Trajectory Tracking for Mobile Manipulators
A Comparison of Control Laws and Optimization of the Reachable Workspace
Degree Project in Engineering Physics
SA104X

May 19, 2012

Authors:
Mikael Brudfors, Clara Lynn & Robert Rogersten

Supervisors:
Xiaoming Hu & Johan Markdahl

Department of Mathematics, Division of Optimization and Systems Theory
KTH Royal Institute of Technology
Stockholm, Sweden
Abstract

This report investigates how mobile manipulators can be controlled when an end-effector is required to follow some certain trajectory. The robot platform is of type (2,0) and is restricted to planar motion under non-holonomic constraints. The manipulator of the robot are modeled as a series of chains, connected by revolute joints. Mathematical theory is presented for both the maximum working range and the reachable workspace of the manipulator. Three different control laws; nonlinear kinematic control with linear and nonlinear feedback and input/output linearization are derived. The performance of these control laws are compared in simulations by analysis of their trajectory tracking errors and their stability. Finally, each control laws pros and cons are presented.
# Contents

1 Introduction 3
   1.1 Motivation ............................................. 3
   1.2 Problem Statement ................................... 3

2 The Platform 5
   2.1 Introduction ........................................... 5
   2.2 Kinematic Constraints ................................ 5
   2.3 Configuration ......................................... 6
   2.4 Flatness ............................................... 7

3 The Manipulator 9
   3.1 Introduction ........................................... 9
   3.2 Joint Angles at Maximum Working Range ............... 9
   3.3 Reachable Workspace .................................. 12

4 Trajectory Planning 18
   4.1 Introduction ........................................... 18
   4.2 Path and Timing Law ................................... 18
   4.3 Planning via Cartesian Polynomials .................... 19

5 Trajectory Tracking 21
   5.1 Introduction ........................................... 21
   5.2 Linear/Nonlinear Dynamic System ..................... 22
   5.3 Input/Output Linearization ........................... 23

6 Simulations 25
   6.1 Introduction ........................................... 25
   6.2 Circular Trajectory .................................... 26
   6.3 Polynomial Trajectory ................................ 30
   6.4 Rectilinear Trajectory ................................. 34

7 Analysis and Conclusion 37
   7.1 Introduction ........................................... 37
   7.2 Unstability ............................................. 37
Chapter 1

Introduction

1.1 Motivation

Men have dreamed of robots for thousands of years. Robots who can free us from toil, robots who can fill our emotional needs, robots who can act and think in our place. In our industrial society, stationary manipulators have been widely implemented, in construction factories, food production and many other areas, where their ability to reliably accomplish dull and repetitive tasks is highly valued. However, as production moves away from conveyor belts and becomes more complex, the need for non-stationary manipulators grows more and more urgent.

The kinematic skeleton of a robot, the manipulator, is modeled as a series of links connected by either revolute or sliding joints forming a serial chain. This skeleton has two basic forms, that of a single serial chain connected to an end-effector called a serial robot and as a set of serial chains supporting a single end-effector, called a parallel robot. To make the robot mobile this kinematic skeleton is connected to some kind of platform.

1.2 Problem Statement

This report will consider a manipulator that can work in a two-dimensional plane, mounted on a platform able to move in the direction it is facing. The manipulator is built up by a serial chain of links, connected by joints that provide single-axis rotation, with the first link connected to the platform and the last one to the end-effector. This robot model can be seen in Figure 1.1. The manipulator and the platform will be modeled separately, with the aim to connect these two models in simulations.

There will be focus on how manipulators on mobile platforms can be coordinated, when the end-effector of the manipulator is required to follow some certain trajectory. Three different control laws will be considered, and compared by analyzing their stability and convergence rate. The control
laws which will be explored are input/output linearization and nonlinear kinematic control with linear and nonlinear feedback.

Figure 1.1: Figure of the robot model used in this report. The manipulator will be built up by an arbitrary number of revolute joints and connected to the platform center. At the end of the manipulator is the end-effector.
Chapter 2

The Platform

2.1 Introduction

In this chapter, the platform of a wheeled robot will be considered. There are several classes of wheeled robots, and they all have certain constraints on their configurations and motion. This report explores the properties of the class (2,0) robot platform [1]. The (2,0) robot has no steering wheels, but it has either one or several fixed wheels with a common axle. A typical example of a class (2,0) vehicle is a wheelchair.

By introducing concepts such as holonomic, nonholonomic, and differential flatness, it shall be seen that this type of robots has several useful qualities.

2.2 Kinematic Constraints

Consider a system described in generalized coordinates\(^1\) by a configuration \(q \in C\), where \(C\) denotes the space of all robot configurations, i.e., the configuration space. The platform has the configuration \(\mathbb{R}^2 \times SO(2)\) and the end-effector has the configuration \(\mathbb{R}^2 \times SO(2)^m\) which in total gives the robot configurations \(C\) discussed in this thesis as \(\mathbb{R}^2 \times SO(2)^{m+1}\), where \(m\) is the number of revolute joints of the manipulator and \(SO(2)\) denotes the special orthonormal group.\(^2\)

The system may have certain constraints. If the constraints can be put on the form

\[ h_i(q) = 0, \quad i = 1, 2, ..., k < n, \quad (2.1) \]

\(^1\)A restriction for a set of coordinates to serve as generalized coordinates is that they should uniquely define any possible configuration of the system relative to the reference configuration [2].

\(^2\)The orientation of the rigid body can be changed with a rotation matrix which belongs to the special orthonormal group \(SO(n)\) of the real \(n \times n\) matrices with orthonormal columns and determinant equal to 1 [3]. In the case of planar rotations \(n = 2\).
then they are called holonomic constraints and are constraints on the accessible configuration, i.e., there are initial conditions from which the system cannot go to certain other configurations. Therefore, holonomic constraints implies a reduction in freedom of motion, which usually is not a good thing.

If the constraints involve both generalized coordinates and their velocities, they are called kinematic:

\[ a_i(q, \dot{q}) = 0, \quad i = 1, 2, ..., k < n, \]  

(2.2)

which means at each point of the configuration, there are constraints on the direction of the robot’s motion. If the kinematic constraints are expressed as

\[ A^T(q)\dot{q} = 0, \]  

(2.3)

then they are on Pfaffian form, consequently linear in the generalized velocities. The columns of \( A \) are supposed to be smooth and linearly independent.

The existence of \( k \) holonomic constraints implies the existence of \( k \) number of kinematic constraints

\[ \dot{h}_i(q(t)) = \nabla h_i(q) \cdot \dot{q} = 0, \quad i = 1, 2, ..., k. \]  

(2.4)

However, (2.3) may or may not be integrable to (2.1). If it is not, the kinematic constraint is said to be nonholonomic. A nonholonomic constraint is a constraint on velocity, which means there are directions the robot cannot go instantaneously, but in time the robot can still get to any position.

If the matrix \( A \) in (2.3) has \( k \) columns, then the system has \( k \) Pfaffian constraints. This entails that \( \dot{q} \) at each configuration \( q \) belongs to the \((n - k)\)-dimensional null space of the matrix \( A^T(q) \). If \( \mathcal{N}(A^T) = \text{span}\{z_1(q), ..., z_n(q)\} \), the generalized velocity can be expressed as:

\[ \dot{q} = \sum_{i=1}^{n} k_i z_i(q), \]  

(2.5)

where \( k_i, i = 1, ..., n \) are constants.

### 2.3 Configuration

The configuration of the \((2,0)\) robot is completely described in generalized coordinates by:

\[ q = [x \ y \ \theta]^T, \]  

(2.6)

where \( q \in C \) and the configuration space \( C \) for the robot is \( \mathbb{R}^2 \times SO(2) \). The Cartesian coordinates \((x, y)\) are the position of the center of the robot, and \( \theta \) is the orientation of this center with respect to the \( x \) axis (see Figure 2.1).
2.4 Flatness

Differential flatness applies to the type (2,0) robot and is relevant in planning problems. The definition of differential flatness is that a nonlinear dynamic system \( \dot{q} = f(q) + G(q)u \) is differentially flat if there exists a set of outputs \( y \), called flat outputs, such that the state \( q \) and the control inputs \( u \) can be expressed algebraically as a function of these flat outputs and its time derivatives up to a certain order. In the type (2,0) robot case \( f(q) = 0 \), and its Cartesian coordinates \( y = (x(t), y(t)) \) can be considered flat outputs, i.e., the third configuration coordinate \( \theta \) and the control input \( u = [v \ \omega]^T \) can be expressed using the Cartesian coordinates of the first and the second time

\[
\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta \\ \sin \theta \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega, \tag{2.7}
\]

where \( v \) and \( \omega \) are available control inputs and represent the constants\(^3\) in (2.5). For the robot to satisfy a Pfaffian constraint, there has to exist a matrix \( A^T \) such that the vectors in (2.7) belong to its null space. By inspection, it is possible to see that \( A^T = [\sin \theta \ - \cos \theta \ 0] \) satisfy this condition. Hence, the Pfaffian constraint for the robot is expressed as

\[
A^T \dot{q} = \dot{x} \sin \theta - \dot{y} \cos \theta = 0. \tag{2.8}
\]

It can be seen from (2.8) that the velocity of the contact point is zero in the direction orthogonal to the sagittal axis\(^4\) of the vehicle.

\(^3\)The control inputs \( v \) and \( \omega \) can here be interpreted as the linear and angular velocities for the robot.

\(^4\)The axis which passes from rear to front through the robot in the direction of motion.
2.4. FLATNESS

derivatives

\[ \theta(t) = \arctan[2(\dot{y}(t), \dot{x}(t))] + k\pi, \quad (2.9) \]
\[ v(t) = \pm \sqrt{(\ddot{x}(t))^2 + (\ddot{y}(t))^2}, \quad (2.10) \]
\[ \omega(t) = \frac{\ddot{y}(t)\dot{x}(t) - \ddot{x}(t)\dot{y}(t)}{(\dot{x}(t))^2 + (\dot{y}(t))^2}, \quad (2.11) \]

where \( k = 0 \) is forward motion and \( k = 1 \) is backward motion. In the general case the definition of differential flatness can be expressed algebraically as:

\[ q = q(y, \dot{y}, \ldots, y^n), \]
\[ u = u(y, \dot{y}, \ldots, y^n). \quad (2.12) \]
Chapter 3

The Manipulator

3.1 Introduction

In this chapter, the manipulator attached to a mobile platform will be considered. Conditions to enable the end-effector to reach as far as possible and to be able to reach all the points between the farthest point and the point where the manipulator is joined to the platform will be analyzed. Several special cases will be considered before the general ones, and the conclusions made will be most useful in the subsequent chapters.

3.2 Joint Angles at Maximum Working Range

A manipulator can schematically be represented from a mechanical viewpoint as a kinematic chain of rigid bodies (links) connected by joints, i.e., a robot arm. One end of the chain is constrained to a base, while the other, the end-effector, is free to perform tasks required of the robot. The position of the manipulator end-effector relative to the platform is described by the function $f(\theta)$.

Any stationary manipulator has a given maximum working range relative to the platform, which the end-effector is able to reach. Suppose that the manipulator is planar and consisting of $m$ links with lengths $l_1, \ldots, l_m$ connected by revolute joints with inter mutual angles $\theta = [\theta_1 \ldots \theta_m]^T$. Then the angle that results in the maximum working range can be determined. Intuitively this must be at $\hat{\theta} = [\theta_1 \ 0 \ldots 0]^T \in \mathbb{R}^m$ were $0 \leq \theta_1 < 2\pi$. The following sections shows that this is the actual case by first presenting a proof for a two-link planar arm and then moving on to the general $m$-link planar arm.
Two-Link Planar Arm

An example of the two-link planar arm can be seen in Figure 3.1. The position of the manipulator end-effector can be described by:

\[ f(\theta) = [l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2), l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2)] , \quad 0 \leq \theta < 2\pi. \]

The angles that result in the maximum working range for the two-link planar arm are stated in Proposition 1:

\[ \| f(\theta) \| \text{ has its maximum iff } \theta_2 = 0. \]

**Proof.**

\[ f(\theta)^T f(\theta) = \| f(\theta) \|^2 = \]

\[ [l_1 l_1 + l_2 l_2]^T [l_1 l_1 + l_2 l_2] = l_1^2 + 2l_1 l_2 l_1 + l_2^2 = \]

\[ l_1^2 + l_2^2 + 2l_1 l_2 (\cos(\theta_1) \cos(\theta_1 + \theta_2) + \sin(\theta_1) \sin(\theta_1 + \theta_2)) = \]

\[ l_1^2 + l_2^2 + 2l_1 l_2 (\cos(\theta_1) \cos(\theta_1) \cos(\theta_2) - \sin(\theta_1) \sin(\theta_2) + \sin(\theta_1) \sin(\theta_1) \cos(\theta_2) - \cos(\theta_1) \sin(\theta_2)) = \]

\[ l_1^2 + l_2^2 + 2l_1 l_2 (\cos^2(\theta_1) \cos(\theta_2) - \cos(\theta_1) \sin(\theta_1) \sin(\theta_2) + \sin^2(\theta_1) \cos(\theta_2)) + \sin(\theta_1) \cos(\theta_1) \sin(\theta_2) = \]

\[ l_1^2 + l_2^2 + 2l_1 l_2 \cos(\theta_2). \]

From this it is obvious that \( \| f(\theta) \| \) has its maximum if and only if \( \theta_2 = 0. \) \( \Box \)
3.2. JOINT ANGLES AT MAXIMUM WORKING RANGE

General Case

For the general case the position of the manipulator end-effector can be described by:

\[ f(\theta) = \sum_{k=1}^{m} l_k \left[ \begin{array}{c} \cos \left( \sum_{j=1}^{k} \theta_j \right) \\ \sin \left( \sum_{j=1}^{k} \theta_j \right) \end{array} \right], \quad 0 \leq \theta < 2\pi. \]

The angles that results in the maximum working range for the \( m \)-link planar arm are stated in Proposition 2:

**Proposition 2.** \( \| f(\theta) \| \) has its maximum iff \( \hat{\theta} = [\theta_1 \ 0 \ ... \ 0]^T \in \mathbb{R}^m \).

**Proof.**

\[ \| f(\theta) \|^2 = f(\theta)^T f(\theta) = \]

\[ \sum_{i=1}^{m} l_i \left[ \cos \left( \sum_{l=1}^{i} \theta_l \right) \sin \left( \sum_{l=1}^{i} \theta_l \right) \right] \sum_{k=1}^{m} l_k \left[ \begin{array}{c} \cos \left( \sum_{j=1}^{k} \theta_j \right) \\ \sin \left( \sum_{j=1}^{k} \theta_j \right) \end{array} \right] = \]

\[ \sum_{i=1}^{m} \sum_{k=1}^{m} l_i l_k \left[ \cos \left( \sum_{l=1}^{i} \theta_l \right) \cos \left( \sum_{j=1}^{k} \theta_j \right) + \sin \left( \sum_{l=1}^{i} \theta_l \right) \sin \left( \sum_{j=1}^{k} \theta_j \right) \right] = \]

\[ \sum_{i=1}^{m} \sum_{k=1}^{m} l_i l_k \cos \left( \sum_{l=1}^{i} \theta_l - \sum_{j=1}^{k} \theta_j \right). \]

It can be seen from this that \( \text{arg max} (\| f(\theta) \|^2) = \hat{\theta} \). Notice that \( \theta_1 \) is not going to be a part of the expression because it is canceled in the sum. \( \square \)
3.3 Reachable Workspace

The reachable workspace for the manipulator is the space in which the origin of the end-effector frame can be placed with at least one orientation. The reachable workspace is denoted as \( S \subset \mathbb{R}^2 \), and the rotations available at each of these points form a dexterous workspace [1]. As proved in the previous section, the maximum working range of any configuration consisting of \( m \) revolute joints is \( \sum_{i=1}^{m} l_i = L \), as seen in Figure 3.2.

![Figure 3.2: Example of a manipulator with \( m = 5 \). This gives a maximum working range \( \sum_{i=1}^{5} l_i = L \) when \( \theta = \hat{\theta} \).](image)

This means that the points given by \( f(\theta) \), the position of the end-effector relative to the platform, must be on the border or inside the circle with radius \( L \) consisting of these revolute joints. For the end-effector to be able to reach all the points inside a circle with radius \( L \), the length of the links should not be chosen arbitrarily. The subsequent sections will describe the restriction imposed on link length using geometry and logical reasoning by first considering relatively simple manipulator configurations, and then a general case with \( m \) joints. To do so, first, consider the following proposition:

**Proposition 3.** Given a point \( p \) in the robot’s coordinate system where \( \|p\| \leq \|f(\hat{\theta})\| \), if \( \exists \theta_0 \) such that \( f(\theta_0) = 0 \), then \( \exists \theta \) such that \( f(\theta) = p \).

**Proof.** If \( \theta = \hat{\theta} = [\theta_1 \ 0 \ ... \ 0] \in \mathbb{R}^m \) then \( \|f(\theta)\| = L \), as shown in the previous section. If \( \exists \theta_0 \) such that \( f(\theta_0) = 0 \), then all the points along a path between \( f(\hat{\theta}) \) and \( f(\theta_0) \) can be reached, because there must exist a path connecting these points. If \( \theta_1 \) is allowed to range from 0 to \( 2\pi \), this path can be rotated by \( 2\pi \), and hence, all the points \( p \) where \( \|p\| \leq \|f(\theta)\| \) can be reached, i.e., there \( \exists \theta \) such that \( f(\theta) = p \). \( \square \)

The robot’s position, i.e, the origin of the robot’s coordinate system in a fixed coordinate system is denoted as \( p_0 \) in the subsequent text. A point \( p \) as in Proposition 3 can then be described in the fixed coordinate system as \( p_0 + p \). Figure 3.3 shows the manipulator from Figure 3.2 with a reachable workspace of \( S = \{p \in \mathbb{R}^2 : \|p\| \leq \|f(\hat{\theta})\|\} \).
3.3. REACHABLE WORKSPACE

Figure 3.3: Here $\theta = \theta_0$, and hence the end-effector can move from the border of the circle to the origin. Therefore, it is possible to reach all points $p$ that satisfy $\|p\| \leq \|f(\theta)\|$.

One- and Two-Link Planar Arm

What restrictions should be imposed on the link length? The simplest case that can be considered is obviously the one-link planar arm. In this case there is no $\theta_0$ such that $f(\theta_0) = 0$. Therefore, the reachable workspace is the border of the circle with radius $l_1 = L$. For the two-link planar arm, Figure 3.4 shows two cases $l_1 < l_2$ and $l_1 > l_2$. Here too, there is no $\theta_0$ such that $f(\theta_0) = 0$, and the closest distance to the origin is $|l_1 - l_2|$, which is a result of setting $\theta_2 = \pi$. The gray area in each subfigure indicates the reachable workspace acquired by allowing all components in $\theta$ to assume all their possible values. It is geometrically clear that to make it possible to find $\theta_0$ such that $f(\theta_0) = 0$, $l_1$ should equal $l_2$ as seen in Figure 3.5.

Figure 3.4: Upper row shows $l_1 < l_2$ and the lower row is $l_1 > l_2$. The grey area is the reachable workspace.
3.3. REACHABLE WORKSPACE

Three-Link Planar Arm

This case is slightly more complex, since it is not obvious from start which configurations maximizes the reachable workspace. To start with, one easily realizes, that if $l_1 = l_2 = l_3$ then the reachable workspace is $S = \{ p \in \mathbb{R}^2 : \| p \| \leq \| f(\theta) \| \}$, since there exists a $\theta_0$ such that $f(\theta_0) = 0$. This can be achieved by choosing $\theta_2 = \pi/3$, $\theta_3 = \pi/3$ and $\theta_1$ arbitrary. The following relations also makes it possible to find $\theta_0$ such that $f(\theta_0) = 0$:

$$\begin{align*}
l_1 + l_2 &= l_3, \\
l_3 + l_2 &= l_1, \\
l_1 + l_3 &= l_2.
\end{align*} \quad (3.1)$$

Possible solutions for (3.1) to $f(\theta_0) = 0$ are $\theta_0 = [\theta_1 \ 0 \ \pi]^T$, $\theta_0 = [\theta_1 \ \pi \ 0]^T$ and $\theta_0 = [\theta_1 \ \pi \ \pi]^T$, respectively. Prior to presenting the most general restriction on the arm lengths for the three link case, the following lemmas\(^1\) needs to be stated:

**Lemma 1.** (Triangle inequality) In a triangle the length of any side is less than the sum of the two other sides.

**Lemma 2.** If $p_0$, $p_1$ and $p_2$ are distinct point in $\mathbb{R}^2$, then $\| p_2 - p_0 \| = \| p_2 - p_1 \| + \| p_1 - p_0 \|$, if and only if the 3 points are collinear and $p_1$ is between $p_0$ and $p_2$.

In the following propositions in this chapter the point $p_0$ denotes the robot’s position in the fixed coordinate system, $p_1$ denotes the end-point of link 1 in the fixed coordinate system, and $p_2$ denotes the end-point of link 2 and so forth. The function for the end-effector position $f(\theta)$ is still a function which describes a point relative to the platform in the moving coordinate system, i.e., the end-effector position in the fixed coordinate system is described by $p_0 + f(\theta) \oplus$. The most general restriction on the arms length can then be stated as a proposition.

**Proposition 4.** Suppose $\exists \theta_0$ such that $f(\theta_0) = 0$, this is possible if and only if $l_j \leq l_i$, $l_k \leq l_i$ and $l_i \leq (l_j + l_k)$, $i,j,k \in \{1,2,3\}$, with i, j, k being different numbers.

\(^1\)For proofs of Lemma 1 and 2 see [4].
3.3. REACHABLE WORKSPACE

Proof. Suppose $\exists \theta_0$ such that $f(\theta_0) = 0$, i.e., $l_1 = \|p_1 - p_0\|, l_2 = \|p_2 - p_1\|$ and $l_3 = \|p_2 - p_0\|$, where $p_0, p_1, p_2 \in \mathbb{R}^2$. Subsequently suppose without loss of generality that $l_1 \leq l_3$ and $l_2 \leq l_3$. There exists two possible cases and one impossible case for the links to fulfill such a geometry:

- The points are collinear, hence it follows from Lemma 2 that $l_3 = l_2 + l_1$.
- The points are non-collinear and $l_3 < l_2 + l_1$, hence it follows from Lemma 1 that the links form a triangle.
- The case when $l_3 > l_2 + l_1$ is impossible, i.e., there does not exist a $\theta_0$ when this is the case.

If for example, $l_3$ is the longest link and $l_3 > l_2 + l_1$. Then the reachable workspace becomes $S = \{p \in \mathbb{R}^2 : r_i \leq \|p\| \leq \|f(\hat{\theta})\|\}$ for some inner radius $r_i$ which depends on the links lengths. In the two-link case the inner radius is $r_i = |l_1 - l_2|$. Figure 3.4 then shows a graphical representation of $S$. 

\[ l_1 \]
\[ l_2 \]
\[ l_3 \]
\[ \theta_0 = [\theta_1, 0, \pi] \implies f(\theta_0) = 0 \]
\[ l_3 = l_1 + l_2 \]
\[ l_3 < l_1 + l_2 \]
\[ l_3 > l_1 + l_2 \]
\[ \theta = [\theta_1, 0, \pi] \implies f(\theta) \neq 0 \]

Figure 3.6: Examples of the collinear, non-collinear and impossible case, respectively.
3.3. REACHABLE WORKSPACE

There exists two possible cases and one impossible case for the links to fulfil the following reasoning: Suppose, without loss of generality, that \( l_1 \) is the longest link. Consider the points \( p_0, p_1 \) and \( p_m \). Then from Lemma 1 and 2 does not motivate the two possible cases for an arbitrary number of links, but the reader can convince oneself that this holds with the following reasoning: Suppose, without loss of generality, that \( l_{m+1} = \|p_0 - p_m\| \) is the longest link. Consider the points \( p_0, p_1 \) and \( p_m \). Then from

General Case

For the general \( n \)-link case the following problem can be formulated: Given a point \( p \) in the robot’s coordinate system where \( \|p\| \leq \|f(\theta)\| \) and is there a \( \theta \) such that \( f(\theta) = p \)? The following proposition will answer that question.

**Proposition 5.** Suppose \( \exists \theta_0 \) such that \( f(\theta_0) = 0 \), this is possible if and only if there is an \( i \) such that \( l_i \) is greater or equal to any of the other \( m \) links and \( l_i \leq (l_1 + \ldots + l_{i-1} + l_{i+1} + \ldots + l_{m+1}) \).

**Proof.** Suppose \( \exists \theta_0 \) such that \( f(\theta_0) = 0 \), i.e., consider \( m + 1 \) points \( p_0, \ldots, p_m \in \mathbb{R}^2 \) and \( m + 1 \) links defined by \( l_j \) for \( j = 1, \ldots, m \), the last one defined by \( l_{m+1} = p_0 - p_m \). Subsequently suppose \( l_i \) is greater or equal to any of the other lengths \( l_k \) for \( k = 1, \ldots, i-1, i+1, \ldots, m+1 \). There exists two possible cases and one impossible case for the links to fulfil such a geometry:

- The points are collinear, and hence it follows that \( l_i = l_1 + \ldots + l_{i-1} + l_{i+1} + \ldots + l_m \).
- The points are non-collinear, and hence it follows that \( l_i < l_1 + \ldots + l_{i-1} + l_{i+1} + \ldots + l_m \).
- Impossible case \( l_i > l_1 + \ldots + l_{i-1} + l_{i+1} + \ldots + l_m \).

Figure 3.7: Different cases using three links. The lower row shows configurations such that \( S = \{ p \in \mathbb{R}^2 : \|p\| \leq \|f(\theta)\| \} \) and the upper row shows configurations such that \( S = \{ p \in \mathbb{R}^2 : r_1 \leq \|p\| \leq \|f(\theta)\| \} \) for some inner radius \( r_1 \) which depends on the links.
Lemma 1 it holds that $\|p_0 - p_m\| \leq \|p_0 - p_1\| + \|p_1 - p_m\| \iff l_{m+1} \leq l_1 + \bar{l}_{1m}$, where $\bar{l}_{1m}$ denotes the length between $p_1$ and $p_m$. Subsequently for the points $p_1$, $p_2$ and $p_m$, it holds that $\|p_1 - p_m\| \leq \|p_1 - p_2\| + \|p_2 - p_m\|$ which implies $l_{m+1} \leq l_1 + l_2 + \bar{l}_{2m}$. For the points $p_2$, $p_3$ and $p_m$, it follows that $l_{m+1} \leq l_1 + l_2 + l_3 + \bar{l}_{3m}$ and so forth. It is possible to continue until $\bar{l}_{(m-1)m} = l_m$ which implies $l_{m+1} \leq l_1 + ... + l_m$, with equality if and only if all the points are collinear.
4.1 Introduction

In this chapter the problem of planning a permitted trajectory for the robot to track will be introduced, that is, how to design a path in space with corresponding time constraints, which the robot is able to track asymptotically. A parameterization of the path will be introduced, and bearing in mind the constraints found in chapter 2, one, out of many, geometrically permissible paths will be defined.

4.2 Path and Timing Law

Two different types of problems are path following and trajectory tracking. Path following is to follow a certain path in space, without time constraints. Trajectory tracking on the other hand is once again to follow a certain path in space, but this time asymptotically and under time constraints \( (x_d(t), y_d(t)) \), starting from an initial configuration \( q_0 = [x_0, y_0, \theta_0]^T \) that may or may not be matched with the trajectory. This report will focus on the latter.

The problem of planning a trajectory for a robot can be divided into finding a path and defining a timing law on the path. Because the type (2,0) robot is subject to nonholonomic constraints, the first of these two problems becomes more difficult, since the path must fulfill the nonholonomic constraints at all points, in addition to meeting the boundary conditions.

A trajectory \( q(t) \), \( t \in [t_i, t_f] \), can be broken down into a geometric path \( q(s) \) and a timing law \( s = s(t) \). This space-time separation implies through the chain rule that

\[
\dot{q}(s(t)) = \frac{dq}{ds} \dot{s} = q' \dot{s}.
\]  

(4.1)

Equation (2.3) can then be rewritten as

\[
A^T(q)q' = 0 \quad \text{if} \quad \dot{s} > 0,
\]  

(4.2)
and geometrically admissible paths can be explicitly defined as the solutions of the nonlinear system

\[ q' = Z(q)\tilde{u}, \]  

where \( \tilde{u} \) are geometric inputs that are related to the velocity inputs \( u \) by

\[ u(t) = \tilde{u}(s)\dot{s}(t), \]

and the columns of \( Z(q) \) form a basis for \( \mathcal{N}(A^T(q)) \), the null space of \( A \). Using the Pfaffian constraints in (2.8) the geometrically permissible paths for the robot is given by:

\[ x' = \tilde{v}\cos\theta, \]
\[ y' = \tilde{v}\sin\theta, \]
\[ \theta' = \tilde{\omega}. \]  

Note that \( x' \) and \( y' \) are directed so that they are the tangent to the Cartesian path. Using this property equation (2.9), (2.10) and (2.11) can be expressed as

\[ \theta(s) = \arctan[2(y'(s), x'(s))] + k\pi, \]
\[ \tilde{v}(s) = \pm\sqrt{(x'(s))^2 + (y'(s))^2}, \]
\[ \tilde{\omega}(s) = \frac{y''(s)x'(s) - x''(s)y'(s)}{(x'(s))^2 + (y'(s))^2}. \]  

### 4.3 Planning via Cartesian Polynomials

From the previous section it could be seen that the type (2,0) robot admits a set of flat outputs, because \( q(s) = [x(s) y(s) \theta(s)]^T \). This property may be used to solve planning problems in an efficient manner using interpolation. The interpolated path will then satisfy the boundary conditions, and the resulting configuration space will satisfy the nonholonomic constraints.

The problem of planning a path for the type (2,0) robot, from an initial configuration \( q(s_i) = q_i = [x_i y_i \theta_i]^T \) to a final configuration \( q(s_f) = q_f = [x_f y_f \theta_f]^T \), should consequently be solved by some interpolating scheme. There are a multiple of these, but in this report a planning method known as planning via Cartesian polynomials will be used. This method interpolates the flat outputs \( x \) and \( y \)'s initial values \( x_i, y_i \) and final values \( x_f, y_f \). If \( s_i = 0 \) and \( s_f = 1 \) the following cubic polynomials may be used:

\[ x(s) = s^3x_f - (s - 1)^3x_i + \alpha_xx^2(s - 1) + \beta_xx(s - 1)^2, \]
\[ y(s) = s^3y_f - (s - 1)^3y_i + \alpha_yy^2(s - 1) + \beta_yy(s - 1)^2. \]  

These polynomials does in turn satisfy the boundary conditions, i.e., \( x(s_i) = x_i, x(s_f) = x_f \) etc. There is also a condition that the robot should arrive at \( q_f \) with the same kind of motion as when leaving \( q_i \). Equation (2.9) gives...
this boundary conditions on rotation at each point:

\[
\begin{align*}
    x'(0) &= \mu_i \cos \theta_i, & x'(1) &= \mu_f \cos \theta_f, \\
    y'(0) &= \mu_i \sin \theta_i, & y'(1) &= \mu_f \sin \theta_f,
\end{align*}
\] (4.9)

where $\mu_i = \mu_f \neq 0$ and $\text{sgn}(\mu_i) = \text{sgn}(\mu_f)$. The choice of $\mu_i$ and $\mu_f$ has a definite influence on the obtained path. Now equations (2.9), (2.10) and (2.11) can be used to compute the geometric inputs.
Chapter 5

Trajectory Tracking

5.1 Introduction

In this chapter three different methods to mathematically simulate tracking of a desired Cartesian trajectory \([x_d(t) \ y_d(t)]\) will be derived, namely nonlinear dynamic system with linear and nonlinear feedback controller and input/output linearization.

One can think of the trajectory tracking problem were the robot should asymptotically track a desired trajectory, as a robot trying to track another, virtual, robot, as shown in Figure 5.1.

As previously stated, the configuration of the type (2,0) robot is completely described by (2.6). The initial position of the robot is set to \(q_0 = [x_0 \ y_0 \ \theta_0]^T\).

For the tracking problem to be solvable, it is necessary that the desired Cartesian trajectory \([x_d(t) \ y_d(t)]\) is admissible for the kinematic model described by equation (2.7), i.e., it has to satisfy:

\[
\begin{align*}
\dot{x}_d &= v_d \cos \theta_d, \\
\dot{y}_d &= v_d \sin \theta_d, \\
\dot{\theta}_d &= \omega_d, \\
\end{align*}
\]

(5.1)
where \( v_d(t) \) and \( \omega_d(t) \) are bounded with bounded derivatives and do not tend to zero when \( t \) goes to infinity.

### 5.2 Linear/Nonlinear Dynamic System

The objective when deriving this control law is for the errors \( x_d - x \), \( y_d - y \) and \( \theta_d - \theta \) to tend to zero. They can be expressed in the robot frame using:

\[
e = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_d - x \\ y_d - y \\ \theta_d - \theta \end{bmatrix}.
\] (5.2)

By differentiating and introducing, according to [3], the following change of inputs

\[
\begin{align*}
\dot{u}_1 &= -v + v_d \cos e_3, \\
\dot{u}_2 &= \omega_d - \omega,
\end{align*}
\]

a nonlinear dynamic system for the tracking error is acquired:

\[
\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \begin{bmatrix} 0 & \omega_d & 0 \\ -\omega_d & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} e + \begin{bmatrix} 0 & 0 \\ 0 & e_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}.
\] (5.3)

The first term of this system is linear, whilst the second and third are nonlinear. Furthermore, the first and second terms are in general time-varying, due to \( v_d(t) \) and \( \omega_d(t) \). It is possible to design both a linear and a nonlinear feedback law such that the error converges to zero from Equation (5.3).

The linear feedback control is known as approximate linearization. Equation (5.3) is linearized around the reference trajectory where \( e = 0 \), giving the linear state equation:

\[
\dot{e} = \begin{bmatrix} 0 & \omega_d & 0 \\ -\omega_d & 0 & v_d \\ 0 & 0 & 0 \end{bmatrix} e + \begin{bmatrix} 1 & 0 \\ 0 & e_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}.
\] (5.4)

By choosing the linear feedback controller [5] as

\[
\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = -\begin{bmatrix} k_1 & 0 & 0 \\ 0 & k_2 & k_3 \end{bmatrix} e,
\] (5.5)

and inserting it into equation (5.4), a closed-loop linearized dynamics is obtained:

\[
\dot{e} = A e = \begin{bmatrix} -k_1 & \omega_d & 0 \\ -\omega_d & 0 & v_d \\ 0 & -k_2 & -k_3 \end{bmatrix} e.
\] (5.6)
The closed-loop linearized error dynamics is thus characterized by three time varying gains:

\[
\begin{align*}
    k_1(v_d(t), \omega_d(t)) &= k_3(v_d(t), \omega_d(t)) = 2ab, \\
    k_2(v_d(t), \omega_d(t)) &= \frac{b^2 - \omega_d^2}{|v_d|},
\end{align*}
\]

(5.7)

where \(a\) and \(b\) are constants with the restrictions \(a \in (0, 1)\) and \(b > 0\). Note that \(k_2\) in equation (5.7) diverges when \(v_d\) goes to zero, i.e., when the reference Cartesian trajectory tends to stop. Therefore, the above control scheme can only be used for Cartesian trajectories were \(|v_d(t)| \geq 0 \ \forall t \geq 0\). By choosing the gains in this manner, equation (5.6) becomes asymptotically stable if the eigenvalues are not time dependent, if they are time dependent the situation is more complex [3].

As stated earlier, a nonlinear feedback controller is also available [5]:

\[
\begin{bmatrix}
    u_1 \\
    u_2
\end{bmatrix} = - \begin{bmatrix}
    k_1 & 0 & 0 \\
    k_2 v_d & k_3 \\
\end{bmatrix} \begin{bmatrix}
    \sin \theta \\
    \cos \theta \\
\end{bmatrix} e,
\]

(5.8)

where

\[
\begin{align*}
    k_1(v_d(t), \omega_d(t)) &= k_3(v_d(t), \omega_d(t)) = 2a \sqrt{v_d^2 + b \omega_d^2}, \\
    k_2 &= b.
\end{align*}
\]

(5.9)

If \(v_d(t)\) and \(\omega_d(t)\) are also bounded with bounded derivatives, and they do not both converge to zero, then the tracking error \(e\) converges to zero for any initial condition.

Hence, using the the nonlinear feedback controller described by equation (5.8), the exact expression of the tracking error dynamics, equation (5.3), can be calculated.

5.3 Input/Output Linearization

A trajectory tracking controller can be designed using input/output linearization via feedback [3]. In this control law, a ”look ahead” point which tracks the desired trajectory is introduced. The following outputs can be constructed

\[
X = \begin{bmatrix}
    y_1 \\
    y_2
\end{bmatrix} = \begin{bmatrix}
    x + \chi \cos \theta \\
    y + \chi \sin \theta
\end{bmatrix},
\]

(5.10)

in which \(X\) represents a point a distance \(\chi\) ahead of the robot. \(\chi \neq 0\), and the point is orientated along the sagittal axis, as shown in Figure 5.2.
5.3. INPUT/OUTPUT LINEARIZATION

Differentiating equation (5.10) with respect to time yields

\[
\begin{bmatrix}
\dot{y}_1 \\
\dot{y}_2
\end{bmatrix} = \begin{bmatrix}
\cos \theta & -\chi \sin \theta \\
\sin \theta & \chi \cos \theta
\end{bmatrix} \begin{bmatrix}
v \\
\omega
\end{bmatrix} = T(\theta) \begin{bmatrix}
v \\
\omega
\end{bmatrix},
\]

(5.11)

where \( T(\theta) \) is invertible as long as \( \chi \neq 0 \), since \( \det(T) = \chi \). By employing the following input transformation

\[
\begin{bmatrix}
v \\
\omega
\end{bmatrix} = T^{-1}(\theta) \begin{bmatrix}
u_1 \\
v_2
\end{bmatrix} = \begin{bmatrix}
\cos \theta & \sin \theta \\
-\sin \theta/\chi & \cos \theta/\chi
\end{bmatrix} \begin{bmatrix}
u_1 \\
v_2
\end{bmatrix},
\]

(5.12)

the equation of the robot can then be put on the form

\[
\begin{align*}
\dot{y}_1 &= u_1, \\
\dot{y}_2 &= u_2, \\
\dot{\theta} &= \frac{u_2 \cos \theta - u_1 \sin \theta}{\chi}.
\end{align*}
\]

(5.13)

Which means that an input/output linearization via feedback has been obtained. A linear controller of the form

\[
\begin{align*}
u_1 &= \dot{y}_{1d} + \kappa_1(y_{1d} - y_1), \\
u_2 &= \dot{y}_{2d} + \kappa_2(y_{2d} - y_2),
\end{align*}
\]

(5.14)

were \( \kappa_1 > 0, \kappa_2 > 0 \) can now be used to calculate the input \( v \) and \( \omega \) in equation (5.12). Equation (5.14) guarantees exponential convergence to zero of the Cartesian tracking error.
Chapter 6

Simulations

6.1 Introduction

The control laws derived in Chapter 5 will now be simulated, with the aim to create groundwork for upcoming analysis. The control laws will be simulated in continuous time, $t \in [0, t_f]$, using an ode solver in MATLAB\textsuperscript{1}. The controllers described by equation (5.5) and (5.8) will be denoted as the linear and nonlinear controller respectively. The controller described by equation (5.14) will be denoted as the input/output controller and the constants will, for simplicity, from now on be chosen as $\kappa_1 = \kappa_2 = \kappa$.

As stated earlier the type (2,0) robot has certain constraints, explained in Chapter 2 and 4. To be able to simulate the trajectory tracking problem in a good way, a valid desired trajectory need to be obtained. Therefore the simulations will start with "simpler" trajectories and move on to more advanced ones further on.

![Figure 6.1: Since the end-effector is able to move faster than the platform, it is often possible to simplify the trajectory of the platform, as seen in the figure.](image)

In this Chapter, the manipulator will be modeled in the simulations as a circle,

\textsuperscript{1}In this report, the solver \textit{ode45}, with time span $t_f$ is used.
that should be interpreted as the reachable workspace $S$. Choosing the lengths of the links according to Proposition 5 gives $S = \{ p \in \mathbb{R}^2 : \| p \| \leq \| f(\hat{\theta}) \| \}$, saying that the end-effector can take any position inside of this circle as explained in Figure 6.1. To be mentioned is that the trajectory do not depend on $\| f(\hat{\theta}) \|$.

To analyse the control laws, the norm of the Cartesian error will be plotted, accompanying the plot of every specific trajectory tracking problem. Furthermore, the input signals $\omega$ and $v$ will be plotted for each problem. Finally, Figure 6.2 explains the different objects in each trajectory plot.

6.2 Circular Trajectory

The first trajectory tracking problem to be simulated is to track the unit circle. This case is the most simple because then both $v_d(t)$ and $\omega_d(t)$ are linear time invariant, i.e., the eigenvalues are constant.

The simulated time interval goes to $t_f = 10$ seconds, in each case the initial position of the robot is $x_i = 0$, $y_i = 5$, $\theta_i = \pi/2$ and $\| f(\hat{\theta}) \| = 0.4$. The simulation for each controller follows.

![Figure 6.2](image-url)
Figure 6.3: (a): Simulation of trajectory tracking using the linear controller, when $a = 1$ and $b = 1$.
(b): Norm of the Cartesian error during the time $t$.
(c): The input signal $v$ during the time $t$.
(d): The input signal $\omega$ during the time $t$. 
Figure 6.4: (a): Simulation of trajectory tracking using the nonlinear controller, when $a = 1$ and $b = 1$.  
(b): Norm of the Cartesian error during the time $t$.  
(c): The input signal $v$ during the time $t$.  
(d): The input signal $\omega$ during the time $t$.  

6.2. CIRCULAR TRAJECTORY
Figure 6.5: (a): Simulation of trajectory tracking using the input/output controller, when $\chi = 0.01$ and $\kappa = 35$.
(b): Norm of the Cartesian error during the time $t$.
(c): The input signal $v$ during the time $t$.
(d): The input signal $\omega$ during the time $t$. 
6.3 Polynomial Trajectory

The next problem simulated is when the robot should track a polynomial trajectory. When the trajectory is a polynomial the method called planning via Cartesian polynomials, explained in Section 4.3, will be used. This gives a valid Cartesian trajectory \([x_d(t) \ y_d(t)]\). If the constants\(^2 \mu_i = \mu_f = \mu\) and the initial and final configuration is chosen for the trajectory one can calculate the corresponding \(\alpha_x, \alpha_y, \beta_x\) and \(\beta_y\) as

\[
\begin{align*}
\alpha_x &= \mu \cos \theta_f - 3x_f , \\
\alpha_y &= \mu \sin \theta_f - 3y_f , \\
\beta_x &= \mu \cos \theta_i - 3x_i , \\
\beta_y &= \mu \sin \theta_i - 3y_i .
\end{align*}
\] (6.1)

The desired trajectory, its first and second time derivative, can be calculated using equation (4.8). With this information one can use equation (2.10), (2.11) and (2.9) to get

\[
\begin{align*}
\theta_d(t) &= \arctan[2(\dot{y}_d(t), \dot{x}_d(t))] + k\pi , \\
v_d(t) &= \pm \sqrt{(x_d(t))^2 + (y_d(t))^2}, \\
\omega_d(t) &= \frac{\ddot{y}_d(t)\dot{x}_d(t) - \ddot{x}_d(t)\dot{y}_d(t)}{(\dot{x}_d(t))^2 + (\dot{y}_d(t))^2} .
\end{align*}
\] (6.2)

Since the eigenvalues are now time dependent, the stability issue becomes more complex. The properties of time dependent eigenvalues will not be discussed in this report though.

The simulated time interval goes to \(t_f = 10\) seconds, and in each case the initial position of the robot is \(x_i = -1, y_i = 5, \theta_i = -\pi/2\) and \(\|f(\dot{\theta})\| = 0.4\). Choosing \(\mu = 8\) gives \(\alpha_x = 9, \alpha_y = 2, \beta_x = 0\) and \(\beta_y = 17\). The simulation for each controller follows.

\(^2\)Editing these values changes the maximum and minimum value of the polynomial.
Figure 6.6: (a): Simulation of trajectory tracking using the linear controller, when \(a = 1\) and \(b = 1\).
(b): Norm of the Cartesian error during the time \(t\).
(c): The input signal \(v\) during the time \(t\).
(d): The input signal \(\omega\) during the time \(t\).
Figure 6.7: (a) Simulation of trajectory tracking using the nonlinear controller, when $a = 1$ and $b = 1$.
(b) Norm of the Cartesian error during the time $t$.
(c) The input signal $v$ during the time $t$.
(d) The input signal $\omega$ during the time $t$. 
Figure 6.8: (a): Simulation of trajectory tracking using the input/output controller, when $\gamma = 0.01$ and $\kappa = 35$.
(b): Norm of the Cartesian error during the time $t$.
(c): The input signal $v$ during the time $t$.
(d): The input signal $\omega$ during the time $t$. 
6.4 Rectilinear Trajectory

Finally a rectilinear trajectory will be simulated, namely the square. The simulated time interval, once again, goes to \( t_f = 10 \) seconds, in each case the initial position of the robot is \( x_i = -1, y_i = 2, \theta_i = \pi/2 \) and \( \| f(\dot{\theta}) \| = 0.4 \). The simulation for each controller follows.

\[ \]

**Figure 6.9:** (a): Simulation of trajectory tracking using the linear controller, when \( a = 1 \) and \( b = 1 \).
(b): Norm of the Cartesian error during the time \( t \).
(c): The input signal \( v \) during the time \( t \).
(d): The input signal \( \omega \) during the time \( t \).
6.4. RECTILINEAR TRAJECTORY

Figure 6.10: (a): Simulation of trajectory tracking using the nonlinear controller, when $a = 1$ and $b = 1$.
(b): Norm of the Cartesian error during the time $t$.
(c): The input signal $v$ during the time $t$.
(d): The input signal $\omega$ during the time $t$. 
6.4. RECTILINEAR TRAJECTORY

Figure 6.11: (a): Simulation of trajectory tracking using the input/output controller, when $\gamma = 0.01$ and $\kappa = 35$.
(b): Norm of the Cartesian error during the time $t$.
(c): The input signal $v$ during the time $t$.
(d): The input signal $\omega$ during the time $t$. 
Chapter 7

Analysis and Conclusion

7.1 Introduction

In this chapter the simulations shown in Chapter 6 will be analyzed, to see what benefits and drawbacks the control laws has when compared to each other.

For the linear/nonlinear controller the constants \( a \) and \( b \) edit the gains \( k_1, k_2 \) and \( k_3 \) which in turn decides if the dynamics are stable. For the input/output controller on the other hand, the distance of the "look ahead" point \( \chi \) and the constant \( \kappa \) will determine the accuracy and stability. For the simulations in Chapter 6 these constants were all chosen so that nice trajectories were obtained. By editing these, unstability may or may not occur.

The impact of editing the various constants mentioned above will be seen in Section 7.2 and 7.3. Section 7.2 will show some cases of unstability and Section 7.3 will show a few other interesting cases worth mentioning.

7.2 Unstability

That a control law is unstable can be seen from the error plot which should diverge with time. The control laws in this report are all asymptotically stable if the constants, in the linear/nonlinear case are chosen as \( a \in (0,1) \) and \( b > 0 \) and in the input/output case as \( \chi > 0 \) and \( \kappa > 0 \). Therefore, the only way to make them unstable is by choosing these constants in an incorrect manner.

In both Figure 7.1 and 7.2 the value of \( a \) is chosen to \(-1\), which gives the unstable behaviors seen. Figure 7.3 shows the input/output controller which becomes unstable when \( \kappa = -1 \). Hence it can be concluded that the restrictions given should be followed most thoroughly.
7.2. UNSTABILITY

Figure 7.1: (a): The linear controller on a circular trajectory when $a = -1$ and $b = 1$. (b): Norm of the Cartesian error during the time $t$.

Figure 7.2: (a): The nonlinear controller on a polynomial trajectory when $a = -1$ and $b = 1$. (b): Norm of the Cartesian error during the time $t$.

Figure 7.3: (a): The input/output controller on a rectilinear trajectory when $\chi = 0.1$ and $\kappa = -1$. (b): Norm of the Cartesian error during the time $t$. 
7.3 Deviations

Here follows some interesting scenarios to deepen the analysis. Subfigure 7.4a show that the linear controller becomes much more accurate for higher values of $b$. The downside can be seen in Subfigure 7.4b, namely that the input signal in $\omega$ becomes very large in the beginning. This holds for the nonlinear version as well.

Figure 7.5 shows the interesting case when $a$ tends to zero. It can be seen that the nonlinear controller starts of with unstable characteristics but still converges to the desired trajectory in the end.

For the input/output controller, the "look ahead" point $\chi$'s distance to the actual robots position plays a crucial role for the accuracy of the controller. Figure 7.6 show when $\chi$ has been chosen to large. Then the trajectory tracking becomes inaccurate. A balance need to be found for the controller to be as effective as possible, since with a small $\chi$, large input signal follows.

\begin{figure}[h]
\centering
\subfigure[Trajectory]{
\includegraphics[width=0.4\textwidth]{trajectory.png}
\caption{(a) Trajectory}
\label{fig:trajectory}
}
\hspace{0.5cm}
\subfigure[Input signal $\omega$]{
\includegraphics[width=0.4\textwidth]{input_signal.png}
\caption{(b) Input signal $\omega$}
\label{fig:input_signal}
}\caption{7.4: (a): The linear controller on a circular trajectory when $a = 1$ and $b = 100$. (b): The Input signal $\omega$ during the time $t$.}
\end{figure}
Figure 7.5: (a): The nonlinear controller on a polynomial trajectory when $a = 0.1$ and $b = 1$.
(b): Norm of the Cartesian error during the time $t$.
(c): The input signal $v$ during the time $t$.
(d): The input signal $\omega$ during the time $t$. 
7.4 Conclusions

It is now time to make some conclusions and to state some guidelines regarding the control laws that have been considered in this report. To choose a suitable control law requires a great deal of reflection, and the final decision does depend largely on the task the robot will be expected to accomplish. The aim of this section is to, by discussion of the results in Chapter 6, help the reader through this decision.

**Nonlinear Controller**

The nonlinear controller has several benefits when compared to the other two control laws. Its error converges rapidly as seen in the error plots accompanying Figure 6.4, 6.7 and 6.10. It should also be noted that for certain kinds of nonlinear control laws, there is a possibility for the error to converge to zero in finite time [6].

A drawback when compared to the linear controller is, as mentioned before, that the nonlinear controller does not stay very close to the desired trajectory, see Figure 6.4. This might make the nonlinear controller unsuitable in closely confined spaces. In less confined spaces on the other hand, the nonlinear controller works very well when tracking polynomial trajectories.

In general it has the advantage of low input signals, nice convergence of its Cartesian error and a trajectory that matches the desired one quite well. Also, the desired velocity \( v_d(t) \) can converge to zero as long as \( \omega_d(t) \) does not, and vice versa. Meaning, this controller can be used to track a Cartesian trajectory that degenerates to a simple rotation on the spot.

![Figure 7.6: (a): The input/output controller when \( \chi = 0.2 \) and \( \kappa = 35 \). (b): Norm of the Cartesian error during the time \( t \).](image-url)
Linear Controller

Compared with the nonlinear controller, Figure 6.3 and 6.4 shows that the linear controller stays closer to the desired trajectory, which is an important advantage when the robot is set to work in a confined space.

This control law works especially well when its eigenvalues are time independent. This is the case when the desired trajectory is a circle, as seen in Figure 7.4b. In that case, the error converges rapidly and the input signal on $v$ is low. Note that because $|v_d(t)| \geq 0 \quad \forall t \geq 0$ in equation (5.7), motion inversions (from forward to backward motion) on the desired trajectory are not allowed.

Input/Output Controller

The input/output controller has the fastest convergence rate of the three control laws that has been considered. This does at the first glance seem as a great advantage, but it comes with cons, namely a very large input signals as seen in the gain plots accompanying Figure 6.5, 6.8 and 6.11. Excessive input signals can cause problems if for example the engine of the robot cannot produce enough force, or if the trajectory tracking-process becomes to power-consuming. This situation is consistent with the fact that $T$ in equation (5.12) tends to become singular when $\chi$ approaches zero.

Since the input/output controller is based on a "look ahead" point there will always be a stationary error, whose size depends on the distance between the robot and this "look ahead" point.

The input/output controller does have the advantages that it heads directly to the desired trajectory without any dissipations, and that it allows the robot to stand still and rotate, a quality that is very useful when tracking rectilinear trajectories for example.

7.5 Final Words

It has been seen that each of the control laws have both benefits and certain drawbacks, and that none of them is indisputably better than the others. Depending on the situation at hand, e.g., the robot needs to work in a confined space, the robot is very durable so it can handle large input signals, the robot need to follow its trajectory with highest accuracy etc, one needs to make a careful choice. Hopefully, this report is a helping hand when faced with such a decision. Finally there are numerous control laws, and if the ones presented in this report do not fulfill certain requirements, other most definitely do.
Chapter 8

Bibliography


