\Big) \tag{4.42}$$

where

$$N(\theta, y) = \left[I_{(n-1)}, 0_{(n-1)\times 1}\right] L^{-1}(\theta, y) M^{-1}(q) B(q) \tag{4.43}$$

$$R(\theta, \dot{\theta}, y, \dot{y}) = \left[I_{(n-1)}, 0_{(n-1)\times 1}\right] L^{-1}(\theta, y) \times \left(M^{-1}(q)[-C(\dot{q}, q)\dot{q} - G(q)] - \dot{L}(\theta, y) \begin{bmatrix} \dot{y} \\ \dot{\theta} \end{bmatrix} \right)$$

$$L(\theta, y) = \begin{bmatrix} I_{(n-1)} & 0_{(n-1)\times 1} \\ 0_{1\times(n-1)} & 0 \end{bmatrix} + \begin{bmatrix} 0_{n\times(n-1)} & \Phi'(\theta) \end{bmatrix} \tag{4.44}$$

As the system matrix functions $M(q)$, $C(q, \dot{q})$, $G(q)$ and $B(q)$, as well as the synchronization functions, $\Phi(\theta)$ are all known, this control law can be implemented directly.

4.3 Dynamics of the Monkey Robot

The proposed brachiating motion consists of two distinct phases with distinct dynamical models. During the single-support phase an underactuated 6-degrees-of-freedom model is used. This model is underactuated due to the lack of torque about the handhold. During double-support the robot becomes fully actuated as it is now able to move to any desired configuration as long as the required amount of torques are available. If this is not the case, some dynamic constraints would show up. However, in the following it is assumed that the preplanned desired trajectory stays well within the bounds of available torques such that the assumption that enough torque is available is valid. The following sections presents briefly how to compute the appropriate models for single-support and double-support phases.

4.3.1 Single-support Phase

During single support, as mentioned, the system is naturally underactuated due to the lack of torque about the handhold. To arrive at a model for this phase, the kinetic energy for each link in the robot is considered. The procedure can be sumarized as follows

1. For each of the links constituting the robot, compute the Jacobian relating the velocities in the local link frame to the velocities in the inertial frame, i.e. compute $J_{\omega_i}(q)$ and J_{v_i} such that the rotational and

linear velocities in the inertia frame, ω_I and v_i, can be expressed as

$$\begin{aligned} \omega_i &= J_{\omega_i}(q)\dot{q} \\ v_i &= J_{v_i}\dot{q} \end{aligned} \tag{4.45}$$

where q is the vector of generalized coordinates and $\dot{q}$ contains the time-derivatives of the generalized coordinates.

2. Find the total kinetic and potential energy of the system by summing the contributions from each link. This can be done using [16]

$$\begin{aligned} K &= \frac{1}{2}\sum_{i=1}^{n}\left(m_i(J_{v_i}(q)\dot{q})^T J_{v_i}(q)\dot{q} + J_{\omega_i}(q)\dot{q})^T I_i J_{\omega_i}(q)\dot{q}\right) \\ &= \frac{1}{2}\dot{q}^T M(q)\dot{q} \end{aligned}$$

and

$$P = \sum_{i=1}^{n} P_i = \sum_{i=1}^{n} g^T r_i m_i$$

where m is the mass I is the inertia tensor, g is the gravity vector, r is the vector from the origin of the inertia frame to the center of mass expressed in the inertia frame. The subscript i denotes the link.

3. Compute the system matrix functions as follows [16]. $M(q)$ can be read directly from the expression for kinetic energy:

$$M(q) = \sum_{i=1}^{n}\left(m_i J_{v_i}^T(q) J_{v_i}(q) + (J_{\omega_i}^T(q) I_i J_{\omega_i}(q)\right)$$

$G(q)$ is a vector function whos k-th element is given by

$$G_k(q) = \frac{\partial P(q)}{\partial q_k}$$

and $C(q, \dot{q})$ is a matrix function where the (j, k)-th element is defined as

$$c_{kj} = \sum_{i=1}^{n} \frac{1}{2}\left(\frac{\partial M_{kj}}{\partial q_i} + \frac{\partial M_{ki}}{\partial q_j} - \frac{\partial M_{ij}}{\partial q_k}\right)$$

where M_{ij} is the (i, j)-th element of $M(q)$. $B(q)$ is a matrix function describing how the control inputs influence the dynamics. In the gorilla robot only the last $(n-1)$ coordinates are actuated, leading to the constant matrix

$$B(q) = B = \begin{bmatrix} 0 \\ I_{(n-1)} \end{bmatrix}$$

This leads to equations of motion of the form

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = B(q)u$$

4.3.2 Double-support Phase

The transition between single-support and double-support occurs the moment the free hand of the robot grips the target handhold. From this moment and throughout the double-support phase, the robot is firmly holding on to two subsequent handholds. Some previously unmodeled contact forces from the front handhold will now impact the system dynamics. To include these, the model valid for the single-support phase must be modified. How this is done is shown in [1] and the main mechanics are described here for convenience.

The configuration of the robot during the double-support phase is shown in Figure 4.5. From geometric limits on the configuration during the double-support phase, it is clear that gripping the target handhold introduces some physical holonomic constraints to the system

$$\theta_3 = f_{\theta_3}(\theta_1, \theta_2) \tag{4.46a}$$

$$\theta_4 = f_{\theta_4}(\theta_1, \theta_2) \tag{4.46b}$$

where $f_{\theta_3}(q)$ and $f_{\theta_4}(q)$ are some (fairly complex) geometric functions defining the values of θ_3 and θ_4 during the loop phase[2].

The model derived for the single-support phase becomes valid for the double-support phase when the contact forces are included. This can be written as

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = B(q)u - J^T(q)\lambda \tag{4.47}$$

where λ a vector of contact forces resulting from the physical constraint.

The generalized coordinates are split into one part representing the minimum set of coordinates needed during the loop phase, q_m, and one part representing the redundant coordinates q_r. In the case of the monkey robot this can be written as

$$q_m = \begin{bmatrix}\theta_1 & \theta_2 & \theta_5 & \theta_6\end{bmatrix}^T$$

$$q_r = \begin{bmatrix}\theta_3 & \theta_4\end{bmatrix}^T$$

[2]For explicit expressions for these functions the reader is referred to [1]

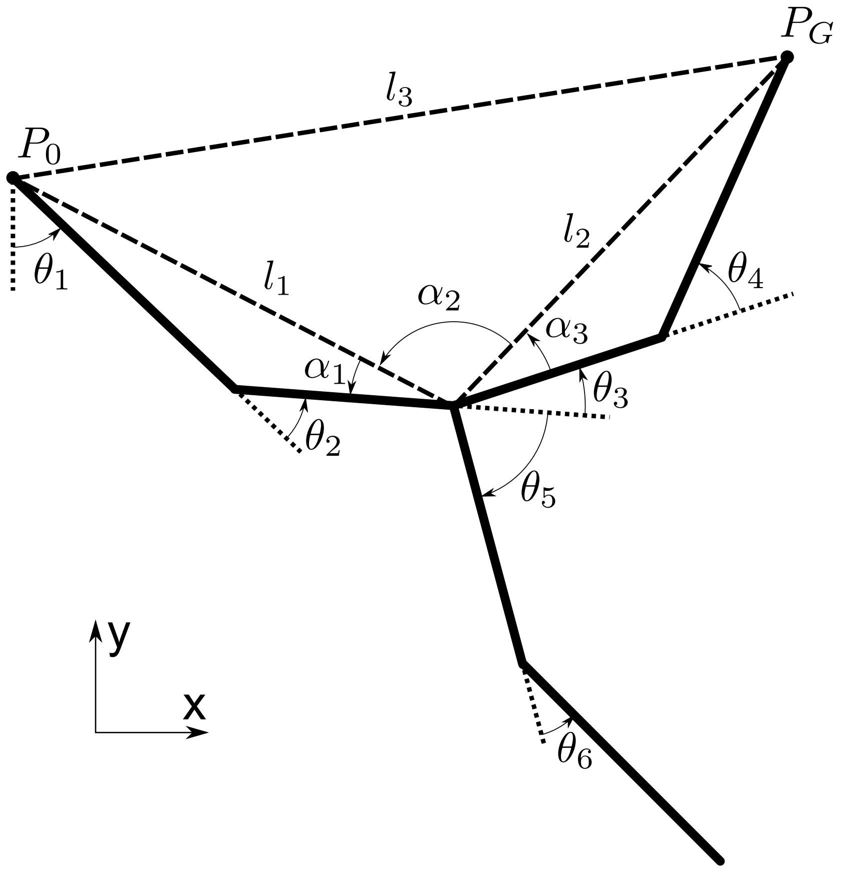


Figure 4.5: Robot configuration during loop phase

As $q_r \in \mathbb{R}^2$ is now written as a function of $q_m \in \mathbb{R}^4$, the equations of motion contains 4 unknown coordinates and 2 unknown constraint forces. As the rank of the dimension of the system matrix functions is 6, we are able to eliminate q_r and λ from the equations, leaving the reduced equations of motion that are valid for the double-support phase.

$$M_r(q_r)\ddot{q}_r + C_r(q_r, \dot{q}_r)\dot{q}_r + G_r(q_r) = B_r(q_r)u \tag{4.48}$$

where M_r and C_r are 4-by-4 matrix functions, G_r is a 4-by-1 vector function and B_r is a 4-by-5 matrix function. Hence the reduced system is actually overactuated by degree 1.

4.3.3 Phase Switching

The desired trajectory proposed in [1] is designed such that the transitions between phases are C^1-smooth, i.e. there are no discontinuities in neither the angles nor angular velocities of the physical joints. Note that this only applies to the angles of the joints. Due to the hybrid nature of the proposed motion, jumps in the generalized coordinates will and should occur as a relabeling of the coordinates is needed in order to choose one of the coordinates as the monotonic quantity θ.

Transitions between the two phases occur when certain switching conditions have been met. One obvious condition is that the target handhold must be reachable. However, as discussed in [1], it is not desirable to switch immediately when the front handhold becomes reachable, as this would leave the system vulnerable to disturbances. In addition, switching as soon as the target handhold becomes reachable necessarily requires the front arm to be completely stretched out, causing a singularity in the elbow joint. For this reason it is desirable to perform a switch when the angle about the handhold from which the robot is suspended, becomes slightly larger than the minimum angle from which the front handhold can be reached. The minimal angle from which the front handhold can be reached, θ_{min} can be computed using geometric relations. It is intuitively clear that this angle is found when both the rear and front arms are completely stretched out, i.e. $\theta_2 = \theta_4 = 0$. If the point $P_G = [x_G; y_G]$ is the position of the target handhold, and $P_0 = [x_0; y_0]$ is the position of the current handhold, then the minimum switching angle is found as follows

$$\theta_{min} = \arcsin\left(\frac{\sqrt{(x_G - x_0)^2 + (y_G - y_0)^2}}{2(l_7 + l_8)}\right) + \arctan\left(\frac{y_G - y_0}{x_G - x_0}\right) \tag{4.49}$$

where l_7 is the length of the upper arm of the robot and l_8 is the length of the lower arm.

Because of the aforementioned problems with singularities and robustness, the switching condition for θ is chosen as

$$\theta = \theta_{min} + \varepsilon \tag{4.50}$$

for some small ε defining the desired amount of overswing.

However, this is not a sufficient condition for a phase switch to be physically possible. For the robot to actually be in a position where it is able to grip the front handhold, the physical constraints (4.46) that are present in the double-support model must be respected. This is an interesting case, as it requires constraints that are enforced physically in the double-support phase to be enforced virtually at the end of the single-support phase in order for the switching conditions to be met. This presents a challenge when designing a controller that stabilizes the hybrid desired trajectory.

4.4 Stabilization of Preplanned Brachiating Motion

The proposed brachiation motion consists of two continuous phases connected by discrete jumps. In the ideal case, when the virtual holonomic constraints are perfectly satisfied, the jump simply consists of a relabeling of the generalized coordinates. It is worth mentioning a few characteristics of this hybrid motion.

- During the single-support phase, the system is underactuated of degree one, and the concept of transverse linearization may be applied. However as transverse linearization in the continuous case is concerned with asymptotic stability, it remains unclear whether a controller designed based on transverse linearization can guarantee that the switching conditions are satisfied.
- When in double-support phase, the system is overactuated of degree one. It is then possible to move the robot to any desired configuration (given that the amount of torque available is sufficient) and conventional non-linear control techniques may be applied in order to achieve convergence to the desired trajectory. For instance the full state feedback linearization described in e.g. [16] may be used.

- If the virtual holonomic constraints are perfectly satisfied, no impact forces effect the system as the phase switches ensures smooth transitions between the different continuous states. However, in an actual robot this will not be the case and impact forces will impact the dynamics to some extent. The controller should therefore be robust enough to stabilize the desired trajectory even when influenced by impact forces.

It becomes evident that the control problem can be split into two parts that can be designed independent of each other:

1. During the single-support phase, the motion is subject to dynamic constraints on the motion of $[\theta; \dot{\theta}]$. In this phase a controller should be designed with a focus towards satisfying the switching conditions. The error throughout the motion should be kept at a minimum, but the main priority should be to reach a configuration where a phase transition is indeed possible. As an error at the end of the phase could cause the robot to miss the target handhold this would obviously have a greater negative impact on the system than not being perfectly aligned with the target trajectory throughout the entire motion. It might be beneficial to divide the swing-controller into a set of different controllers with the switching between controllers to be determined by e.g. the distance traveled along the trajectory. This would allow a "swing"-controller to ensure sufficient proximity to the desired orbit throughout the first part of the swing, while a "grasping"-controller might be used to ensure the successful grasping of the target handhold when the distance to the target handhold is less than some threshold. This approach is taken in e.g. [11] where it was applied to a two-link brachiating robot.

2. During the double-support phase a controller based on feedback linearization is proposed. How to compute such a controller is well-documented (see e.g. [8] [16]), and the desired trajectory during the double-support phase should be possible to stabilize with relative ease.

The focus of this thesis is to achieve the first part, i.e. to find a controller that successfully brings the states during the single-support phase to a set in the state-space where the switching conditions are met and a jump to the double-support phase can be performed without violating any physical constraints.

As a relaxation of this problem, one can consider the *continued single-support phase*. We define the continued single-support phase as the trajectory the system exert if no phase switch is performed at the end of the swing phase while the virtual holonomic constraints are perfectly satisfied. This gives rise

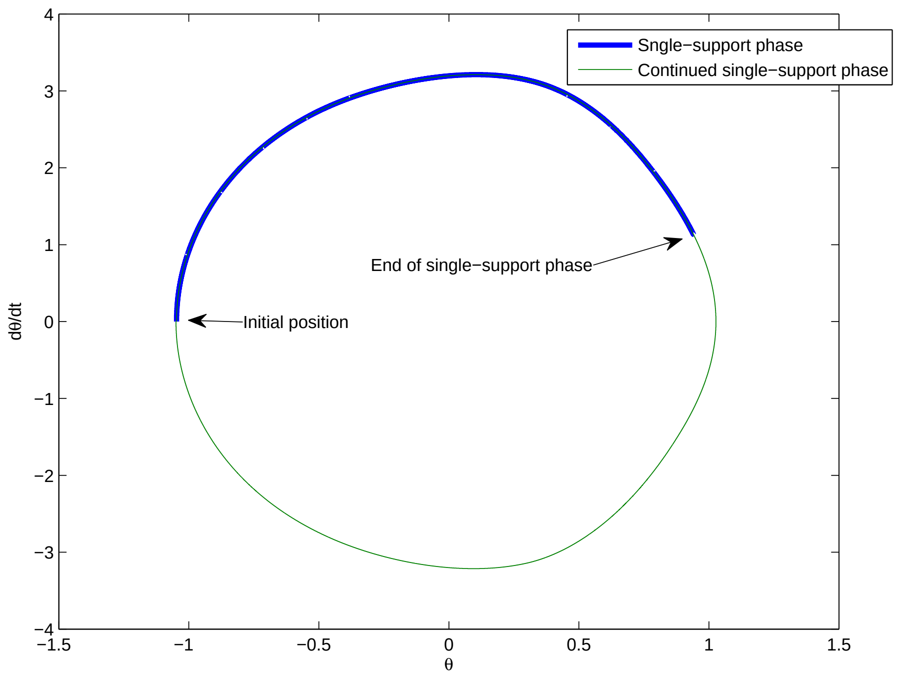


Figure 4.6: Phase plot of the θ during swing phase. The blue line shows the single-support phase while the green line shows the continued single-support phase.

to a continuous, pendulum-like motion in which the robot swings back and forth while suspended from one arm. This continued single-support phase is shown as the green line in Figure 4.6. Stabilizing the continued single-support phase can be done asymptotically and is considerably easier than stabilizing the regular single-support phase in finite time. As an initial step in stabilizing the full motion we therefore propose to stabilize the continued single-support phase, as this is a necessary condition for stabilizing the regular single-support phase. If the continued single-support phase can not be stabilized, there is no hope to stabilize the regular single-support phase.

Chapter 5

Results and Discussion

In this chapter simulation results for both the monkey robot and the inverted pendulum are presented and discussed for a number of cases.

First, the correctness of the partial feedback linearization will be verified by running simulations with only this part of the controller engaged, i.e. with the auxiliary control input, $v(t)$ set to zero. The systems will be simulated for a number of initial conditions and the behavior will be asserted. In this case, the system control variable, $u(t)$ is given by

$$u(t) = N^{-1}(\theta, y)R(\theta, \dot{\theta}, y, \dot{y}) \tag{5.1}$$

and the dynamics of y is given by

$$\begin{aligned} v(t) &= 0 \\ \ddot{y}(t) &= v(t) = 0 \\ \dot{y}(t) &= \dot{y}(0) + \int_0^t \ddot{y}(\tau)d\tau = \dot{y}(0) = \dot{y}_0 \\ y(t) &= y(0) + \int_0^t \dot{y}(\tau)d\tau = y_0 + \dot{y}_0 t \end{aligned} \tag{5.2}$$

Next, a simple controller that will asymptotically zero the error $[y; \dot{y}]$ in the actuated states will be tested. The control law is in the form

$$\begin{aligned} v(t) &= -K_y \begin{bmatrix} y \\ \dot{y} \end{bmatrix} && (5.3) \\ &= -\begin{bmatrix} 0 & K_y \end{bmatrix} x_\perp(t) && (5.4) \end{aligned}$$

for some appropriate constant matrix K_y. By noting that the dynamics of $y(t)$ can be written as the linear time-invariant system

$$\begin{bmatrix} \dot{y} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} v \tag{5.5}$$

it becomes evident that such a stabilizing K_y can be found with the use of well-established linear system theory. The approach taken in this work is simply to place the poles of the system (5.5) such that the real value of all poles is less than zero.

Finally, simulations will be run on the systems with a controller for the full transverse dynamics engaged, i.e. with a control law

$$v(t) = -K(\tau)x_\perp(t) \tag{5.6}$$

Finding such a controller is considerably more challenging than in the previous case as it involves controlling the linear time-varying system of transverse linearized dynamics.

5.1 Inverted Pendulum

The inverted pendulum system were simulated for initial conditions of the form

$$\theta_0 = \theta_\star(0) + \delta_\theta \tag{5.7a}$$
$$\dot{\theta}_0 = \dot{\theta}_\star(0) + \delta_{\dot{\theta}} \tag{5.7b}$$
$$q_0 = \Phi(\theta_0) + \begin{bmatrix} y_0 \\ 0 \end{bmatrix} \tag{5.7c}$$
$$\dot{q}_0 = \Phi'(\theta_0)\dot{\theta}_0 + \begin{bmatrix} \dot{y}_0 \\ 0 \end{bmatrix} \tag{5.7d}$$

These initial conditions are chosen such that if the systems are initialized with $[\theta; \dot{\theta}]$ away from the desired orbit, the rest of the states $[q; \dot{q}]$ satisfy the virtual holonomic constraints as long as $y_0 = \dot{y}_0 = 0$. This is consistent with the notation used previously, where y and $\dot{y}$ specifies the amount of error in the actuated coordinates with respect to the virtual holonomic constraints.

5.1.1 Feedback Linearization

In this case only the partial feedback linearization is active. As the auxiliary control input, $v(t)$ is set to identically zero, the control input to the system becomes

$$u(t) = N^{-1}(\theta, y)R(\theta, \dot{\theta}, y, \dot{y}) \tag{5.8}$$

First, the system is simulated starting exactly from the desired starting position, i.e. the initial conditions are given by (5.7) with

$$\delta_\theta = \delta_{\dot{\theta}} = y_0 = \dot{y}_0 = 0 \tag{5.9}$$

Figure 5.1a shows the phase plot of both $[\theta; \dot{\theta}]$ and $[x; \dot{x}]$. Figure 5.1b shows the time-evolution of $x_\perp(t)$. As expected, the actual orbit coincides with the desired orbit, and the errors $x_\perp$ are identically zero, indicating that the partial feedback linearization works as intended.

Next, the system is simulated from the initial conditions (5.7) with

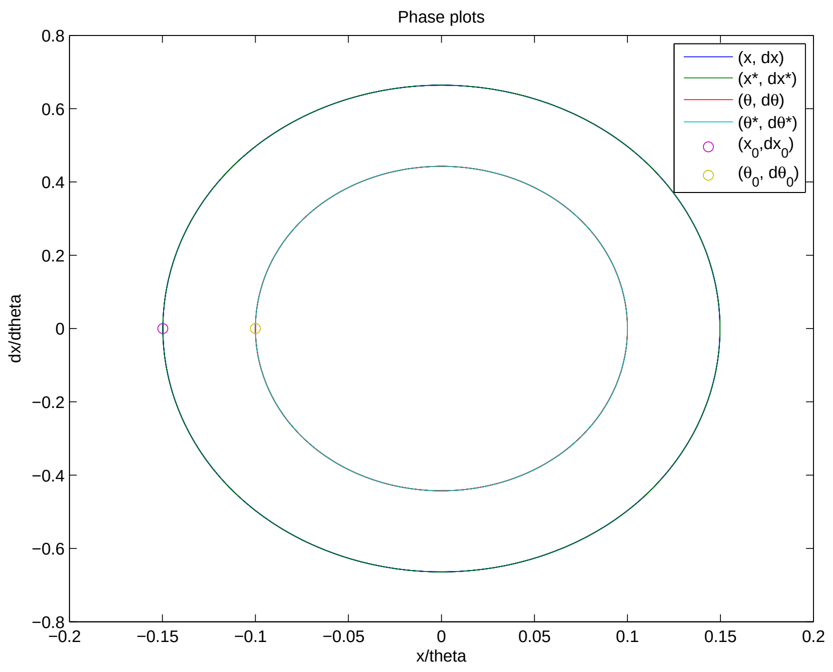
$$\begin{aligned} \delta_\theta &= -0.1 \\ \delta_{\dot{\theta}} &= 0.1 \\ y_0 &= \dot{y}_0 = 0 \end{aligned}$$

i.e. the virtual holonomic constraints are perfectly satisfied, but $[\theta, \dot{\theta}]$ is initialized away from the desired orbit. The resulting phase plots are shown in Figure 5.2a and $x_\perp(t)$ is shown in Figure 5.2b. From the plots we see that the result is a closed periodic orbit, but not the desired one.

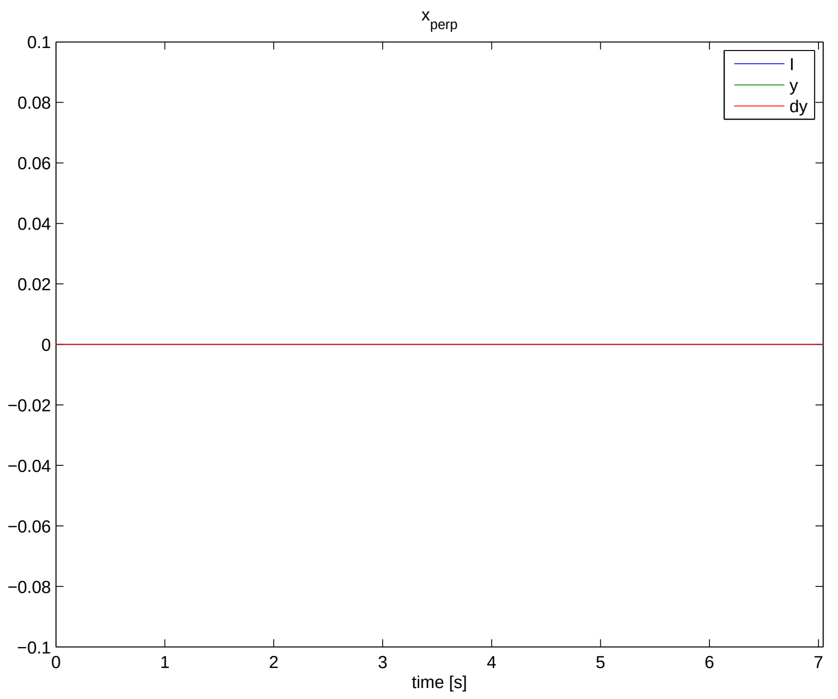
It is also interesting to see how the system responds when initialized such that $[\theta; \dot{\theta}]$ is on the desired orbit, while $[x, \dot{x}]$ is away from the desired orbit. This is done by choosing

$$\begin{aligned} \delta_\theta &= \delta_{\dot{\theta}} = 0 \\ y_0 &= -0.1 \\ \dot{y}_0 &= 0.1 \end{aligned}$$

This is shown in Figure 5.3. The expected result is that $\dot{y}(t) = y_0$ is constant, while $y(t)$ is a ramp function with slope $\dot{y}_0$ and initial value y_0. This is exactly what is seen in Figure 5.3b. The result is also verified by Figure 5.3a where we see that the x-coordinate drifts away from the desired orbit.

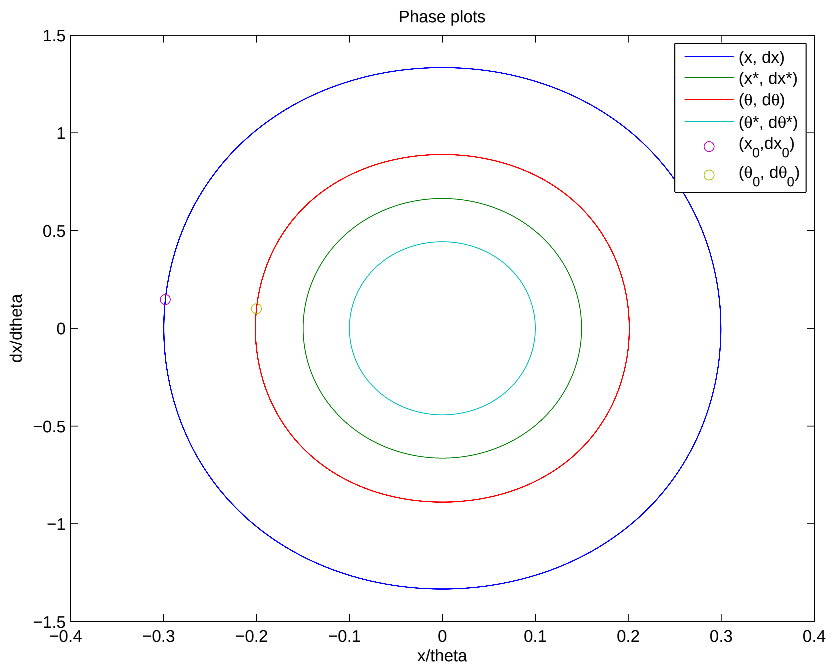


(a) Phase plots. The desired orbits are perfectly overlapped by the orbit experienced by the system.

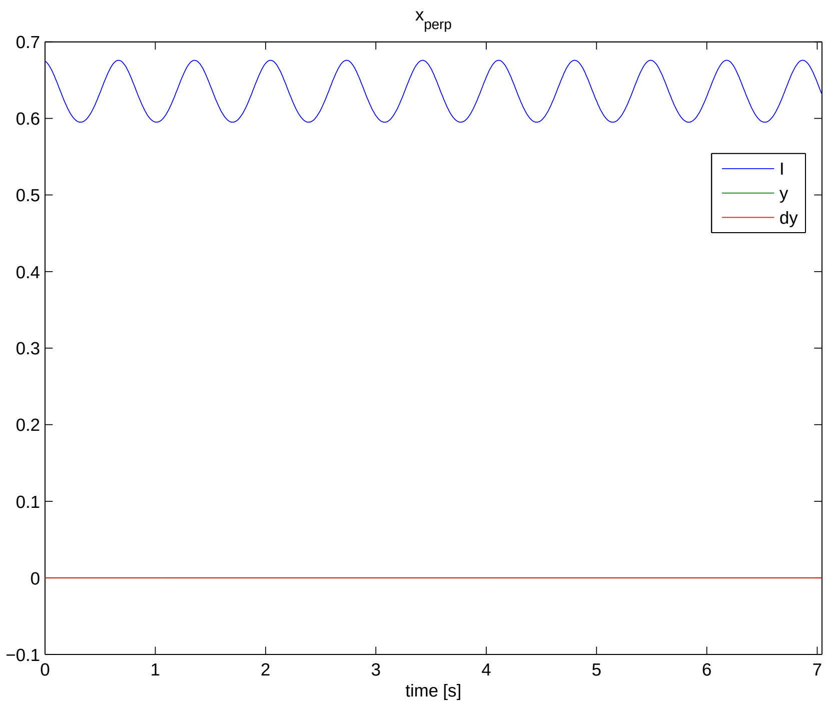


(b) Time-evolution of $x_\perp$.

Figure 5.1: Phase plots and time-evolution of error when initialized with $\delta_\theta = \delta_{\dot\theta} = y_0 = \dot y_0 = 0$.

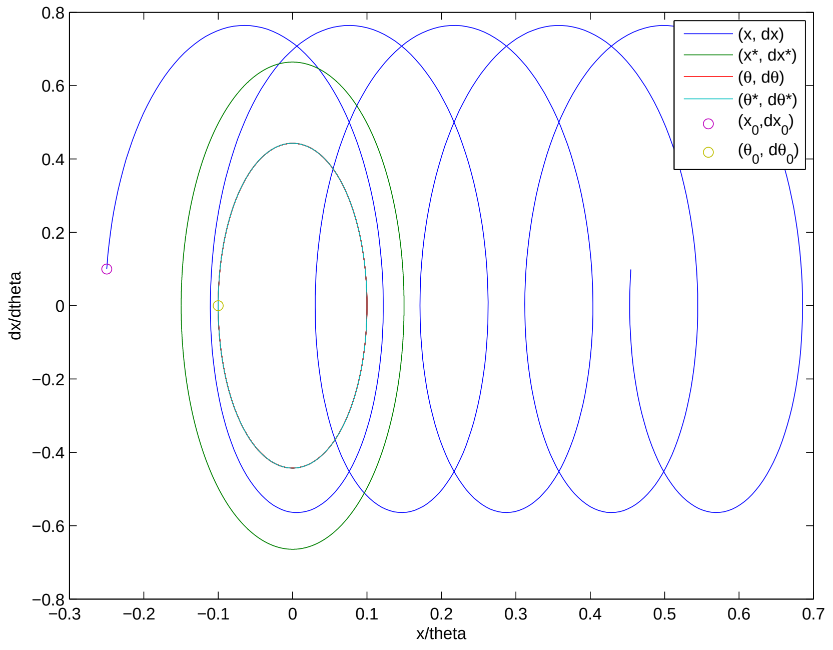


(a) Phase plot.

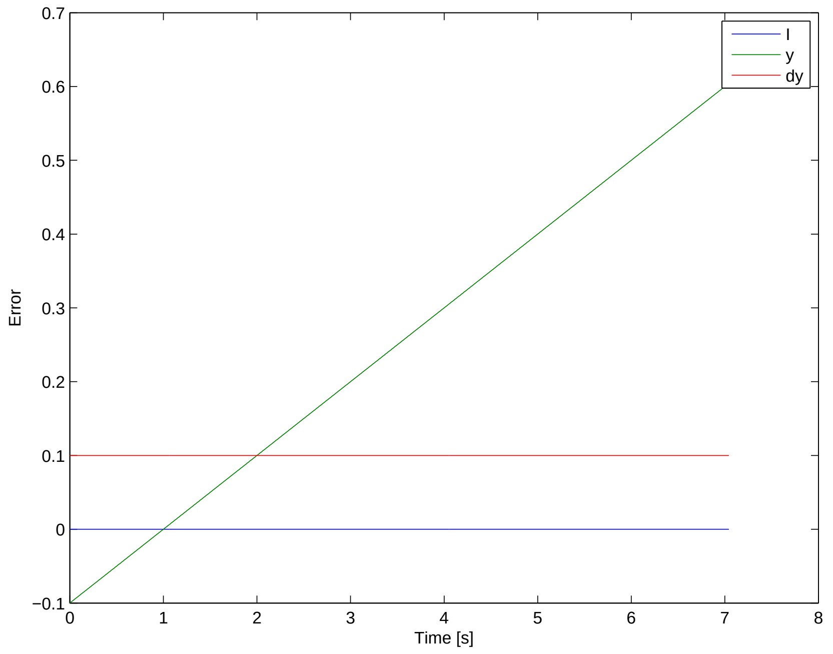


(b) Time-evolution of $x_\perp$.

Figure 5.2: Phase plots and time-evolution of error when initialized with $\delta_\theta = -0.1$, $\delta_{\dot\theta} = 0.1$ and $y_0 = \dot y_0 = 0$.



(a) Phase plot.



(b) Time-evolution of $x_\perp$.

Figure 5.3: Phase plots and time-evolution of error when initialized with $\delta_\theta = \delta_{\dot\theta} = 0$, $y_0 = -0.1$ and $\dot y_0 = 0.1$.

5.1.2 Enforcing the Virtual Holonomic Constraints

As an initial attempt to stabilize the the desired orbit, one might attempt to bring the errors $[y; \dot{y}]$ to zero. This is not difficult to do as $[y; \dot{y}]$ evolves according to the linear time-invariant system (5.5). As mentioned, standard control theory for LTI systems is applicable, and a stabilizing control law on the form

$$v(t) = -K_y \begin{bmatrix} y \\ \dot{y} \end{bmatrix} \tag{5.10}$$

can be found e.g. by using the built-in MATLAB function place. In this case the linear system has two states y and $\dot{y}$ and thus has two poles. By placing these in the left half plane (with real value less than zero) the closed-loop system

$$\begin{bmatrix} \dot{y} \\ \ddot{y} \end{bmatrix} = \left(\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix} K_y \right) \begin{bmatrix} y \\ \dot{y} \end{bmatrix} \tag{5.11}$$

is asymptotically stable [3]. We choose to place the poles in $[-2, -5]$ by using the MATLAB command

```
>> K = place(A,B,[-2,-5])
```

The output of this is the vector of control gains K_y

$$K_y = \begin{bmatrix} 10 & 7 \end{bmatrix}.$$

This stabilizes the y-dynamics, such that the control law

$$v(t) = -\begin{bmatrix} 0 & K_y \end{bmatrix} x_\perp(t) \tag{5.12}$$

stabilizes the y and $\dot{y}$ parts of $x_\perp$. The system is simulated with an initial error to both $[x; \dot{x}]$ and $[y; \dot{y}]$, specifically

$$\delta_\theta = -0.1 \tag{5.13a}$$
$$\delta_{\dot{\theta}} = 0.1 \tag{5.13b}$$
$$y_0 = -0.1 \tag{5.13c}$$
$$\dot{y}_0 = 0.1. \tag{5.13d}$$

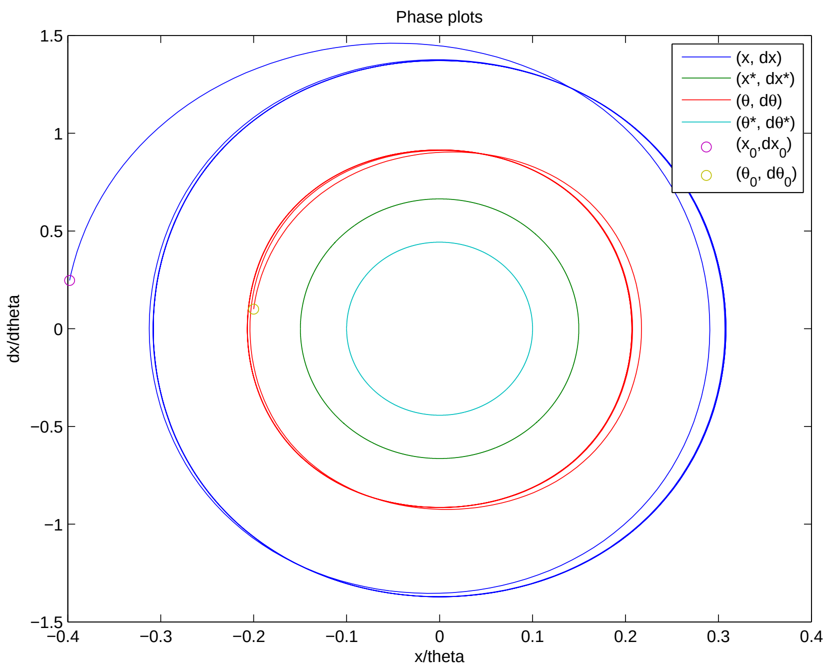
The resulting phase plots and $x_\perp$ are shown in Figure 5.4. It is clear that this controller asymptotically enforces the virtual holonomic constraints. However, as there is still an error in $[\theta, \dot{\theta}]$, the orbit of $[x; \dot{x}]$ is not the desired orbit $[x_\star; \dot{x}_\star]$. This is because

$$\begin{bmatrix}\theta(t)\\ \dot{\theta}(t)\end{bmatrix} \neq \begin{bmatrix}\theta_\star(t)\\ \dot{\theta}_\star(t)\end{bmatrix}$$

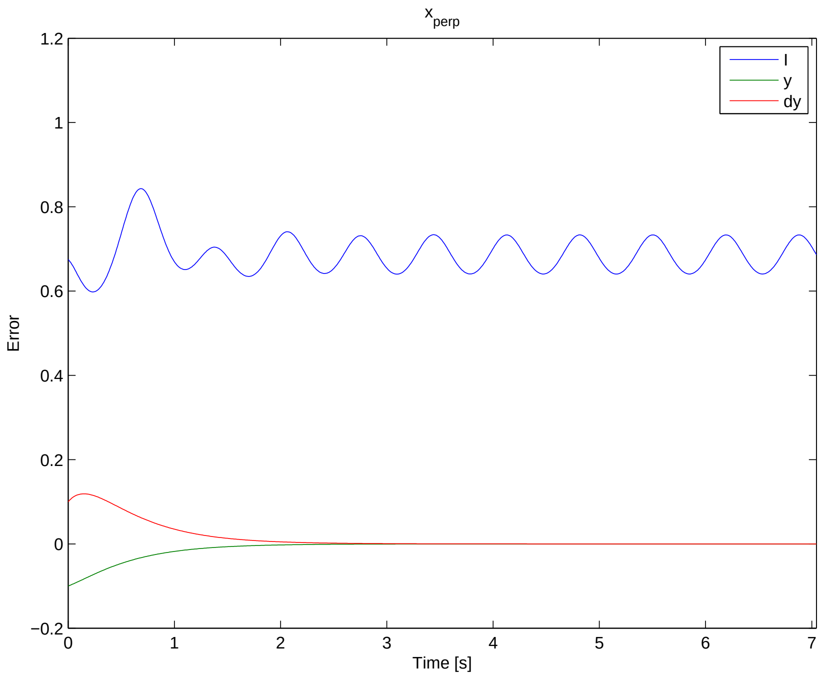
so that

$$\begin{bmatrix}x(t)\\ \dot{x}(t)\end{bmatrix} = \begin{bmatrix}\phi(\theta(t))\\ \phi'(\theta(t))\dot{\theta}(t)\end{bmatrix} \neq \begin{bmatrix}\phi(\theta_\star(t))\\ \phi'(\theta_\star(t))\dot{\theta}_\star(t)\end{bmatrix} = \begin{bmatrix}x_\star(t)\\ \dot{x}_\star(t)\end{bmatrix}$$

Hence, simply enforcing the virtual holonomic constraints does not, in general, stabilize the desired periodic orbit.



(a) Phase plot.



(b) Time-evolution of $x_\perp$.

Figure 5.4: Phase plot and time-evolution of error when initialized with $\delta_\theta = -0.1$, $\delta_{\dot{\theta}} = 0.1$, $y_0 = -0.1$ and $\dot{y}_0 = 0.1$.

5.1.3 Transverse Dynamics Controller

In the previous section an approach to asymptotically zeroing the dynamics $[y; \dot{y}]$ was demonstrated. As noted, this is however not sufficient for stabilization of a the desired periodic orbit, as the orbit $[\theta; \dot{\theta}]$ does not converge to that of $[\theta_\star; \dot{\theta}_\star]$. To do this the origin of the transverse dynamics, $x_\perp = [I(\theta, \dot{\theta}, \theta_0, \dot{\theta}_0), y, \dot{y}]^T$ must be stabilized. In order to do so, the system of linearized transverse dynamics, given by (3.40) and (3.41) is considered.

$$\frac{d}{d\tau} z = A(\tau) z(\tau) + B(\tau) V_\bullet(\tau) \tag{5.14}$$

where z is the linearized transverse dynamics $z = [I_\bullet; Y_\bullet; \dot{Y}_\bullet]$. It is not surprising that the control law derived in the previous section does not stabilize this dynamics as it does not include feedback from the state $I_\bullet$.

In order to stabilize the transverse dynamics, a stabilizing controller should be found for the linear periodic time-varying system (5.14). This is a considerably more challenging task than stabilizing the time-invariant system (5.5). A common approach is to compute the solution to the periodic Riccati equation and use this solution in the control law [14]. This is in general a challenging task and instead of doing this we perform the numerical search described in Section 4.2.3. The search is done for a periodic matrix of control gains

$$K(\tau) = K(\tau + T) \tag{5.15}$$

where the elements are Beziér curves parametrized by the optimization coefficients. In the inverted pendulum case there are one actuated state and 3 transverse errors. A search is thus made for three control gains $K(\tau, c) = [k_1(\tau, c), k_2(\tau, c), k_3(\tau, c)]$ that minimizes the cost function

$$f(\mathbf{c}) = \int_{t_0}^{T} w(\tau) \|\Phi(\tau, t_0)\| d\tau \tag{5.16}$$

where the weight function $w(\tau)$ is chosen as

$$w(\tau) = \frac{e^\tau - e^{t_0}}{e^T - e^{t_0}} \tag{5.17}$$

This weight function and the resulting time-evolution of the norm of the state transition matrix for (5.14) is shown in Figure 5.5. The control gains resulting from the optimization are shown in Figure 5.6.

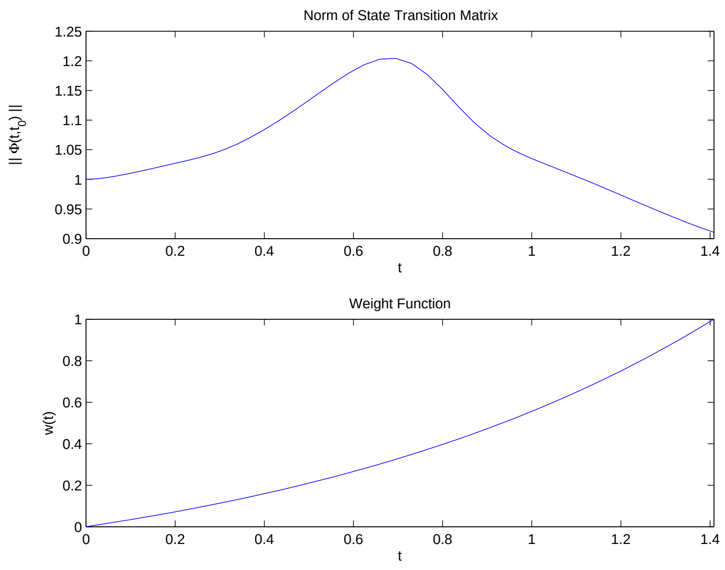


Figure 5.5: The time evolution of the norm of the state transition matrix (top) and the weight (bottom) for the resulting controller.

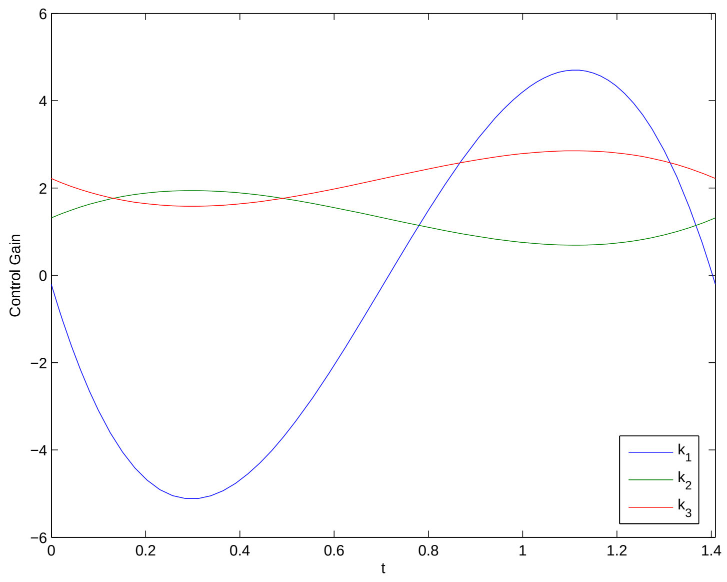


Figure 5.6: Time evolution of each element in the control gain matrix function $K(\tau) = [k_1(\tau), k_2(\tau), k_3(\tau)]$

Note that this system is clearly not asymptotic exponentially stable as there does not exist constants k and λ such that the condition from Section 3.3 holds.

$$\|\Phi(t, t_0)\| \leq ke^{-\lambda(t-t_0)}, \qquad \forall\, t \geq t_0 \geq 0 \tag{5.18}$$

However, as the norm is clearly bounded above and the norm at the end of the period is less than one, $\|\Phi(T, t_0)\| < 1$, we have that

$$\begin{aligned}\|\Phi(nT, t_0)\| &= \|\Phi\Big(nT, (n-1)T\Big)\Phi\Big((n-1)T, (n-2)T\Big)\dots\Phi\Big(T, t_0\Big)\| \\ &\leq \|\Phi\Big(nT, (n-1)T\Big)\|\|\Phi\Big((n-1)T, (n-2)T\Big)\|\dots\|\Phi\Big(T, t_0\Big)\| \\ &\to 0 \text{ as } n \to \infty\end{aligned}$$

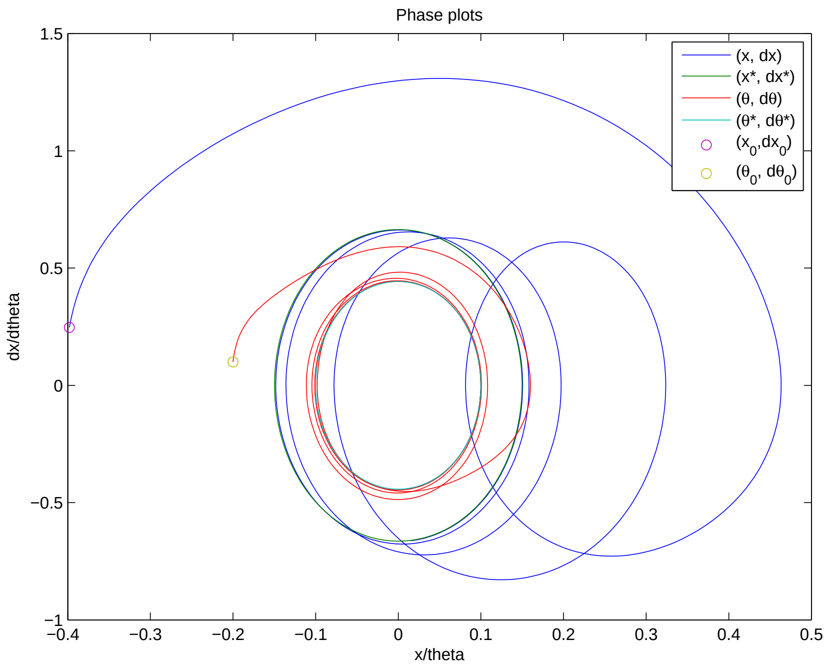
Where we have used that $\Phi(t, t_0) = \Phi(t, t_1)\Phi(t_1, t_0)$ (from [3]) and well-known properties of the norm of square matrices, $\|AB\| \leq \|A\|\|B\|$ for A, B square. Thus the closed-loop system is asymptotically stable, but not asymptotic exponentially stable.

The system is now simulated from the same initial position as in the case where only the y-dynamics were attempted controlled, i.e. the initial conditions (5.7) with the initial errors given in (5.13). The phase plot and time evolution of $x_\perp$ are shown in Figure 5.7. It is evident that using this controller does in fact render the system asymptotically stable.

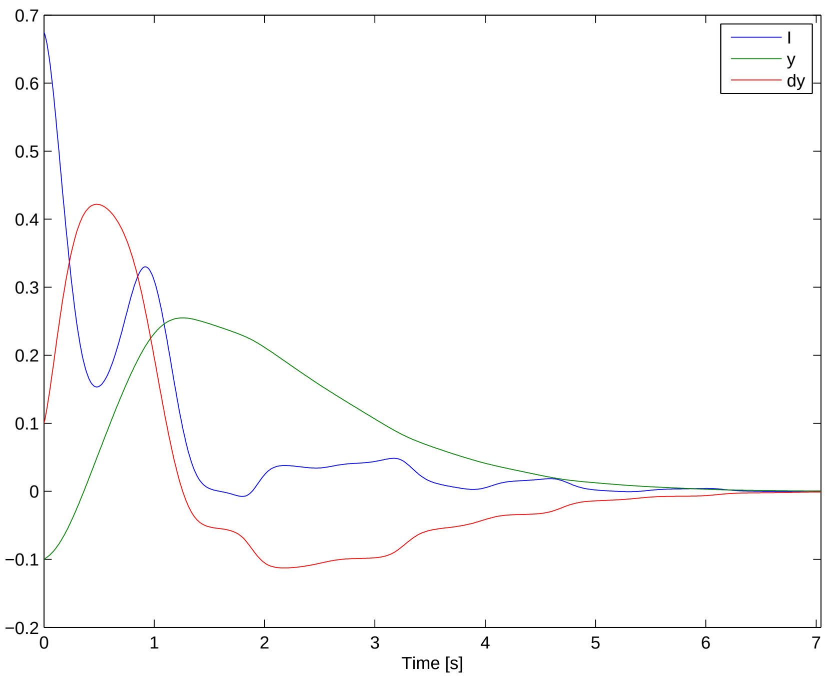
In order to test the stability for more challenging initial conditions, the system is simulated again. This time initialized with larger errors. Specifically, we choose

$$\delta_\theta = -0.1 \tag{5.19a}$$
$$\delta_{\dot\theta} = -0.2 \tag{5.19b}$$
$$y_0 = 2.0 \tag{5.19c}$$
$$\dot y_0 = 0.5. \tag{5.19d}$$

The resulting phase plots and $x_\perp(t)$ are shown in Figure 5.8a and Figure 5.8b, respectively. In addition, the control inputs and the control gains are shown in Figure 5.9 and the time evolution of the states are shown in Figure 5.10.

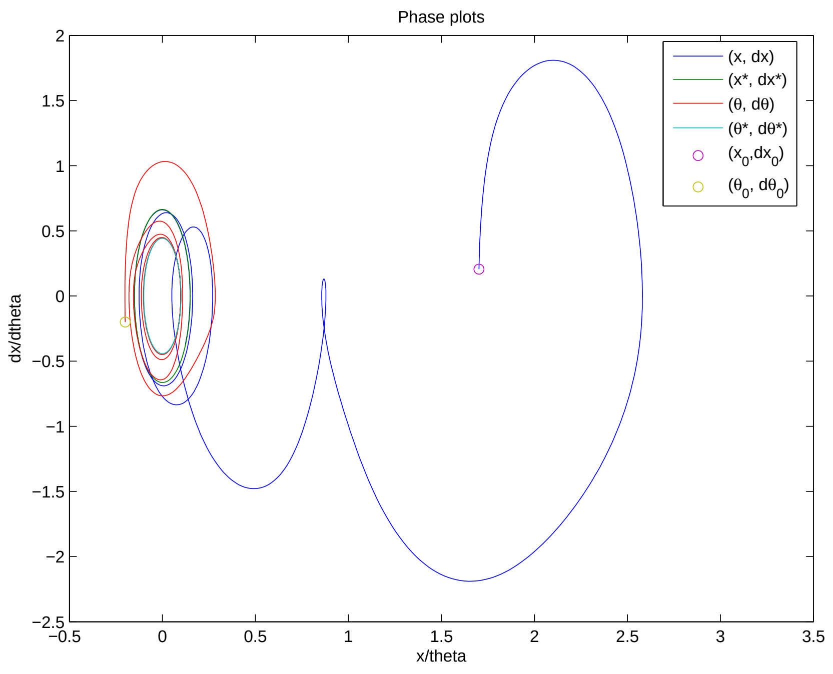


(a) Phase plot.

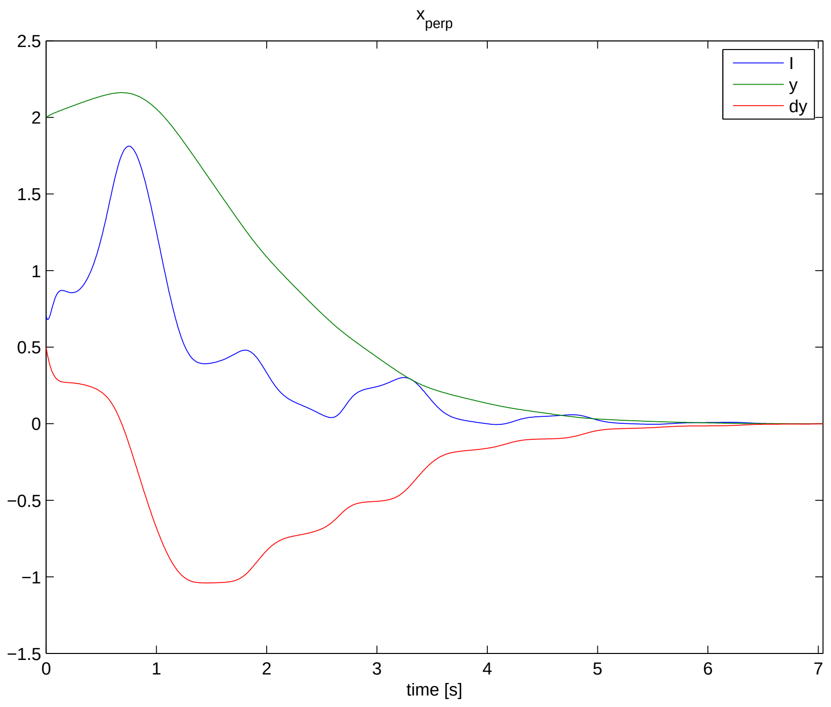


(b) Time-evolution of $x_\perp$.

Figure 5.7: Phase plot and time-evolution of error with transverse dynamics controller engaged, initialized with $\delta_\theta = y_0 = -0.1$ and $\delta_{\dot\theta} = \dot y_0 = 0.1$.



(a) Phase plots



(b) Time-evolution of $x_\perp$.

Figure 5.8: Phase plot and time-evolution of error with scalar controller engaged, initialized with $\delta_\theta = -0.1$, $\delta_{\dot{\theta}} = -0.2$, $y_0 = 2$ and $\dot{y}_0 = 0.5$.

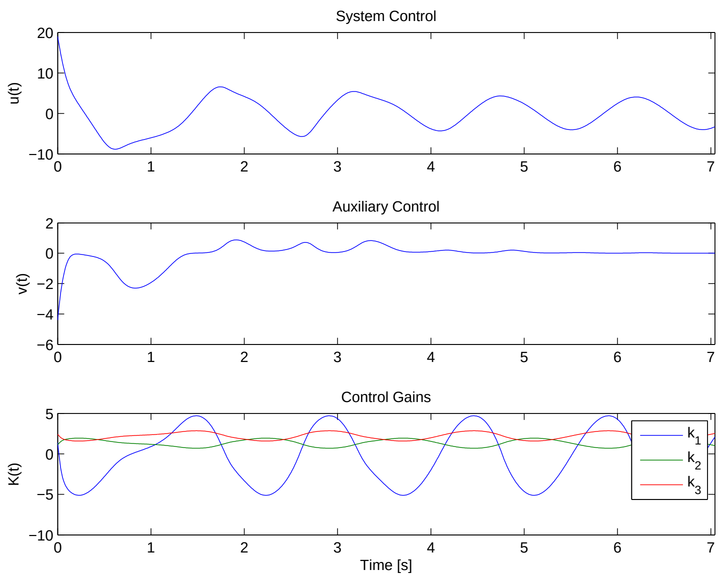


Figure 5.9: Control input and control gains initialized with large errors.

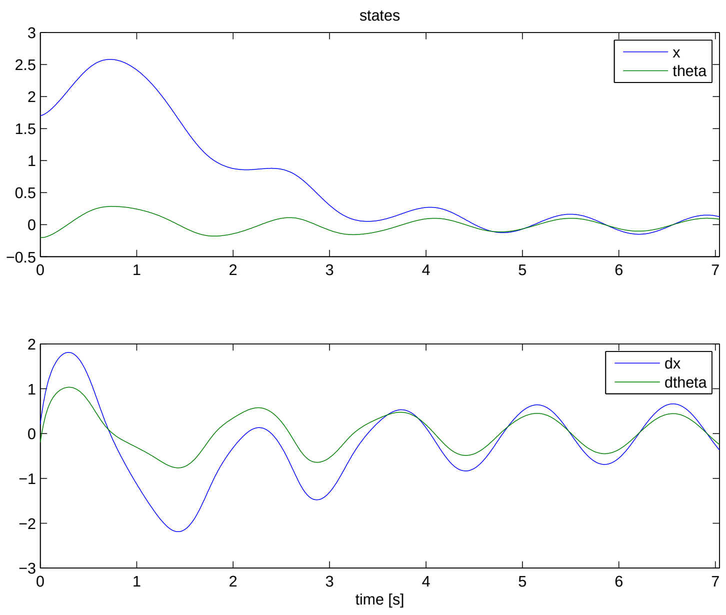


Figure 5.10: Time evolution of states, initialized with large errors.

5.2 Monkey Robot

The monkey robot was simulated over two periods from initial conditions of the form

$$\theta_0 = \theta_\star(0) + \delta_\theta \tag{5.20a}$$

$$\dot{\theta}_0 = \dot{\theta}_\star(0) + \delta_{\dot{\theta}} \tag{5.20b}$$

$$q_0 = \Phi(\theta_0) + \begin{bmatrix} 0 \\ y_0 \end{bmatrix} \tag{5.20c}$$

$$\dot{q}_0 = \Phi'(\theta_0)\dot{\theta}_0 + \begin{bmatrix} 0 \\ \dot{y}_0 \end{bmatrix} \tag{5.20d}$$

This differs slightly from the ones used on the inverted pendulum, in that the errors y_0 and $\dot{y}_0$ are now on the last $(n-1)$ generalized cordinates instead of the first. This is because in the monkey robot case the θ variable is chosen to be the first generalized coordiate instead of the last. For simplicity we further limit the choices of y_0 and $\dot{y}_0$ to errors on the coordinate q_2, i.e.

$$y_0 = \begin{bmatrix} y_{02} & 0 & 0 & 0 & 0 \end{bmatrix}^T$$

$$\dot{y}_0 = \begin{bmatrix} \dot{y}_{02} & 0 & 0 & 0 & 0 \end{bmatrix}^T$$

The obtained results are still valid for errors on the remaining coordinates.

5.2.1 Partial Feedback Linearization

To verify the partial feedback linearization module, the auxiliary control input v is set to zero and the system is simulated for various initial conditions.

First, the system is simulated from the ideal initial conditions, i.e. (5.20) with

$$\delta_\theta = \delta_{\dot{\theta}} = y_{02} = \dot{y}_{02} = 0. \tag{5.21}$$

The resulting phase portrait for $\theta(t)$ and $q_2(t)$ are shown in Figure 5.11. We note that there is a small misalignment in between the actual and desired phase portrait of q_2. This is because the desired trajectory for $q_2(t)$ is given by

$$q_2 = \phi_2(\theta) \neq \phi_2(\theta_\star) = q_{2\star} \tag{5.22}$$

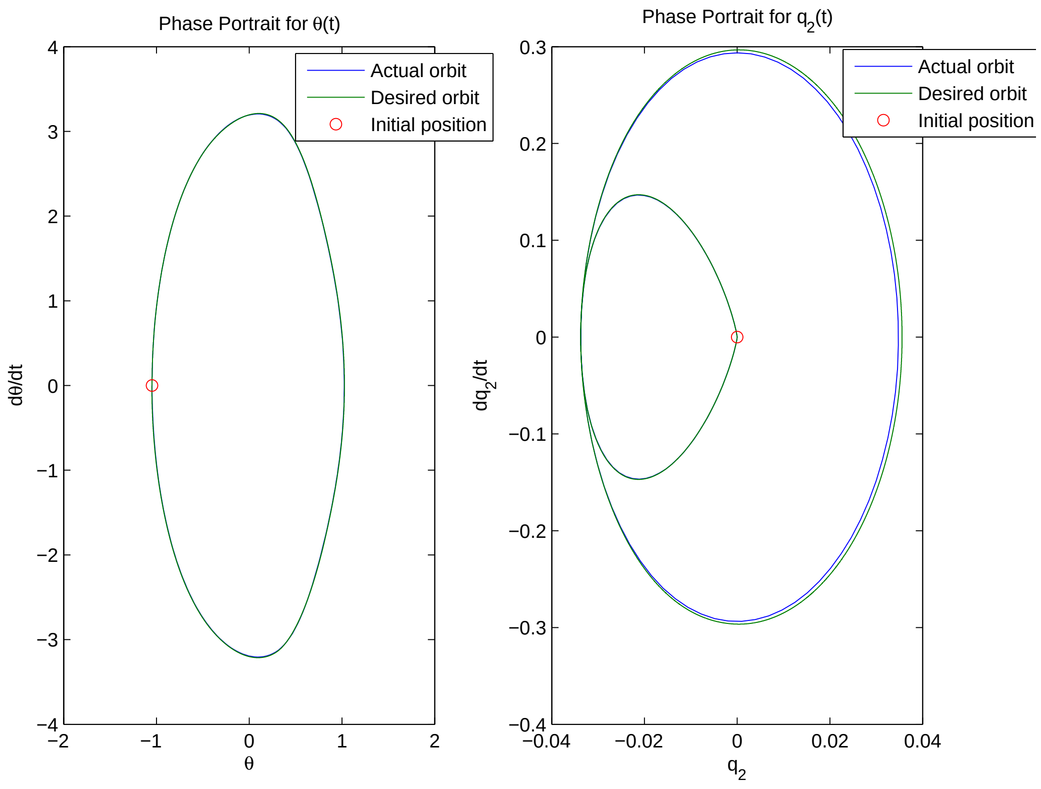


Figure 5.11: Phase plots of θ and q_2. Initialized without errors

when there are small numerical errors in $\theta(t)$. This is confirmed when plotting the orbit $[\phi_2(\theta); \phi_2'(\theta)\dot{\theta})]$, as this perfectly overlaps the orbit $[q_2(t); \dot{q}_2(t)]$.

Next, the system is simulated from an error in $[\theta; \dot{\theta}]$.

$$\begin{aligned} \delta_\theta &= 0.1 \\ \delta_{\dot{\theta}} &= -0.1 \\ y_{20} = \dot{y}_{20} &= 0 \end{aligned}$$

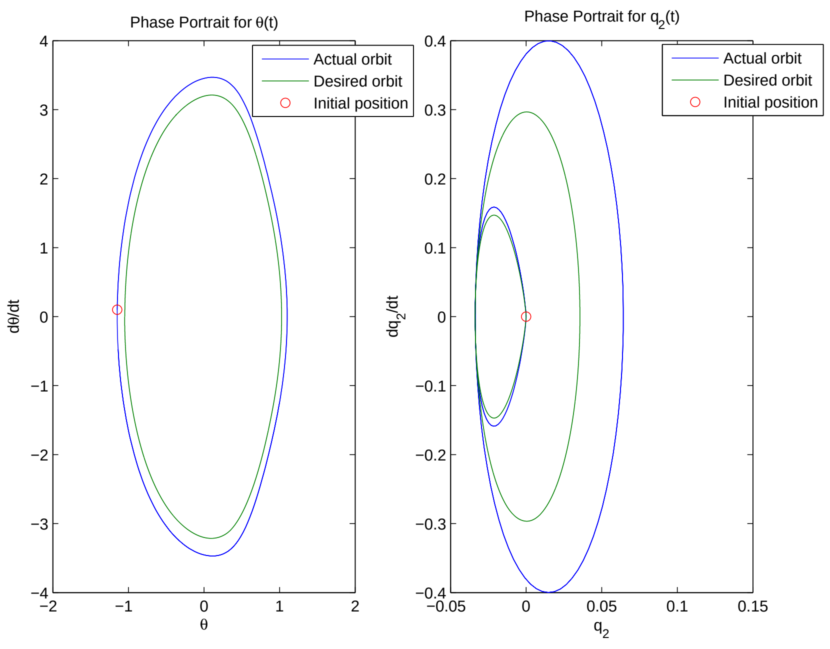
The resulting phase portraits are shown in Figure 5.12a. We see that the phase portraits converge to periodic orbits, but these orbits are not the desired ones. This is expected when no controller is engaged. From Figure 5.12b we see that the errors $[y_2; \dot{y}_2]$ remain zero, indicating that the virtual holonomic constraints are satisfied.

Finally, the system is simulated from an error in $[q; \dot{q}]$, but not in $[\theta; \dot{\theta}]$. The initial conditions are

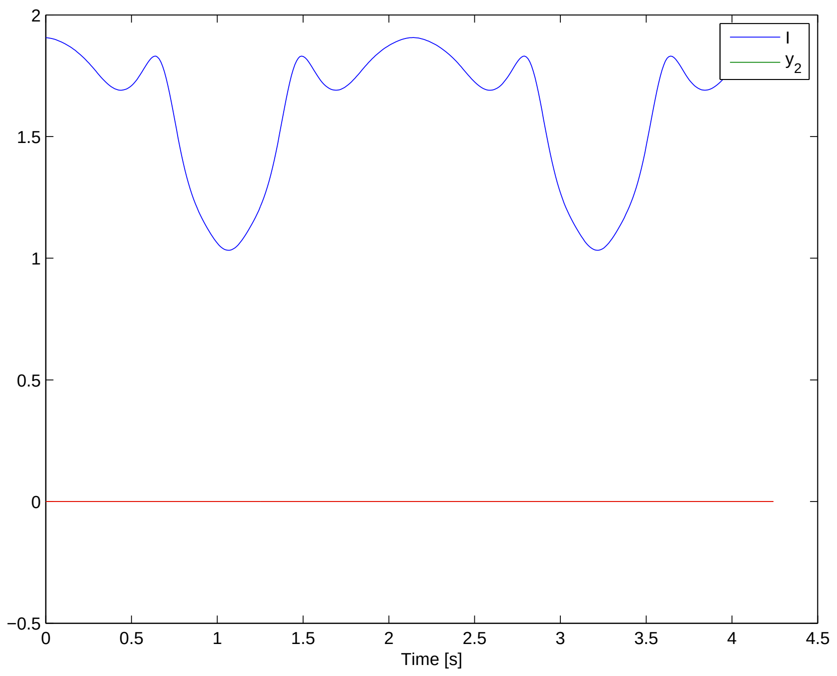
$$\begin{aligned} \delta_\theta = \delta_{\dot{\theta}} &= 0 \\ y_{20} &= -0.1 \\ \dot{y}_{20} &= 0.1. \end{aligned}$$

As expected, we see that the system drifts away from the desired orbit, with y linearly increasing with time, while $\dot{y}$ is constant.

From the above results, we conclude that the expected behavior from the partial feedback linearization module is in fact the behavior seen in the results, and we conclude that this part of the controller works well.

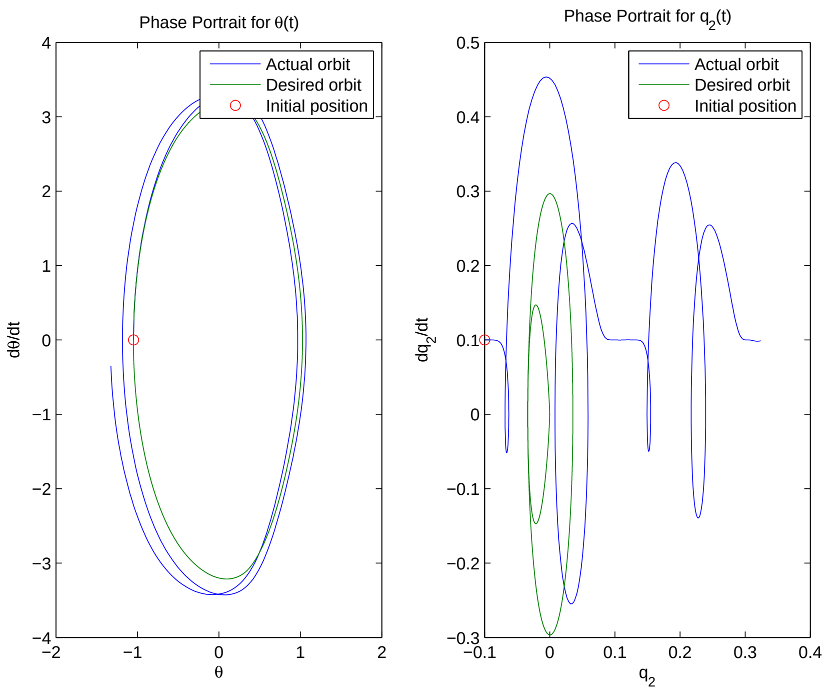


(a) Phase portraits.

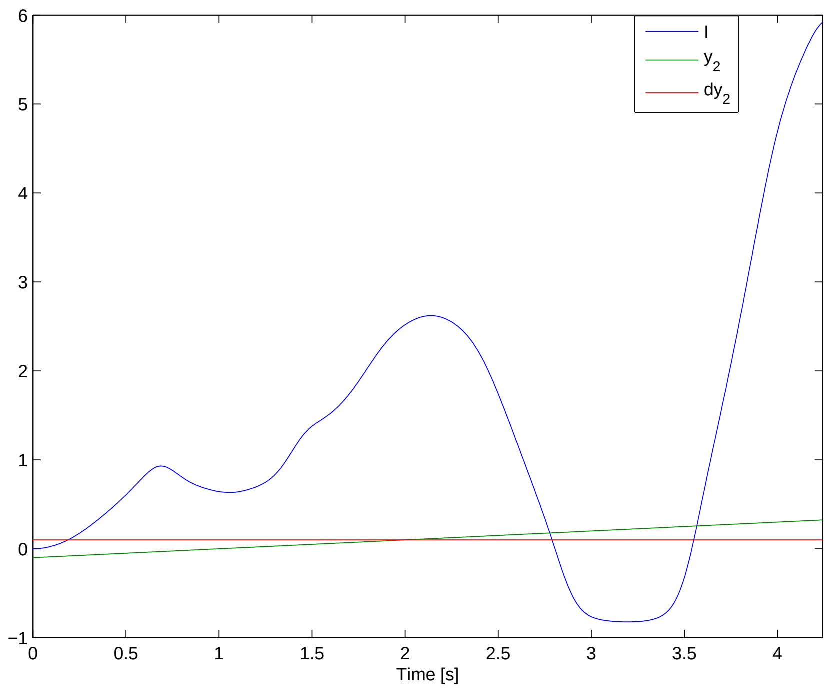


(b) Time evolution of $x_\perp$.

Figure 5.12: Simulation with an initial error $\delta_\theta = -0.1$, $\delta_{\dot{\theta}} = 0.1$, $y_{02} = \dot{y}_{02} = 0$.



(a) Phase portraits.



(b) Time evolution of the $x_{\perp}$.

Figure 5.13: Simulation with an initial error $\delta_\theta = \delta_{\dot{\theta}} = 0$, $y_{02} = -0.1$, $\dot{y}_{02} = 0.1$.

5.2.2 Enforcing the Virtual Holonomic Constraints

Equivalent to the inverted pendulum system, we now aim to utilize the partial feedback linearization in order to asymptotically satisfy the virtual holonomic constraints. As mentioned, the y-dynamics evolves according to the linear time-invariant system (5.5), and a stabilizing controller can be found with the following MATLAB command

```
>> K = place(A,B,-1:-1:-10)
```

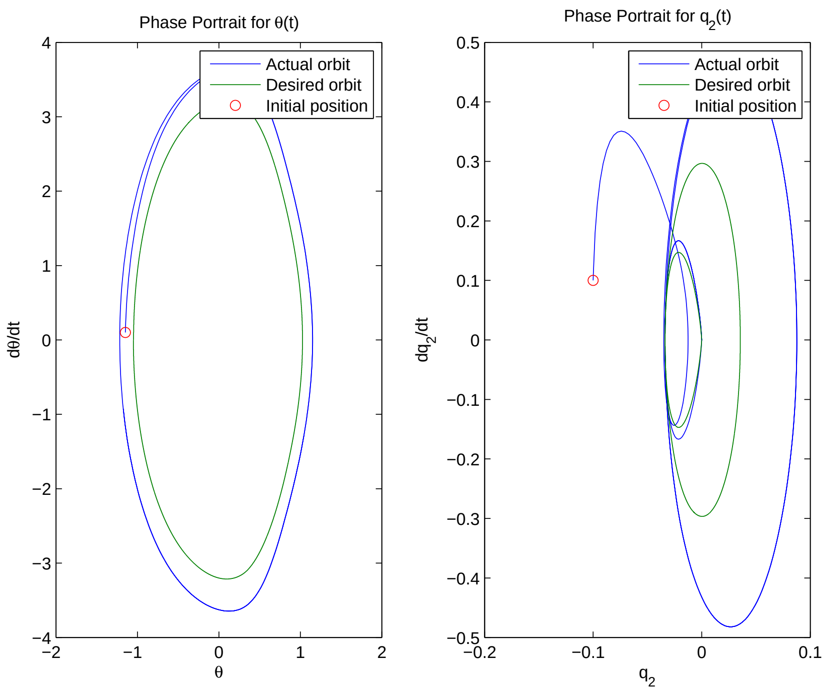
This computes the matrix of control gains

$$K = \begin{bmatrix} 90 & 0 & 0 & 0 & 0 & 19 & 0 & 0 & 0 & 0 \\ 0 & 20 & 0 & 0 & 0 & 0 & 9 & 0 & 0 & 0 \\ 0 & 0 & 24 & 0 & 0 & 0 & 0 & 11 & 0 & 0 \\ 0 & 0 & 0 & 14 & 0 & 0 & 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 7 \end{bmatrix}$$

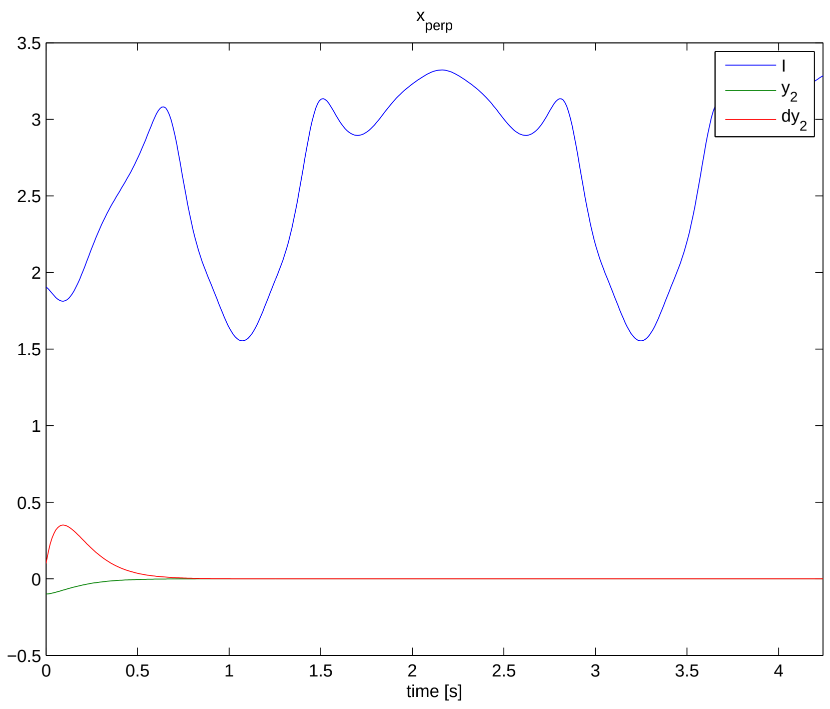
The system is then simulated for initial errors

$$\delta_\theta = -0.1 \tag{5.23}$$
$$\delta_{\dot\theta} = 0.1 \tag{5.24}$$
$$y_{02} = -0.1 \tag{5.25}$$
$$\dot y_{02} = 0.1. \tag{5.26}$$

The resulting phase portrait and the time-evolution of $x_\perp$ are shown in Figure 5.14. A plot of the time-evolution of q_2 is shown in Figure 5.15. We see that the error in the coordinate q_2 is asymptotically zeroed and q_2 converges to a periodic orbit, but I remains uncontrolled and the phase portrait of $[\theta; \dot\theta]$ remain away from the desired orbit.



(a) Phase plots of θ and q_2.



(b) Time evolution of $x_\perp$.

Figure 5.14: Simulations with an initial error to both y and $\dot{y}$.

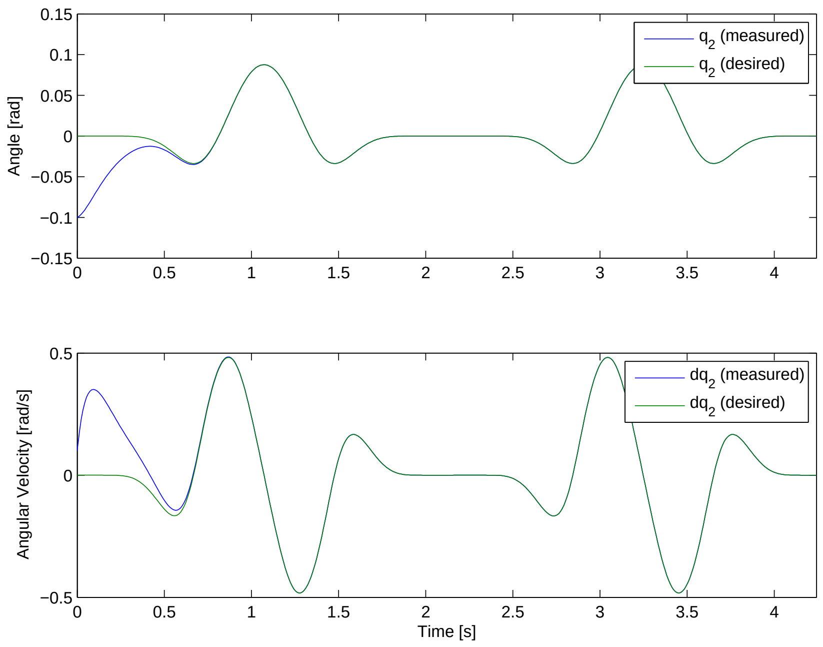


Figure 5.15: Time evolution of q_2 and $\dot{q}_2$.

5.2.3 Transverse Dynamics Controller

As in the inverted pendulum case, an attempt was made to find a controller that stabilizes the transverse dynamics. To do this a search for a controller that successfully stabilizes the linearized transverse dynamics was performed. However, no such controller was found. This might be due to a number of reasons.

- The number of optimization coefficients is very large. The matrix of control gains, $K(\tau)$ is a $(2n-1) \times (n-1)$ time-varying matrix function. If each of the elements in this matrix if chosen as a C^1-smooth Bézier curve with p control points, the total number of optimization coefficients is $(2n-1) \times (n-1) \times (p-2)$[1]. In our case the number of coordinates n equals 6, while we search for Bézier curves with 4 control points, causing the total number of optimization coefficients to be $11 \times 5 \times 2 = 110$. This is a huge search space, and it is expected that better results would be obtained by computing the matrix of control gains $K(\tau)$ in some way requiring less control points. One could attempt to restrict the possible choices of $K(\tau)$ in some way. A different approach would be to solve the periodic Riccati equation, as done in [14].
- The implemented controller is quite complex and consists of several modules. Any of these modules might contain some implementing error, small enough that it does not impact the stabilization of the inverted pendulum system, but large enough that the stabilization of the more complex monkey robot system fails.

[1]A Bézier curve with p control points that is C^1-smooth, requires that 2 control points are set as a function of the others, thus reducing the number of "free" control points for the curve to $p-2$.

Chapter 6

Conclusion and Further Work

In this thesis a MATLAB toolbox aiding the computation of a controller that stabilizes closed periodic orbits has been described. The partial feedback linearization module of the toolbox has been demonstrated, both through simulations where the auxiliary control input, $v(t)$ have been set to zero, and for cases where it is chosen such that the virtual holonomic constraints are asymptotically enforced.

Further, a controller stabilizing the full transverse dynamics has been demonstrated on the inverted pendulum system. Simulations from various challenging initial conditions show that the controller is indeed able to enforce convergence to the desired closed periodic orbit, even when the initial errors are large.

Further work is needed in order to show orbital stabilization of the desired brachiating trajectory in the monkey robot. A controller that enforces the virtual holonomic constraints have been demonstrated, but a but a controller that stabilizes the full transverse dynamics was not found. This might be due to the large amount of optimization coefficients used in the search for a stabilizing controller, and it is expected that better results might be obtained by reducing the number of optimization coefficients in some reasonable way.

6.1 Further Work

As we were unable to find a controller that stabilized the linear time-varying system of linearized transverse dynamics, this should be done in further work. Such a controller was found for the simpler inverted pendulum system, and combining this with the rest of the system lead to successful stabilization of the desired orbit. It is expected that the same will be the case for the monkey robot if a controller for the linearzied transverse dynamics is found. In order to find this one should attempt to reduce the number of optimization coefficients in the optimization problem.

This thesis have only been concerned with stabilizing a simplified brachiating motion, in which the monkey robot swings back and forth from one hand-hold. When a controller that successfully stabilizes this motion is found, it is suggested that the full brachiating motion, involving both the single-support phase and the double-support phase and the jumps that connect the two, should be stabilized. This is a more challenging task than stabilizing the continued single-support phase, as the full brachiating motion is a hybrid motion, thus requiring that the jumps between the phases are included in the linearization of the transverse dynamics.

Appendix A

Appendix

A.1 Solving the Reduced Dynamics

The reduced dynamics can be written as

$$\alpha(\theta)\ddot{\theta} + \beta(\theta)\dot{\theta}^2 + \gamma(\theta) = 0 \tag{A.1}$$

rearranging leads to

$$\ddot{\theta} + \frac{\beta(\theta)}{\alpha(\theta)}\dot{\theta}^2 + \frac{\gamma(\theta)}{\alpha(\theta)} = 0 \tag{A.2}$$

By defining $Y = \dot{\theta}^2$ and differentiating w.r.t. time we obtain

$$\begin{aligned} \frac{dY}{dt} &= 2\dot{\theta}\ddot{\theta} = \frac{dY}{d\theta}\dot{\theta} \\ \Rightarrow \ddot{\theta} &= \frac{1}{2}\frac{dY}{d\theta} \end{aligned} \tag{A.3}$$

substituting into (A.2) yields

$$\frac{dY}{d\theta} + p(\theta)Y = q(\theta) \tag{A.4}$$

where

$$\begin{aligned} p(\theta) &= \frac{2\beta(\theta)}{\alpha(\theta)} \\ q(\theta) &= -\frac{2\gamma(\theta)}{\alpha(\theta)} \end{aligned} \tag{A.5}$$

The homogeneous part of the solution is then found as follows

$$\begin{aligned}
\frac{dY}{d\theta} + p(\theta)Y &= 0 \\
\frac{1}{Y_h}\frac{dY_h}{d\theta} &= -p(\theta) \\
\ln|Y_h| - \ln|Y_0| &= -\int_{\theta_0}^{\theta} p(\tau)d\tau \\
Y_h &= \exp\left\{-\int_{\theta_0}^{\theta} p(\tau)d\tau\right\} Y_0 \\
&= \Psi_h(\theta,\theta_0)Y_0 \qquad \text{(A.6)}
\end{aligned}$$

The particular solution, Y_p can then be found by varying the constant Y_0, i.e.

$$Y_p = \Psi_h(\theta,\theta_0)z(\theta) \qquad \text{(A.7)}$$

Differentiating with respect to θ yields

$$\frac{dY_p}{d\theta} = -p(\theta)\Psi_h(\theta,\theta_0)z(\theta) + \Psi_h(\theta,\theta_0)\frac{dz}{d\theta} \qquad \text{(A.8)}$$

where we have used that

$$\frac{d\Psi_h}{d\theta} = -p(\theta)\Psi_h(\theta,\theta_0). \qquad \text{(A.9)}$$

Inserting (A.7) and (A.8) into (A.4) we obtain

$$\Psi_h(\theta,\theta_0)\frac{dz}{d\theta} + p(\theta)\Psi_h(\theta,\theta_0)z(\theta) - p(\theta)\Psi_h(\theta_0,\theta)z(\theta) = q(\theta) \qquad \text{(A.10)}$$

which simplifies to

$$\frac{dz}{d\theta} = \frac{q(\theta)}{\Psi_h(\theta,\theta_0)} \qquad \text{(A.11)}$$

which can be integrated to obtain an expression for $z(\theta)$

$$z(\theta) = \int_{\theta_0}^{\theta} \frac{q(s)}{\Psi_h(s,\theta_0)}ds \qquad \text{(A.12)}$$

The particular solution is then obtained by substituting this into (A.7)

$$\begin{aligned} Y_p &= \Psi_h(\theta,\theta_0)z(\theta) \\ &= \Psi_h(\theta,\theta_0)\int_{\theta_0}^{\theta}\frac{q(s)}{\Psi_h(s,\theta_0)}ds \\ &= \int_{\theta_0}^{\theta}\frac{\Psi_h(\theta,\theta_0)}{\Psi_h(s,\theta_0)}q(s)ds \\ &= \int_{\theta_0}^{\theta}\frac{\exp\left\{-\int_{\theta_0}^{\theta}p(\tau)d\tau\right\}}{\exp\left\{-\int_{\theta_0}^{s}p(\tau)d\tau\right\}}q(s)ds \\ &= \int_{\theta_0}^{\theta}\exp\left\{\int_{\theta_0}^{s}p(\tau)d\tau-\int_{\theta_0}^{\theta}p(\tau)d\tau\right\}q(s)ds \\ &= \int_{\theta_0}^{\theta}\exp\left\{-\int_{s}^{\theta}p(\tau)d\tau\right\}q(s)ds \\ &= \int_{\theta_0}^{\theta}\Psi_h(\theta,s)q(s)ds \\ &= -\int_{\theta_0}^{\theta}\Psi_h(\theta,s)\frac{2\gamma(s)}{\alpha(s)}ds \\ &= -\Psi_p(\theta,\theta_0) \end{aligned} \tag{A.13}$$

The solution of (A.4) can then be found by summing of the homogeneous and particular solutions

$$\begin{aligned} Y &= Y_h + Y_p \\ &= \Psi_h(\theta,\theta_0)Y_0 - \Psi_p(\theta,\theta_0) \\ &= \exp\left\{-\int_{\theta_0}^{\theta}p(\tau)d\tau\right\}Y_0 + \int_{\theta_0}^{\theta}\exp\left\{-\int_{s}^{\theta}p(\tau)d\tau\right\}q(s)ds \\ &= \exp\left\{-2\int_{\theta_0}^{\theta}\frac{\beta(\tau)}{\alpha(\tau)}d\tau\right\}Y_0 - \int_{\theta_0}^{\theta}\exp\left\{-2\int_{s}^{\theta}\frac{\beta(\tau)}{\alpha(\tau)}d\tau\right\}\frac{2\gamma(\theta)}{\alpha(\theta)}ds \end{aligned} \tag{A.14}$$

Now, by substituting $Y = \dot{\theta}^2$ and $Y_0 = \dot{\theta}_0^2$ we obtain the integral function

$$\begin{aligned} I(\theta,\dot{\theta},\theta_0,\dot{\theta}_0) &= \dot{\theta}^2 - \exp\left\{-2\int_{\theta_0}^{\theta}\frac{\beta(\tau)}{\alpha(\tau)}d\tau\right\}\dot{\theta}_0^2 \\ &\quad + \int_{\theta_0}^{\theta}\exp\left\{-2\int_{s}^{\theta}\frac{\beta(\tau)}{\alpha(\tau)}d\tau\right\}\frac{2\gamma(\theta)}{\alpha(\theta)}ds \\ &= \dot{\theta}^2 - \Psi_h(\theta,\theta_0)\dot{\theta}_0^2 + \Psi_p(\theta,\theta_0) \end{aligned} \tag{A.15}$$

A.2 Differentiating the Integral Function

This integral function can be differentiated with respect to time as follows.

$$\frac{\mathrm{d}}{\mathrm{dt}} I(\theta,\dot\theta,\theta_0,\dot\theta_0) = \frac{\partial I}{\partial \theta}\dot\theta + \frac{\partial I}{\partial \dot\theta}\ddot\theta \tag{A.16}$$

where

$$\begin{aligned}
\frac{\partial I}{\partial \dot\theta} &= 2\dot\theta \\
\frac{\partial I}{\partial \theta} &= -\frac{\partial}{\partial\theta}\Psi_h(\theta,\theta_0)\dot\theta_0^2 + \frac{\partial}{\partial\theta}\Psi_p(\theta,\theta_0) && \text{(A.17)} \\
\frac{\partial \Psi_h}{\partial \theta} &= -\frac{2\beta(\theta)}{\alpha(\theta)}\Psi_h(\theta,\theta_0) && \text{(A.18)} \\
\frac{\partial \Psi_p}{\partial \theta} &= \Psi_h(\theta,\theta)\frac{2\gamma(\theta)}{\alpha(\theta)} + \int_{\theta_0}^{\theta}\frac{\partial}{\partial\theta}\Psi_h(\theta,s)\frac{2\gamma(s)}{\alpha(s)}ds \\
&= \frac{2\gamma(\theta)}{\alpha(\theta)} - \frac{2\beta(\theta)}{\alpha(\theta)}\int_{\theta_0}^{\theta}\Psi_h(\theta,s)\frac{2\gamma(s)}{\alpha(s)}ds \\
&= \frac{2}{\alpha(\theta)}\Big(\gamma(\theta) - \beta(\theta)\Psi_p(\theta,\theta_0)\Big) && \text{(A.19)}
\end{aligned}$$

inserting (A.18) and (A.19) into (A.17) leads to

$$\begin{aligned}
\frac{\partial I}{\partial \theta} &= \frac{2\beta(\theta)}{\alpha(\theta)}\Psi_h(\theta,\theta_0)\dot\theta_0^2 + \frac{2\gamma(\theta)}{\alpha(\theta)} - \frac{2\beta(\theta)}{\alpha(\theta)}\Psi_p(\theta,\theta_0) \\
&= \frac{2}{\alpha(\theta)}\Big[\gamma(\theta) + \beta(\theta)\Big(\Psi_h(\theta,\theta_0)\dot\theta_0^2 - \Psi_p(\theta,\theta_0)\Big)\Big] \\
&= \frac{2}{\alpha(\theta)}\Big[\gamma(\theta) + \beta(\theta)\Big(\dot\theta^2 - I(\theta,\dot\theta,\theta_0,\dot\theta_0)\Big)\Big] && \text{(A.20)}
\end{aligned}$$

and the time-derivative of the integral function is

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{dt}} I(\theta,\dot\theta,\theta_0,\dot\theta_0) &= \frac{\partial I}{\partial \theta}\dot\theta + \frac{\partial I}{\partial \dot\theta}\ddot\theta \\
&= \frac{2}{\alpha(\theta)}\Big[\gamma(\theta) + \beta(\theta)\Big(\dot\theta^2 - I(\theta,\dot\theta,\theta_0,\dot\theta_0)\Big)\Big]\dot\theta + 2\dot\theta\ddot\theta \\
&= \frac{2\dot\theta}{\alpha(\theta)}\Big[\gamma(\theta) + \beta(\theta)\Big(\dot\theta^2 - I(\theta,\dot\theta,\theta_0,\dot\theta_0)\Big)\Big] \\
&\qquad + 2\dot\theta\Big[\frac{g(\theta,\dot\theta,y,\dot y,v) - \beta(\theta)\dot\theta^2 - \gamma(\theta)}{\alpha(\theta)}\Big] \\
&= \frac{2\dot\theta}{\alpha(\theta)}\Big[g(\theta,\dot\theta,\ddot\theta,y,\dot y,v) - \beta(\theta)I(\theta,\dot\theta,\theta_0,\dot\theta_0)\Big] && \text{(A.21)}
\end{aligned}$$

Bibliography

[1] Stian H. Askeland. Elements of trajectory generation in brachiation of a monkey robot. Project Report, December 2011.

[2] S. Bittanti, P. Colaneri, and G. Guardabassi. Periodic solutions of periodic riccati equations. *Automatic Control, IEEE Transactions on*, 29(7):665 – 667, jul 1984.

[3] C.T. Chen. *Linear System Theory and Design*. The Oxford Series In Electrical And Computer Engineering. Oxford University Press, 1999.

[4] C. Chevallereau, G. Abba, Y. Aoustin, F. Plestan, E.R. Westervelt, C. Canudas-De-Wit, and J.W. Grizzle. Rabbit: a testbed for advanced control theory. *Control Systems, IEEE*, 23(5):57 – 79, oct. 2003.

[5] T. Fukuda, Y. Hasegawa, M. Doi, and Y. Asano. Multi-locomotion robot - energy-based motion control for dexterous brachiation -. In *Robotics and Biomimetics (ROBIO). 2005 IEEE International Conference on*, pages 4 –9, 0-0 2005.

[6] T. Fukuda, F. Saito, and F. Arai. A study on the brachiation type of mobile robot (heuristic creation of driving input and control using cmac). In *Intelligent Robots and Systems '91. 'Intelligence for Mechanical Systems, Proceedings IROS '91. IEEE/RSJ International Workshop on*, pages 478 –483 vol.2, nov 1991.

[7] Encyclopedia Britannica Inc. Brachiation, December 2011. `http://www.britannica.com/EBchecked/topic/76645/brachiation`.

[8] H.K. Khalil. *Nonlinear Systems*. Prentice Hall, 2002.

[9] Uwe Mettin. *Principles for planning and analyzing motions of underactuated mechanical systems and redundant manipulators*. PhD thesis, Ume University, Department of Applied Physics and Electronics, 2009.

[10] J. Nakanishi, T. Fukuda, and D.E. Koditschek. A brachiating robot controller. *Robotics and Automation, IEEE Transactions on*, 16(2):109 –123, apr 2000.

[11] F. Saito, T. Fukuda, and F. Arai. Swing and locomotion control for two-link brachiation robot. In *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pages 719 –724 vol.2, may 1993.

[12] T. Sasagawa. A necessary and sufficient condition for the solution of the riccati equation to be periodic. *Automatic Control, IEEE Transactions on*, 25(3):564 – 566, jun 1980.

[13] A. Shiriaev, J. Perram, A. Robertsson, and A. Sandberg. Explicit formulas for general integrals of motion for a class of mechanical systems subject to virtual constraints. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 2, pages 1158 – 1163 Vol.2, dec. 2004.

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

[15] A.S. Shiriaev, L.B. Freidovich, and S.V. Gusev. Transverse linearization for controlled mechanical systems with several passive degrees of freedom. *Automatic Control, IEEE Transactions on*, 55(4):893 –906, april 2010.

[16] M.W. Spong, S. Hutchinson, and M. Vidyasagar. *Robot modeling and control*. John Wiley & Sons, 2006.

[17] James R. Usherwood and John E. A. Bertram. Understanding brachiation: insight from a collisional perspective. *Journal of Experimental Biology*, 206(10):1631–1642, 2003.