Benjamin Troxler

Trajectory Generation for an Ackermann Vehicle Carrying a Dynamic Load

Master Thesis
Institute for Dynamic Systems and Control
Swiss Federal Institute of Technology (ETH) Zurich

Supervision
Prof. Dr. John Baillieul
Prof. Dr. Raffaello D’Andrea
Dr. Angela Schöllig

October 2012
IDSC-RD-AS-08
Preface

First and foremost, I would like to thank Prof. John Baillieul for giving me the opportunity to come to Boston University as well as providing advice and support throughout this project. Furthermore I thank Prof. Victor Yakhot, Konstantinos Oikonomopoulos, Ryan Hunter and Trevor Ashley for the numerous fruitful discussions. I am especially grateful for the invaluable and kind support by Dr. Kenn Sebesta both in scientific and private matters. I am also pleased to have been part of a great team at the Intelligent Mechatronics Lab. Thanks for creating the atmosphere that made my time in Boston not only instructive but also very joyful. Moreover I would like to thank Prof. Raffaello D’Andrea and Dr. Angela Schöllig for making this journey possible. What an experience.
Contents

Zusammenfassung v
Abstract vii
Nomenclature ix

1 Introduction 1

2 System Modeling 3
  2.1 Wheeled Mobile Robot 3
  2.2 Liquid Dynamics Inside the Container 6
      2.2.1 The Pendulum Model 6

3 Slosh Measuring Systems 13
  3.1 Camera Based Measuring System 13
      3.1.1 General Concept 13
      3.1.2 Implementation 15
      3.1.3 Results 18
  3.2 Voltage Divider Based Measuring System 20
      3.2.1 General Concept 20
      3.2.2 Implementation 22
      3.2.3 Results 25

4 Trajectory Generation 27
  4.1 Path 28
  4.2 Speed Profile 29
  4.3 Results 33

5 Trajectory Tracking 37
  5.1 Observer 37
  5.2 Control Loop 39
  5.3 Iterative Learning Control 40
  5.4 Results 45

6 Conclusion and Outlook 49

A Technical Drawings and Pictures 51
B List of Submitted Code 56
Zusammenfassung

Abstract

Trajectory generation for a wheeled mobile robot in Ackermann configuration carrying a partially filled cylindrical liquid container is investigated. Time-optimal trajectories from a given initial position to a desired final position that satisfy the vehicle's constraints and keep the liquid sloshing bounded are computed numerically and tested on a real system. To render the problem solvable a simple model for the liquid dynamics is introduced and empirically validated. Two custom slosh measuring systems are proposed, implemented and tested.
# Nomenclature

## Symbols

\[\begin{align*}
C_1, C_2 & \quad \text{Central points of cameras} \\
E & \quad \text{Highest point of liquid surface} \\
F_1, F_2 & \quad \text{Pixels corresponding to point E} \\
H_{\text{liq}} & \quad \text{Height of liquid in container} \\
H_o, K_o, R_o, P_o & \quad \text{Matrices of observer} \\
L_{\text{wb}} & \quad \text{Length of wheelbase} \\
L, L_v, Q, Q_v & \quad \text{Matrices of ILC} \\
P & \quad \text{Point corresponding to tip of pendulum} \\
R & \quad \text{Radius of curvature of path} \\
R_c & \quad \text{Inner radius of liquid container} \\
R_b & \quad \text{Known ballast resistor} \\
R_{\text{rod}} & \quad \text{Resistance between rod to disk} \\
S & \quad \text{Total length of path} \\
V_{\text{bat}} & \quad \text{Battery voltage} \\
V_{\text{rod}} & \quad \text{Voltage between rod and disk} \\
T_s & \quad \text{Sampling time of motion capturing system} \\
c_p & \quad \text{Damping coefficient of pendulum} \\
l_p & \quad \text{Length of pendulum} \\
m_p & \quad \text{Mass of pendulum} \\
s(t) & \quad \text{Traveled path of vehicle} \\
v(t) & \quad \text{Velocity of the vehicle} \\
u_v(t), u_\gamma(t) & \quad \text{Inputs of first order systems} \\
\tau_v, \tau_\gamma & \quad \text{Time constants of first order systems} \\
a_v, a_\gamma & \quad \text{Scalar factors of first order systems} \\
\theta(t) & \quad \text{Angle of pendulum} \\
\phi(t) & \quad \text{Angle of pendulum} \\
\psi(t) & \quad \text{Yaw angle of vehicle} \\
\gamma(t) & \quad \text{Steering angle of vehicle}
\end{align*}\]
<table>
<thead>
<tr>
<th>Indicies</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>bat</td>
<td>Battery</td>
</tr>
<tr>
<td>c</td>
<td>Container</td>
</tr>
<tr>
<td>m</td>
<td>Motor</td>
</tr>
<tr>
<td>o</td>
<td>Observer</td>
</tr>
<tr>
<td>p</td>
<td>Pendulum</td>
</tr>
<tr>
<td>r</td>
<td>Robot</td>
</tr>
<tr>
<td>rod</td>
<td>Rod</td>
</tr>
<tr>
<td>NP</td>
<td>Non-potential</td>
</tr>
<tr>
<td>TG</td>
<td>Trajectory generation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Acronyms and Abbreviations</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADC</td>
</tr>
<tr>
<td>ILC</td>
</tr>
<tr>
<td>PDE</td>
</tr>
<tr>
<td>SPI</td>
</tr>
<tr>
<td>WLOG</td>
</tr>
<tr>
<td>WMR</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

One of the basic problems in mobile robotics is trajectory generation. Given an initial state of the robot, usually comprised of its position and orientation, the task is to find a continuous curve in the state space leading to a desired final state. As opposed to path planning, this curve has an associated time-law. Typically, a trajectory is sought that minimizes a predefined cost function that contains penalties for time and energy. Depending on the robot, various kinematic and dynamic constraints arise, such as minimally achievable radius of curvature, maximum velocity or maximum acceleration.

This work extends the basic trajectory generation problem described above by introducing additional dynamic constraints. Early research in this field deals with the problem of moving an elastic load to a final destination such that no energy is stored in the load as the vehicle reaches its destination [1]. Here a slightly different problem is considered: The robot carries a cylindrical liquid container, partially filled with water, that must not be spilled during the maneuver.

The used vehicle is an off-the-shelf chassis with Ackermann steering geometry that is driven by a DC-motor and equipped with a servomechanism for the steering. The liquid container is fixed to the chassis near the rear axle. Generation of time-optimal trajectories, that keep the liquid from spilling and satisfy the vehicle’s constraints, is formulated as a numerical optimal control problem and solved offline. The found open-loop trajectories are then verified on the actual vehicle, iterative learning control is used to improve tracking performance. A motion capturing system measures the position and heading of the vehicle while the liquid sloshing is measured by two custom slosh measuring systems.

In the second chapter of this report the equations of motion of the vehicle are discussed and its actuators are characterized. A simple model for the liquid dynamics inside the container is introduced and its parameters are estimated through experiments. In the third chapter the two custom slosh measuring systems are proposed and their results are compared. The trajectory generation algorithm is discussed in chapter four. Finally, the resulting trajectories are tested on the vehicle. The experimental setup, the used control laws and the results are described in chapter five.
Chapter 2

System Modeling

2.1 Wheeled Mobile Robot

The wheeled mobile robot (WMR) used in this work is a four-wheel vehicle in Ackermann configuration: It has two steered wheels in the front and two motorized wheels in the back. The vehicle’s position is defined by the origin of a coordinate frame that is fixed to the body of the car. This $v$-coordinate frame has its origin $O_v$ in the middle of the vehicles rear axle and its position with respect to the world coordinate frame $w$ is denoted by $wO_v = (x(t), y(t))^T$.

The basis vector $v_{ex}$ points in the direction of the vehicle’s current heading, the yaw angle is denoted by $\psi(t)$, the steering angle by $\gamma(t)$, see Figure 2.1. The kinematic equations read as

$$
\begin{align*}
\dot{x}(t) &= v(t) \cos(\psi(t)) \\
\dot{y}(t) &= v(t) \sin(\psi(t)) \\
\dot{\psi}(t) &= \frac{v(t)}{L_{wb}} \tan(\gamma(t)) ,
\end{align*}
$$

where $v(t)$ is the vehicle’s velocity and $L_{wb}$ is the vehicle’s wheel base.

Figure 2.1: Definition of the world coordinate frame $\{w_x, w_y, w_z\}$, the $v$-coordinate frame $\{v_x, v_y, v_z\}$, that is fixed to the vehicle, and the steering and yaw angles $\gamma(t)$ and $\psi(t)$, respectively.
DC-Motor of Powered Wheels

The back wheels are powered by a sensored brushless DC-motor\(^1\), that features hall sensors for enhanced performance at low motor speeds. Its peak current of 100 A enables fast and aggressive maneuvers.

The well known voltage controlled DC-motor model comprises the two differential equations

\[
\begin{align*}
\Theta_m \dot{\omega}_m(t) &= \kappa_m i_m(t) - d_m \omega_m(t) \\
L_m \dot{i}_m(t) &= u_m(t) - \kappa_m \omega_m(t) - R_m i_m(t),
\end{align*}
\]

where

- \(\omega_m(t)\) is the motor’s rotational speed [rad/s]
- \(i_m(t)\) is the motor’s current [A]
- \(u_m(t)\) is the input voltage [V]
- \(\Theta_m\) is the motor’s rotational inertia [kgm\(^2\)]
- \(R_m\) is the resistance of the equivalent RL-circuit [Ohm]
- \(L_m\) is the electric inductance of the equivalent RL-circuit [H]
- \(\kappa_m\) is the motor constant [Nm/A]
- \(d_m\) is the damping constant [Nms].

![Figure 2.2: The WMR’s velocity after a step input at time \(t = 0.5\) s. Step from 0% to 27% of the maximal input and back to 0% at \(t = 1.3\) s.](image)

It is a reasonable simplification to neglect the dynamics of the current \(i(t)\) because they are a lot faster than the dynamics of the mechanical part. Setting \(\dot{i}(t) = 0\) in Eq. (2.3) and solving for the current \(i(t)\) yields

\[
\dot{i}_m(t) = \frac{1}{R_m} (u_m(t) - \kappa_m \omega_m(t))
\]

Inserting this into Eq. (2.2) and solving for \(\dot{\omega}_m(t)\) yields

\[
\dot{\omega}_m(t) = \frac{\kappa_m}{R_m \Theta_m} (u_m(t) - (\kappa_m + d_m) \omega_m(t)) \quad (2.5)
\]

\[
= \frac{1}{\tau_\omega} (u_m(t) - \omega_\omega \omega_m(t)),
\]

\(1\) The used ESC/Motor combo is a Fuse Sensored 1/10-Scale Brushless 8.5T System.
which is a first order system. Assuming that there is no wheel slip or gear backlash, the motor speed \( \omega_m(t) \) is proportional to the vehicle’s speed \( v(t) \). Hence one can write

\[
\dot{v}(t) = \frac{1}{\tau_v} (a_v(t) - a_v v(t)) . \tag{2.7}
\]

In order to find the constants \( \tau_v \) and \( a_v \), multiple step responses of the WMR’s velocity are recorded, see Figure 2.2. Numerical curve fits yield the values \( \tau_v = 0.35 \) s and \( a_v = 11.79 \).

Servomechanism of Steered Wheels

The front wheels are steered by a digital servomechanism, that consists of a servo motor and a position encoder. The motor’s position is controlled by an internal feedback loop governed by the input voltage. Because the vehicle is equipped with an off-the-shelf unit, this control law (typically a PID controller) is not known. The behavior is therefore characterized by looking at step responses of the system, see Figure 2.3.

![Figure 2.3: Response of the servomechanism’s rotor after a step input at time \( t = 0.5 \) s.](image)

Note that the data corresponds to the motor shaft’s movement and not the movement of the actual wheels. The steering angle of the wheels is assumed to be a linear function of the rotor position, backlash in the steering mechanism is neglected. The overshoot reveals higher order dynamics in the system. Nevertheless, taking into account the inaccuracies described above and for the sake of simplicity, only a first order model is fitted to the data

\[
\dot{\gamma}(t) = \frac{1}{\tau_\gamma} (a_\gamma(t) - a_\gamma \gamma(t)) . \tag{2.8}
\]

Numerical curve fits result in the parameters \( \tau_\gamma = 0.0795 \) s and \( a_\gamma = 0.0035 \).

The WMR’s Full Set of Dynamic Equations

Defining the state of the WMR

\[
q_r = (x(t), y(t), \psi(t), v(t), \gamma(t))^T \tag{2.9}
\]
and recalling Eq. (2.1), (2.7) and (2.8) the WMR is described by

\[
\dot{q}_v(t) = \int_{v_0}^{v(t)} = \begin{pmatrix}
\dot{z}(t) \\
\dot{y}(t) \\
\dot{\psi}(t) \\
\dot{\gamma}(t)
\end{pmatrix} = \begin{pmatrix}
v(t) \cos(\psi(t)) \\
v(t) \sin(\psi(t)) \\
\frac{v(t)}{r_{wa}} \tan(\gamma(t)) \\
\frac{1}{r_s} (u_v(t) - a_v v(t)) - \frac{1}{r_s} (u_{\gamma}(t) - a_{\gamma} \gamma(t))
\end{pmatrix},
\]

(2.10)

with the two inputs \( u_v(t) \) and \( u_{\gamma}(t) \).

### 2.2 Liquid Dynamics Inside the Container

Modeling the dynamics of water inside a cylindrical container that is subject to angular and translational accelerations is in general a very complicated task. The liquid’s behavior is described best by a set of partial differential equations (PDE). However, such models are not suitable for control purposes because solving the PDEs is both challenging and time consuming. This work uses a heavily simplified model instead: The water’s dynamics are modeled as a damped spherical pendulum. This model has shown to approximate the behavior of liquids in cylindrical containers sufficiently well if the liquid is not excited too strongly [24], [13], [14].

![Figure 2.4: The liquid dynamics are modeled as a damped spherical pendulum that is suspended at the origin \( O_v \). The pendulum is orthogonal to the liquid surface at all times.](image)

#### 2.2.1 The Pendulum Model

The container is placed on the vehicle such that its cylinder axis contains the origin \( O_v \) defined in section 2.1 and such that the \( z \)-coordinate of the origin \( O_v \) corresponds to the steady state liquid height. The basic idea is that the pendulum is orthogonal to the liquid’s surface and suspended at the origin \( O_v \), see Figure 2.4. This simplified model entails the simplifications that the liquid’s surface is assumed to be planar and to contain the origin \( O_v \) at all times.
Parameter Identification

The length of the pendulum can be determined by calculating the natural frequency of water in a cylindrical container with free fluid surface. This frequency is given by [3]

\[ f_n = \frac{\omega_n}{2\pi} = \frac{1}{2\pi} \sqrt{\frac{g}{R_c}} \epsilon_0 \tanh \left( \epsilon_0 \frac{H_{\text{liq}}}{R_c} \right), \tag{2.11} \]

where

- \( g = 9.81 \text{ m/s}^2 \) is the gravitational acceleration
- \( R_c \) is the radius of the liquid container [m]
- \( \epsilon_0 = 1.8412 \) is the first zero of the first derivative of the Bessel function of first kind and first order [-]
- \( H_{\text{liq}} \) is the liquid’s height when it is at rest [m].

The length of the pendulum \( l_p \) can then be determined using the well known relationship for the natural frequency of a 1-link pendulum

\[ f_n = \frac{1}{2\pi} \sqrt{\frac{g}{l_p}}. \tag{2.12} \]

For the liquid container used in this work, with \( R_c = 20.08 \text{ cm} \) and \( H_{\text{liq}} = 1.875 \text{ cm} \approx 4.76 \text{ cm} \), this yields \( f_n = 2.9 \text{ Hz} \) and equivalently \( l_p = 0.029 \text{ m} \).

Experiments were carried out to get a better estimate of the equivalent pendulum length and the damping constant: The liquid container was excited and then held in place while the sloshing was measured. The length and the damping constant are estimated by simulating the model of a damped 1-link pendulum

\[ \ddot{\theta}(t) = -\frac{g}{l_p} \sin(\theta(t)) - \frac{c_p}{l_p^2 m_p} \dot{\theta}(t), \tag{2.13} \]
2.2. Liquid Dynamics Inside the Container

where \( m_p = 0.385 \text{kg} \) is the mass of the liquid inside the container. The parameters \( l_p \) and \( c_p \) are varied such that the difference between the measured and simulated data is minimized. Several experiments were carried out, the means of the found parameters are taken as the final parameters

\[
l_p = 0.028 \text{m} , \quad c_p = 1.83 \cdot 10^{-4} \text{m}^2\text{kg/s} .
\]

Note that the experimentally determined pendulum length is very close to the theoretical pendulum length given by Eq. (2.11).

The pendulum model also entails the assumption that the sloshing frequency is independent of the amplitude. In reality this is not true, Eq. (2.11) is derived under the assumption

\[
\frac{\partial H}{H_{\text{liq}}} \ll 1 , \tag{2.14}
\]

where \( \partial H \) is the maximal height of the surface waves inside the container. For the given setup, the assumption of a constant sloshing frequency showed to be good for angles below 15 deg = \( \frac{\pi}{12} \) rad.

Equations of Motion of the Liquid

The liquid container is fixed to the WMR, its rotational and lateral accelerations are governed by the vehicle. This directly translates to the pendulum model: The pendulum is suspended at the point \( O_v \) which is fixed to the vehicle. In what follows, the pendulum’s equations of motion are derived using the Euler-Lagrange Equation.

\[
\begin{align*}
\dot{v}_e^x &= R(t) \omega(t) \\
\dot{v}_e^y &= s(t) \\
\dot{v}_e^z &= 0 \\
\end{align*}
\]

Figure 2.6: Definition of the variables used in the derivation of the pendulum’s equations of motion.

The pendulum’s velocity is expressed in the \( v \)-frame. The \( v \)-frame is rotating about \( v_e^z \) with an angular velocity of \( \omega(t) \). The angular velocity is defined by \( R(t) \), the current radius of curvature of the WMR’s path, and the WMR’s current velocity \( v(t) \) as

\[
\omega(t) = \frac{v(t)}{R(t)} , \tag{2.15}
\]
see Figure 2.6. The relationship between the steering angle and the radius of curvature depends on the vehicle’s chassis and is given by

$$R(t) = \frac{L_{wb}}{\tan(\psi(t))}.$$  \hfill (2.16)

Three coordinate frames are used in the derivation of the pendulum’s equations of motion: The vehicle coordinate frame $v$ introduced above, the intermediate coordinate frame $i \langle i_x, i_y, i_z \rangle$ and the pendulum coordinate frame $p \langle p_x, p_y, p_z \rangle$. The $i$-frame results from the $v$-frame by a rotation of $\theta(t)$ about $v_y$. The $p$-frame is fixed to the pendulum and results from a rotation by $\phi(t)$ about $i_x$, see Figure 2.7.

![Figure 2.7: The three coordinate frames used in the derivation of the equations of motion.](image)

The position of the pendulum $P$ in the $p$-frame is

$$p_{OP} = \begin{pmatrix} 0 \\ 0 \\ -l_p \end{pmatrix},$$  \hfill (2.17)

with $\|p_{OP}\| = l_p$ the length of the pendulum. This vector can be expressed in terms of the $v$-frame

$$v_{OP} = A_{vi} \cdot A_{ip} \cdot p_{OP}$$

$$= \begin{pmatrix} \cos(\theta(t)) & 0 & \sin(\theta(t)) \\ 0 & 1 & 0 \\ -\sin(\theta(t)) & 0 & \cos(\theta(t)) \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi(t)) & -\sin(\phi(t)) \\ 0 & \sin(\phi(t)) & \cos(\phi(t)) \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ -l_p \end{pmatrix}$$

$$= \begin{pmatrix} -l_p \cos(\phi(t)) \sin(\theta(t)) \\ l_p \sin(\phi(t)) \\ -l_p \cos(\phi(t)) \cos(\theta(t)) \end{pmatrix},$$  \hfill (2.18)
where \( A_{vi} \) and \( A_{ip} \) are the rotation matrices from the \( i \) to the \( v \)-frame and from the \( p \) to the \( i \)-frame, respectively.

In order to define the pendulum’s kinematic energy, the velocity of the pendulum needs to be determined. The velocity is expressed in the \( v \)-frame, which is moving itself with velocity \( v(t) \) in the direction of \( v_e \) and spinning about \( v_e \) with an angular velocity of \( \omega(t) \). Using the Euler derivative rule the velocity of the pendulum is determined as

\[
v_{vP} = v_{vi} + \omega_{iv} \times v_{iP} \]

The kinematic energy of the pendulum can now be calculated as

\[
T = \frac{1}{2} \cdot m_p \cdot v_{vP}^T v_{vP},
\]

and the potential energy is given by

\[
V = -m_p g l_p \cos(\theta(t)) \cos(\phi(t)).
\]

The Euler-Lagrange Equation can now be formulated

\[
\frac{d}{dt} \frac{\partial T}{\partial \dot{q}_p} - \frac{\partial T}{\partial q_p} + \frac{\partial V}{\partial \dot{q}_p} = f_{NP},
\]

where \( q_p = [\theta(t), \phi(t)]^T \) and \( f_{NP} \) is the vector of non-potential moments introduced by the damping

\[
f_{NP} = -c_p \cdot \left( \frac{\dot{\theta}(t)}{\phi(t)} \right),
\]

where \( c_p \) is the damping constant.

Evaluation of (2.21) yields the pendulum’s equations of motion, which read as

\[
\ddot{\theta}(t) = -\frac{g \sin(\theta)}{l_p \cos(\phi)} + 2 \dot{\phi} \tan(\phi) + \frac{\dot{v} \cos(\theta)}{l_p \cos(\phi)} - \frac{c_p \dot{\theta}}{l_p^2 m_p \cos(\phi^2)} - \frac{2 v \cos(\theta) \dot{\phi}}{R} - \frac{\dot{v} \cos(\theta) \tan(\phi)}{R} + \frac{v^2 \sin(\theta) \cos(\theta)}{R^2}
\]

\(^2\)For improved readability the indication of time dependent variables \( \cdot(t) \) is dropped on the right hand side of the equations.
\[
\ddot{\phi}(t) = - \frac{g \cos(\theta) \sin(\phi)}{l_p} - \dot{\theta}^2 \sin(\phi) \cos(\phi) - \frac{\dot{v} \sin(\theta) \sin(\phi)}{l_p} - \frac{c_p \dot{\phi}}{l_p^2 m_p}
\]
\[
- \frac{\dot{v}^2 \cos(\phi)}{l_p R} + \frac{2 \dot{v} \dot{\theta} \cos(\theta) \cos(\phi)}{R} + \frac{\dot{v} \sin(\theta)}{R}
\]
\[
+ \frac{v^2 \cos(\theta)^2 \sin(\phi) \cos(\phi)}{R^2}
\]  
(2.24)

In case the WMR follows a straight path, \( R(t) \) goes to infinity and the terms with \( R(t) \) in the denominator vanish.
2.2. Liquid Dynamics Inside the Container
Chapter 3

Slosh Measuring Systems

This section investigates methods to measure the sloshing of water inside a cylindrical container. Capturing the full shape of the liquid surface would require multiple liquid level measurements distributed over the whole cross section of the liquid container. However, the strong simplifications introduced by the pendulum assumption facilitate measuring the liquid’s current state: The liquid surface is assumed to be planar and to contain the origin at all times. Because a plane is fully constrained by three distinct points, measuring the liquid level at two points (different from the origin) is sufficient to define the current state of the liquid.

Because point measurements are needed, common ultrasound liquid level sensors are not suitable. Instead, laser displacement sensors have been used for this purpose [24], [13]. However, aside from being expensive, heavy and bulky, these sensors are in general not well suited for the task: As the liquid is sloshing, the laser beam does not impinge on the fluid surface orthogonally at all times, which is a major source for errors. Other commercial liquid level sensors are fiber-optic devices\(^1\). While their measurement principle is promising, they suffer from the same basic shortcomings as the laser displacement sensors: They are costly, heavy and large.

Because of the lack of inexpensive commercial sensors, two custom measuring systems have been developed which are described below.

3.1 Camera Based Measuring System

3.1.1 General Concept

It follows directly from the pendulum assumption, that the highest point \( E \) of the liquid surface is always on the boundary of the container. Furthermore, its position defines the current posture of the liquid surface completely: The liquid surface contains the peak \( E \) and the origin \( O_x \). Moreover, it contains the horizontal line \( AB \) that is orthogonal to the line between the peak and the origin, see Figure 3.1.

Measuring the current state of the liquid can therefore be achieved by measuring the current location of the peak \( E \). This can be accomplished by two cameras

\(^1\)For example the SICK WLL190T-2
placed outside of the beaker, one on the x-axis and one on the y-axis and both facing the container, see Figure 3.1.

Figure 3.1: The camera based measuring principle: One camera is placed on the x-axis and one on the y-axis. The position of the peak E defines the plane ABE and therefore the current posture of the liquid surface.

$C_1$ and $C_2$ are the central points of the two cameras according to the so-called central projection model: This model assumes, that the image sensor only measures the intensity of light passing through a single point, the central point. The distance between the central point of a camera and its image plane is the focal length. The actual image acquired by the camera’s sensor is located behind the central point and upside down. For reasons of clarity, so-called virtual images are used. They are identical to the actual images, but are located a focal length before the central point and are the right way up.

The process of finding the current position of the peak E comprises three steps:

1. Determining the coordinates $(u_{F1}, v_{F1})$, $(u_{F2}, v_{F2})$ of the pixels $F_1$, $F_2$ corresponding to the peak.

2. Given the pixel coordinates, compute the optical rays $C_1F_1$ and $C_2F_2$.

3. Intersect the optical rays with the cylinder and compute the final estimate based on the four intersection points.

The individual steps are described in more detail below.
3.1.2 Implementation

1. Finding the Peak’s Pixel Coordinates

To find the peak’s pixel coordinates \((u_{F1}, v_{F1})\) and \((u_{F2}, v_{F2})\) for both camera frames, intensity thresholding is applied to the images. Depending on the used cameras, the acquired images need to be converted to grayscale images for further processing. Subsequently the images are transformed into binary images: All pixels with an intensity value above a certain threshold are set to maximum intensity (white), all pixels with an intensity value equal to or below the threshold are set to minimum intensity (black). The threshold is chosen according to Otsu’s Method [16]. This method assumes that the image has a bimodal histogram and finds the threshold that minimizes the intra-class variance of the two pixel classes. Next the outlines of the fluid are extracted with a Canny edge detector [7]. Once the outlines are known, finding the peak amounts to finding the highest edge pixel, that is the edge pixel with the smallest \(u\) coordinate (recall the definition of the image coordinate frames in Figure 3.1). Image frames of the individual steps are depicted in Figure 3.4.

![Camera 1](image1.png) ![Camera 2](image2.png)

Figure 3.2: Example image frames of fluid close to \(\theta(t) = 0\) and \(\phi(t) = 0\). Note that the location of the peak \(E\) is not clearly defined.

The process described above has one caveat: If the fluid is close to the state \(\theta(t) = 0\) and \(\phi(t) = 0\) the peak is not a good indicator for the fluid’s state as it is not clearly defined, see Figure 3.2. Aside from being very inaccurate, the position reading is also very jittery because of ripples on the liquid surface. To overcome this problem a least squares line is fitted through all the edge pixels. If the sum of absolute errors between the fitted line and the actual edge and the slope of the fitted line are below specific thresholds in both frames, the state of the liquid is set to \(\theta(t) = 0\) and \(\phi(t) = 0\).

The intensity thresholding works best if there is a stark contrast between the fluid and the surroundings. The contrast can be enhanced easily by adding color to the water.

2. Finding the Optical Ray Corresponding to the Peak

If the camera’s focal length is known and constant for the whole image plane, finding the optical ray is trivial: It is the straight line defined by the central point of the camera and the corresponding pixel in the image plane, \(C_1F_1\) and \(C_2F_2\), see Figure 3.1. However, due to lens distortion and other irregularities inside the camera, the focal length typically varies across the image plane. In light of the central projection model this means that the distance between the
central point and the image plane is pixel dependent. Using the OCamCalib Toolbox for MATLAB by D. Scaramuzza et al. [18], [20], [21] the cameras are calibrated and the focal lengths are expressed as functions of the distances $\rho_i$ from the image centers. The optical rays are then found as $C_1(\rho_1)F_1$, $C_2(\rho_2)F_2$ with

$$
\rho_i = \sqrt{(u_{F_1} - u_{M_i})^2 + (v_{F_1} - v_{M_i})^2},
$$

(3.1)

where $(u_{M_i}, v_{M_i})$ are the pixel coordinates of the image center of camera $i$.

3. Finding the Peak’s 3-D Coordinates

The peak $E$ lies on the boundary of the fluid container and on the optical rays going through the two pixels corresponding to the peak. Once the optical rays are known, they can be intersected with the cylindrical container boundary. For each optical ray, this corresponds to a system of equations with either zero, one or two solutions. If there are no solutions, the optical ray does not intersect the fluid container. Clearly the peak’s pixel coordinates are erroneous and the measurement is discarded. The case with only one solution arises if the optical ray is tangent to the container, which is very unlikely to occur in practice. Nevertheless, the code handles this exception. In general there are two solutions and thus each of the images give rise to two candidates for the peak’s location.

![Figure 3.3: Top views of the measurement setup and the actual location of the peak $E$ (red). Both images give rise to two candidates for the peak’s location, $\{E_{11}, E_{12}\}$ and $\{E_{21}, E_{22}\}$, respectively. The correct two candidates can be chosen by looking at the $v$-coordinates of the pixels $F_1$ and $F_2$.](image)

Out of those four candidates, only two correspond to the actual peak, see Figure
3.3. The correct two candidates can be chosen by looking at the coordinates \( v_{E1} \) and \( v_{E2} \). Recall the \( v \)-coordinates of the two image centers, \( v_{M1} \) and \( v_{M2} \), respectively. Then

- If \( (v_{E1} < v_{M1}) \land (v_{E2} > v_{M2}) \) the peak is in the first quadrant,  
  \( \rightarrow \) the correct two candidates are \( E_{11} \) and \( E_{21} \).

- If \( (v_{E1} > v_{M1}) \land (v_{E2} > v_{M2}) \) the peak is in the second quadrant,  
  \( \rightarrow \) the correct two candidates are \( E_{11} \) and \( E_{22} \).

- If \( (v_{E1} > v_{M1}) \land (v_{E2} < v_{M2}) \) the peak is in the third quadrant,  
  \( \rightarrow \) the correct two candidates are \( E_{12} \) and \( E_{22} \).

- If \( (v_{E1} < v_{M1}) \land (v_{E2} < v_{M2}) \) the peak is in the fourth quadrant,  
  \( \rightarrow \) the correct two candidates are \( E_{12} \) and \( E_{21} \).

The handling of cases for which \( (v_{E1} = v_{M2}) \) or \( (v_{E2} = v_{M2}) \) is straightforward and included in the code.

Figure 3.4: Left column: Frames of camera 1. Right column: Frames of camera 2. Top row: Actual image frames. Second row: Images after thresholding. Third row: Extracted edges with peak (red dots).
Due to inaccuracies and noise the two peak locations coming from the two cameras are in general not equal. The final estimate is taken as the mean position of the two peak locations.

The pendulum is perpendicular to the plane defined by points $A$, $B$ and $E$ in Figure 3.1. Denoting the current location of the peak by $E(t)$, the pendulum’s current direction is given by

$$\mathbf{P}(t) = \begin{pmatrix} P_x(t) \\ P_y(t) \\ P_z(t) \end{pmatrix} = \overrightarrow{AB} \times \overrightarrow{OE}(t),$$

where $\times$ is the standard vector cross product. Straight forward trigonometric considerations show that the angles read as

$$\theta(t) = -\arctan\left(\frac{P_x(t)}{|P_x(t)|}\right)$$

$$\phi(t) = \arctan\left(\frac{P_y(t)}{|\cos(\theta(t)) + P_z(t)\sin(\theta(t))|}\right).$$

The main script of the algorithm is `main_CameraBasedSloshMeasuring.m`.

### 3.1.3 Results

The algorithm is implemented both in MATLAB and OpenCV. Aside from greatly increased runtimes, the MATLAB version performs equally well. The following plots and figures have been produced with the MATLAB version. Figure 3.4 shows typical image frames of the two camera views. It is evident that there are some shortcomings:

- The thresholding does not always produce smooth boundaries, the two regions sometimes have spikes. This can be attenuated by applying erosion and dilation operations at the price of erasing some of the finer details of the boundary (note the horizontal boundaries in both edge images on the right side). A lot more powerful and hence computationally expensive techniques would be required for further improvements.

- The pixel corresponding to the peak in each of the two views can be ambiguous. Note the horizontal boundary of the black region in the frame of camera 1.

- Reflections on the liquid surface and on the container walls can cause major flaws. Note the black spike on the right side of the thresholded image of camera 2 due to the reflection of a dark object in the container wall. Because the measurement setup is used on a moving platform, the environment is subject to change and introduces serious disturbances: Objects that appear behind the measurement setup or that are reflected in the container walls and changing lighting conditions are not easily dealt with.

Figure 3.5 shows the final output of the algorithm after the container was shaken. Overall the performance is satisfactory. However, the algorithm struggles if the liquid angles are too small, note the noisy section for the the estimation of $\theta(t)$ in between frames 70 to 120. The performance is better for stronger sloshing, due to the fact that the peaks are more distinct the higher they are and
can hence be localized with higher precision. This shortcoming is especially unfavorable regarding slosh damping tasks: The desired state of the liquid, \((θ(t), φ(t))^T = (0, 0)^T\), has the highest uncertainty. An advantage of the algorithm is the fact that it is generic and works for any cylindrical container. The container itself does not need to be modified. However, there are major problems with the computational burden: The algorithm is computationally very expensive and needs a powerful computer to run in real time. Hence the processing cannot be done on board of the vehicle. For the algorithm to run on a computer, the two videos need to be streamed wirelessly, which is a very demanding task in itself. Furthermore, because the cameras need to be placed at a distance from the liquid container, the setup is quite large.
3.2 Voltage Divider Based Measuring System

3.2.1 General Concept

Measuring the Liquid Level at one Point

If two pieces of metal are placed inside an electric conductive fluid and each of the metal pieces is connected to one of the poles of a battery, current will start to flow from one piece to the other. The amount of current that is flowing depends on the applied battery voltage and the resistance of the circuit. The resistance of the circuit is mainly determined by the size of the contact area between the two pieces and the fluid. This simple observation gives rise to the following liquid level measuring principle: A metal disk is placed on the bottom of the container, a metal rod is attached to the lid such that it is partially submerged into the liquid and both pieces are connected with one of the poles of a battery. As the liquid sloshes, the surface area of the rod that is below the water line changes and the resistance $R_{rod}$ between the rod and the disk changes accordingly. To measure this change in resistance a second resistor $R_b$ of known size is connected in series with the rod to construct a voltage divider. The voltage drop across the rod can be expressed as

$$V_{rod} = V_{bat} \frac{R_{rod}}{R_{rod} + R_b},$$

where $V_{bat}$ is the battery voltage, see Figure 3.6.

Figure 3.6: The electric circuit corresponding to the measurement setup to determine liquid levels.

The measured voltage $V_{rod}$ is a function of $R_{rod}$ and hence of the liquid level at the rod. Calibrating the setup by measuring the voltage corresponding to known liquid heights yields a liquid level measurement device. Of course, in principle water is not an electric conductive fluid. However, because tap water is used, there are many additional particles in the liquid that render it conductive.

Extension to Liquid Level Measurements at Multiple Points

To estimate the current angles of the liquid surface at least two level measurements and hence two rods are necessary (recall the assumption that the liquid surface always contains the origin). Of course, the more level measurements are available, the better the achievable estimate, which is why a total of five rods are used. The naive approach to adding more rods is to simply replicate
the voltage divider in Figure 3.6 the required number of times. However, this leads to problems: As the liquid sloshes and the voltage drops across the rods vary, potential differences between the rods arise. Hence current is no longer only flowing between the rods and the disk, but also between the rods. As a result the rods’ dynamics are coupled and Eq. (3.5) does no longer hold. There is still a relation between the measured voltages across the rods and the liquid levels at the rods, however, it is of more complicated nature. Because the rods are coupled, they cannot be calibrated separately which results in an extremely time consuming calibration procedure.

![Diagram](image)

**Figure 3.7:** The measurement setup to measure the liquid level at five points.

Instead, the setup depicted in Figure 3.7 is used. The five rods are connected to an analog demultiplexing device\(^2\) that is connected in series with the ballast resistor \(R_b\). The demultiplexer takes in one channel and connects it to one of five output channels according to a control input. The four not connected channels are set to a high impedance state. This ensures that current only flows through one rod at a time and hence the five rods are decoupled. An ATmega 1280 microprocessor on an Arduino Mega board is used to control the demultiplexer such that one rod after another is connected for \(T_{rod} = 1\) ms in an infinite loop. The order is chosen as \([1, 3, 5, 2, 4, \ldots]\) such that measurement point moves around the container quickly. Also supplied by the Arduino Mega board is the battery voltage \(V_{bat} = 5\) V. The voltage drops across the rods is measured with a 10-bit analog to digital converter (ADC) with a range of 0 to 5 V on the Arduino Mega board. This gives a resolution of

\[
\frac{5}{2^{10}} \approx 4.8 \text{ mV}
\]

which is sufficient in light of the overall accuracy of the system.

\(^2\)The used device is the DIP16 version of the 74HC4051 by NXP Semiconductors
3.2.2 Implementation

Manufacturing of the Measurement Device

The liquid container was machined out of transparent polycarbonate. The material was chosen because of its high machinability, low electric conductivity and lightness. The disk and the rods are made out of copper. Top parts for the rods were machined out of copper. These parts have tapped holes on the top which are used to connect the external leads with brass screws and ring connectors. The top pieces are soldered onto the rods.

An explosion drawing of the whole assembly is depicted in Figure 3.8. The technical drawings as well as a picture of the device are in appendix A.

Figure 3.8: Explosion view of the voltage divider based measuring system (without the electric circuit).
Optimal Ballast Resistor

The ballast resistor \( R_b \) has an influence on the voltage drops across the rods according to Eq. (3.5). In order to maximize the resolution of the measurement system, \( R_b \) should be chosen such that the change in the voltage reading is maximal as the liquid sloshes. Let \( R_{\text{min}} \) be the minimal resistance between the rods and the disk (corresponding to the maximal liquid level for which the system is used) and \( R_{\text{max}} \) be the maximal resistance between the rods and the disk (corresponding to the minimal liquid level for which the system is used).

The difference between the voltage drops in function of the ballast resistor is

\[
V_{\text{diff}}(R_b) = V_{\text{bat}} \left( \frac{R_{\text{max}}}{R_{\text{max}} + R_b} - \frac{R_{\text{min}}}{R_{\text{min}} + R_b} \right).
\]  

(3.6)

Setting the first derivative of \( V_{\text{diff}} \) with respect to \( R_b \) to zero and checking that the second derivative is negative gives the maximizing ballast resistor

\[
R_b^\star = \sqrt{R_{\text{min}} \cdot R_{\text{max}}}.
\]  

(3.7)

Calibration

The setup is calibrated by filling the beaker with water up to predefined levels and measuring the corresponding voltage drops across all five rods. For each rod a least squares second order polynomial is fitted through the corresponding data points to get a function

\[
\text{level}_i = f_i(V_{\text{rodi}}),
\]

(3.8)

see Figure 3.9. The calibration routine is implemented in the MATLAB function `main_performCalibration.m`.

![Figure 3.9: Typical calibration results: Measured voltages at defined liquid levels (green asterisks) and second order polynomial fit (blue).](image)

Serial Communication

The acquired data is wirelessly sent to a computer using the ZigBee protocol on two XBee Pro 900 modules. Each voltage reading has to be paired with an identifier for the rod that was connected when the voltage was measured. Choosing the identifiers to be the numbers 0 to 4, one data package comprises 13 bits: 10 bits for the voltage and 3 bits for the identifier.

In SPI serial communication the smallest amount of data that can be sent at a time is one byte (which equals 8 bits). Hence two bytes are needed to send one data package. Because data is being sent continuously there is no way of telling
when a data package ends and when a new package starts. For this reason, the first bit of each byte is used as an identifier. The identifier bit is set to 0 for byte 1 and 1 for byte 2. The complete partitioning of a data package is shown in Figure 3.10.

![Figure 3.10: Partitioning of a data package. Byte identifier (yellow), rod identifier (green) and voltage reading (blue). Bit 4 of the second byte is not used.](image)

Each rod is connected with the battery for $T_{rod} = 1\, \text{ms}$ and voltages are read out by the ADC every $T_S = 0.2\, \text{ms}$. This results in five voltage readings per rod per cycle. Because of the limited baud rate of the XBee serial link, the data cannot be sent out at the rate it is acquired. Instead, the voltages corresponding to the same rod are buffered onboard the Arduino Mega and their average is sent out as soon as the demultiplexing device switches rods.

**Post Processing**

The data is received by MATLAB and the voltages and the corresponding rod identifiers are extracted from the data packages. Using the functions relating voltages and liquid levels generated during calibration, the current liquid level at each rod is determined.

In the next step, a least squares plane containing the origin is fitted through the five data points. A plane through the origin is given by

$$ax + by + z = 0$$

with parameters $a$ and $b$. Defining the matrix $X$

$$X = \begin{pmatrix} x_{rods} \\ y_{rods} \end{pmatrix} = d \begin{pmatrix} \cos(0) & \cos\left(\frac{\pi}{5}\right) & \cos\left(\frac{2\pi}{5}\right) & \cos\left(\frac{3\pi}{5}\right) & \cos\left(\frac{4\pi}{5}\right) \\ \sin(0) & \sin\left(\frac{\pi}{5}\right) & \sin\left(\frac{2\pi}{5}\right) & \sin\left(\frac{3\pi}{5}\right) & \sin\left(\frac{4\pi}{5}\right) \end{pmatrix}$$

where $x_{rods}$ and $y_{rods}$ are the rod’s $x$- and $y$-coordinates and $d = 1.75\, \text{in} = 44.45\, \text{mm}$ is the distance of the rods from the container’s cylinder axis. Further denoting the parameter vector

$$\beta = \begin{pmatrix} a \\ b \end{pmatrix},$$

the overdetermined system of equations can be written as

$$X\beta = -z_{rods},$$

where $z_{rods}$ are the measured liquid levels at each rod. The squared error that is to be minimized is

$$(X\beta + z_{rods})^T (X\beta + z_{rods})$$

$$= \beta^T X^T \beta + 2\beta^T z_{rods} + z_{rods}^T z_{rods}$$
Taking the derivative with respect to the parameter vector $\overline{\beta}$ and setting it to zero yields

$$2X^TX\overline{\beta} + 2X^T\overline{z_{rods}} = 0.$$ 

Hence, the parameters $a^*$ and $b^*$ that minimize the squared error are found as

$$\beta^* = \begin{pmatrix} a^* \\ b^* \end{pmatrix} = - \left(X^TX\right)^{-1}X^T\overline{z_{rods}}. \tag{3.13}$$

Once the plane parameters are found, the angles $\theta(t)$ and $\phi(t)$ are given by Eq. (3.3) and (3.4), where

$$P(t) = \begin{pmatrix} P_x(t) \\ P_y(t) \\ P_z(t) \end{pmatrix} = \begin{pmatrix} a^*(t) \\ b^*(t) \\ 1 \end{pmatrix}. \tag{3.14}$$

The sloshing can be measured using `main_VoltDivBasedSloshMeasuring.m`.

### 3.2.3 Results

Figure 3.11 shows the voltage drops across the rods measured by an oscilloscope. The data on the left hand side was collected while the liquid was at rest. Note that nevertheless, the voltage drops are not of equal size. This is due to irregularities of the measurement device such as corrosion on the rods and disk and varying quality of soldered connections. All those effects are accounted for during calibration.

![Figure 3.11](image.png)

Figure 3.11: Data from an oscilloscope measuring the voltage drop across the rods. Liquid at rest (left), and slanted liquid surface (right). Scale: 1ms and 100mV.

A closer look at the voltage readings corresponding to the same rod reveals transient behavior: Due to chemical effects involving the rods, the liquid and the disk, the voltage increases the longer the rod is firing. This means that the exact time at which the voltage is measured matters. The negative consequences of this characteristic are attenuated by taking five measurements for each rod in each cycle as described above. The final voltage for each rod is taken as the mean of the five measurements.

The container was subjected to lateral accelerations and the liquid angles were measured. The resulting data is depicted in Figure 3.12. It is evident that there
is much less noise than with the camera based measurement system, which is mainly due to the inherit smoothing by the least squares plane fitting. Furthermore the data rate is much higher at 1 kHz as opposed to 50 Hz with the camera based measurement system.

![Graph showing voltage and angle variations over time.](image)

Figure 3.12: Results of the voltage divider based measurement system. Top: Voltage drops across the five rods. Bottom: Angles $\theta(t)$ (green) and $\phi(t)$ (blue) of the liquid surface.

An inherent flaw arises because the gap between the rods and the container wall is only $\frac{1}{16}$ in $\approx 1.58 \text{ mm}$ wide. Capillary action can cause liquid to be pulled up into the gap, see Figure 3.13. This can cause additional noise, especially for small liquid angles.

![Image showing liquid being pulled up between the container wall and rod.](image)

Figure 3.13: Due to capillary action the water can be pulled up between the container wall and the rod and cause irregularities.

The main drawback of this measurement system is the necessity of manufacturing a custom liquid container. Moreover, the characteristics of the liquid, the rods and the disk change over time and the calibration is thrown off, which requires frequent calibration.
Chapter 4

Trajectory Generation

The WMR’s position and orientation is described by the state
\[ q = (x(t), y(t), \psi(t))^T. \] (4.1)

Given an initial and final state \( q_0 \) and \( q_f \), the problem of trajectory generation amounts to finding a continuous curve in the state space that connects \( q_0 \) and \( q_f \). As opposed to path planning, where this curve is time-independent, in trajectory generation this curve is parametrized by the time \( t \). Typically the goal is to find trajectories that minimize a cost function composed of energy and time terms and that satisfy constraints such as bounded velocities, accelerations and steering angles.

The minimum-time trajectory generation problem for WMRs that can only achieve radii of curvature above a certain threshold has been solved using Pontryagin’s Maximum Principle for vehicles with unit forward velocity [10] and bounded forward and backward velocity [17], [22]. The problem has also been solved for differential drive vehicles with bounded wheel speeds [2].

A popular approach is to divide the trajectory generation problem into two subproblems by first finding suitable paths and then admissible speed-profiles along these paths [11], [12]. Some of the constraints, such as maximal steering angle, must be taken into account in the path planning stage, other constraints, such as maximum acceleration, have to be dealt with when finding the speed profile.

In the complete approach presented in [12] candidate paths that avoid obstacles are generated. For each of the candidate paths a speed profile that minimizes a cost function is determined. A global optimization routine then finds the path with the lowest associated cost.

In this work the separation approach is used. Let the WMR’s configuration be defined by the state vector \( q \) and let the initial and goal configurations be \( q_0 \) and \( q_f \), respectively. A suitable path is a continuous curve \( q(\tau) \) with \( q(\tau = 0) = q_0 \) and \( q(\tau = 1) = q_f \). A speed profile along the path is a strictly increasing function \( \tau(t) \)
\[ \tau : t \rightarrow [0, 1], \] (4.2)
where \( t \) is the time.
The generation of suitable paths and speed profiles is discussed in the following two sections.

4.1 Path

This thesis focuses on the calculation of speed profiles for given paths. Therefore paths are not determined by an algorithm but chosen by hand. Dubins classic result states that shortest paths with bounded radius of curvature are comprised of straight and circular path segments [10]. This result translates to vehicles in Ackermann configuration because the bounded steering angle introduces bounds on the achievable radius of curvature. However, the shortest paths proposed by Dubins are in general not feasible for vehicles in Ackermann configuration, because the steering angle is discontinuous on transition points between two segments with different radii of curvature. A curve type for which the radius of curvature is continuous and changes linearly along the track is the clothoid. A complete curve can therefore be described by two clothoids: Starting at zero, the steering angle is increased such that the radius of curvature changes linearly, until the maximum steering angle is reached. Then the steering angle is decreased again until it reaches zero. However, the maximum steering angle is much larger for such curves than for circular curves of the same length and total angle, see Figure 4.1.

To overcome this problem, a third curve type is introduced which comprises three segments: A clothoid in the beginning, a circular segment in the middle and a second clothoid in the end. Beginning at zero, the steering angle is increased until the desired steering angle is reached. This maximum steering angle is then held constant over a certain path length before it is decreased again to reach zero, see Figure 4.1 on the right.

Figure 4.1: Steering angle as a function of the path. Circular curve (left), two clothoids (middle) and a circular curve with a clothoid in the beginning and in the end (right). Note that for the clothoid segments, the steering angle does not change linearly.

The paths corresponding to the three types of curves are depicted in Figure 4.2. Paths are generated by hand, comprising straight line segments and any of the three curve types mentioned above.

1Unless the vehicle comes to a complete stop at every connection between two segments with different steering angles.

2This is also how curves of modern roads are built.
Figure 4.2: The three curve types: Circular (top), circular with two clothoids (middle) and two clothoids (bottom). Circular segments are colored in black, clothoids in light and dark blue. All curves have length $L = 2\,\text{m}$ and a total angle of $\frac{\pi}{2}$.

4.2 Speed Profile

The problem of finding a time-optimal speed profile for a given path can be formulated as an optimal control problem using the two ODEs

$$
\begin{align*}
\dot{s}(t) &= v(t) \\
\dot{v}(t) &= u(t),
\end{align*}
$$

where $s(t)$ denotes the travelled path and $v(t)$ the velocity along the path. The vehicle always starts and finishes with zero velocity. Moreover, WLOG, it is assumed that $s(0) = 0$. The problem then consists of finding an input sequence $u(t), t \in [0, T_f]$ that minimizes $T_f$ such that

$$
\begin{align*}
s(T_f) &= S \\
v(T_f) &= 0,
\end{align*}
$$

where $S$ is the total length of the path that is to be travelled and $s(0) = v(0) = 0$. The liquid container introduces the additional constraint of the bounded fluid sloshing. To prevent the liquid from spilling, the angle $\alpha(t)$ between the $z$-axis and the pendulum must not exceed a certain angle $\alpha_{\text{spill}}$. The angle $\alpha(t)$ can be expressed as a function of the pendulum angles

$$
\alpha(t) = \arctan \left( \frac{\sqrt{\cos(\phi(t))^2 \sin(\theta(t))^2 + \sin(\phi(t))^2}}{\cos(\theta(t)) \cos(\phi(t))} \right). \tag{4.6}
$$

However, this relationship proved to be too complicated for the optimization routine to find solutions. For this reason, bounds on the pendulum angles itself are imposed. For this purpose, the set of ODEs (4.3) has to be extended by
adding the dynamics of the damped spherical pendulum. However, the full dynamics are highly non-linear and therefore cause major problems for numerical solvers. To simplify the dynamics, Eq. (2.23) and (2.24) are linearized around the point \((\theta(t), \dot{\theta}(t), \phi(t), \dot{\phi}(t))= (0, 0, 0, 0)^T\) and \(\ddot{v}(t) = u(t)\) is inserted. This yields

\[
\ddot{\theta}(t) = f_1(\theta, \dot{\theta}, \phi, \dot{\phi}, v, u) = -\frac{g\theta(t)}{l_p} - \frac{c_p \dot{\theta}(t)}{l_p^2 m_p} + 2\phi(t)\dot{\phi}(t) - \frac{2v(t)\dot{\phi}(t)}{R(s(t))} + \frac{v(t)^2\theta(t)}{R(s(t))^2} + \frac{u(t)}{l_p} - \frac{\phi(t)u(t)}{R(s(t))},
\]

\[
\ddot{\phi}(t) = f_2(\theta, \dot{\theta}, \phi, \dot{\phi}, v, u) = -\frac{g\phi(t)}{l_p} - \frac{c_p \dot{\phi}(t)}{l_p^2 m_p} + 2v(t)\dot{\phi}(t) - \frac{v(t)^2}{l_p R(s(t))} + \frac{\phi(t)v(t)^2}{R(s(t))^2} + \frac{\theta(t)u(t)}{l_p} - \frac{\theta(t)\phi(t)u(t)}{R(s(t))}.
\]

Introducing the state vector

\[
\dot{q}_{TG}(t) = (x_1(t), x_2(t), x_3(t), x_4(t), x_5(t), x_6(t))^T = (\theta(t), \dot{\theta}(t), \phi(t), \dot{\phi}(t), s(t), v(t))^T,
\]

the full set of ODEs used to calculate a suitable speed profile is given by

\[
\dot{\dot{q}}_{TG}(t) = \begin{pmatrix}
\dot{x}_1(t) \\
\dot{x}_2(t) \\
\dot{x}_3(t) \\
\dot{x}_4(t) \\
\dot{x}_5(t) \\
\dot{x}_6(t)
\end{pmatrix} = \begin{pmatrix}
x_2(t) \\
f_1(q_{TG}, u) \\
x_4(t) \\
f_2(q_{TG}, u) \\
x_6(t) \\
u(t)
\end{pmatrix},
\]

where \(q_{TG}(0) = 0\). The constraints are

\[
\begin{align*}
\min u < u(t) < \max u \\
\min \theta < x_1(t) < \max \theta \\
\min \phi < x_3(t) < \max \phi \\
x_5(T_f) = S \\
x_6(T_f) = 0.
\end{align*}
\]

The path comes into play firstly by the constraint \(x_5(T_f) = S\) and secondly by the radius of curvature which is a function of the travelled path

\[
R = R(s(t)) = R(x_5(t)).
\]

**Solving the Numerical Optimal Control Problem**

Finding a control input \(u(t)\) for the system (4.9) under the constraints (4.10) that minimizes \(T_f\) is very demanding and requires elaborate numerical algorithms. Simple problems of this form can be solved with the MATLAB function
fmincon [4]. However, for the given problem, the calculation time is in the order of days and the program often fails to find a suitable solution. Speed profile generation is implemented using the third-party solver Tomlab for MATLAB instead. Tomlab satisfies the given equations and constraints only in discrete points, the so-called collocation points. The user can define the number of collocation points that should be used by the optimization routine by setting the vector nvec. If nvec has only one entry, the problem is solved once with the given number of collocation points. If the user provides more than one number in nvec the problem is solved once for every entry in nvec using the previous result as an initial guess. This is useful to refine or validate solutions that were found using a smaller number of collocation points. Moreover, the user has to provide a path and the settings $\theta_{\text{max}}$, $\phi_{\text{max}}$, $u_{\text{min}}$, and $u_{\text{max}}$.

```
settings.umin = -1;
settings.theta_max = 15*(pi/180);
settings.phi_max = 15*(pi/180);
settings.nvec = [50, 100, 200];
```

The path is entered by defining a struct for every segment and storing the segments in the vector path. Paths can contain any number of the four segment types straight, circle, clothoid and clothCirc, where the three latter ones correspond to the curve types discussed in section 4.1. The individual segments are defined via a set of parameters, such as their total length (*.length) or the total angle for curve segments (*.angle). For the segment type clothCirc the total length can not be chosen by the user as this results in an overconstrained curve. Instead the user provides the total angle of the segment and the length (*.b) and angle (*.angleCirc) of the circular section in the middle.

```
seg1.type = 'straight';
seg1.length = 1;
seg2.type = 'circle';
seg2.length = 1;
seg2.angle = pi/2;
seg3.type = 'clothoid';
seg3.length = 2;
seg3.angle = pi/2;
seg4.type = 'clothCirc';
seg4.angle = pi/2;
seg4.angleCirc = 3*pi/8;
seg4.b = 1;

% User specified path
path = [seg1, seg4, seg1];
```

The path in the above code segment comprises three segments: A 1 m long straight segment in the beginning and in the end and a circular curve with clothoidal transition segments and a total angle of $\frac{\pi}{2}$. The script main_SpeedProfileGeneration.m is where the user has to enter the path and the settings described above. The routine then autonomously analyzes the given data, formulates the optimal control problem and solves it using the
Tomlab solver.

**Objective Function**

The goal is to find time-optimal speed profiles, which corresponds to the accumulated cost

\[
\int_0^{T_f} g(q^T_G, u, t) \, dt = \int_0^{T_f} 1 \, dt ,
\]

(4.12)

with cost function \( g(q^T_G, u, t) = 1 \). However, for problems with active state constraints this can result in a very quickly changing input \( u(t) \). To get smoother inputs a penalty for the derivative of the input is added and the accumulated cost is

\[
\int_0^{T_f} \tilde{g}(q^T_G, u, t) \, dt = \int_0^{T_f} 1 + \dot{u}(t)^2 \, dt .
\]

(4.13)

The extended cost function \( \tilde{g}(q^T_G, u, t) = 1 + \dot{u}(t)^2 \) is used throughout this work.

**Optimality**

Tomlab uses a pseudospectral method to solve the given problem and not Pontryagin’s Maximum Principle. However, the results are mathematically equivalent [19]. That means the found solutions satisfy necessary conditions for optimality, that is the Hamiltonian is optimal along optimal trajectories. Sufficient conditions for optimality are provided by the Hamilton-Jacobi-Bellman Equation, which is in general very hard to solve. Another caveat of Tomlab is that the equations are only satisfied in a number of discrete points, the collocation points. As discussed above, it is therefore desirable to verify a found solution by repeating the calculation using twice the number of collocation points.
4.3 Results

Straight Path

The speed profile for a straight path with length 2 m is calculated. The bounds on the acceleration are set to $-0.75 \text{m/s}^2 < u(t) < 1.5 \text{m/s}^2$, the bounds on the angles to $|\theta(t)| < 5\,\text{deg}$, $|\phi(t)| < 5\,\text{deg}$. The results are depicted in Figure 4.3. Naturally, the angle $\phi(t)$ is not excited, as no lateral accelerations are occurring. Note that in the first half of the maneuver, the input $u(t)$ is chosen such that the angle $\theta(t)$ smoothly reaches $\theta_{\text{max}}$. Subsequently $\theta(t)$ is held at $\theta_{\text{max}}$ until the de-acceleration phase begins. The required input $u_{ss}$ can be determined analytically:

Inserting $\phi(t) = \dot{\phi}(t) = 0$ in Eq. (4.7) and discarding all terms with $R$ in the denominator, because the vehicle is on a straight path, gives the simplified dynamics of the liquid

$$\ddot{\theta}(t) = -\frac{g\theta(t)}{I_p} - \frac{c_p \dot{\theta}(t)}{I_p^2 m_p} + \frac{u(t)}{I_p}, \quad (4.14)$$

Setting $\dot{\theta}(t) = \ddot{\theta}(t) = 0$ and solving for $u(t) = u_{ss}$ gives the required acceleration to keep the angle $\theta(t)$ at $\theta_{\text{max}}$

$$u_{ss} = g\theta_{\text{max}} = 0.8561 \text{m/s}^2,$$ \quad (4.15)

which exactly corresponds to the forward acceleration that is proposed by the speed profile generation algorithm in the first half of the maneuver. Of course, the same holds in the de-accelerating phase, just with opposite signs. However, because $-0.75 \text{m/s}^2 < u(t)$ this acceleration is too small and $u(t)$ saturates at $u_{\text{min}}$. As a consequence the angle $\theta(t)$ is not kept at $\theta_{\text{min}}$. 

![Figure 4.3: Time-optimal speed profile with constraints on fluid surface angles for a straight path. Top: Speed profile for the WMR. Bottom: Angles $\theta(t)$ (green) and $\phi(t)$ (blue). Bounds on acceleration are $-0.75 \text{m/s}^2 < u(t) < 1.5 \text{m/s}^2$, maximum angles are $|\theta(t)| < 5\,\text{deg}$, $|\phi(t)| < 5\,\text{deg}$ (red).](Image)
Circular Path

The speed profile for a full circle with radius 1 m is calculated. The bounds on the acceleration are set to $-0.75 \text{m/s}^2 < u(t) < 1.5 \text{m/s}^2$, the bounds on the angles are set to $|\theta(t)| < 15 \text{deg}$, $|\phi(t)| < 15 \text{deg}$. The results are depicted in Figure 4.4.

It is evident that the lateral acceleration

$$a_\perp = \frac{v(t)^2}{R(t)} \quad (4.16)$$

is the limiting factor in this case. Note how the velocity is kept constant as soon as the angle $\phi(t)$ reaches $-\phi_{\max}$ in order to keep the lateral acceleration bounded. Similar considerations as in the case for a straight path lead to the velocity at which $\phi(t) = -\phi_{\max}$

$$v(t) = \sqrt{-\phi_{\max} \frac{g}{l_p} \left(-\phi_{\max} - \frac{1}{l_p}\right)^{-1}} = 1.5966 \text{m/s}, \quad (4.17)$$

which is exactly what the speed profile generation algorithm outputs.

Curved Path

Speed profiles are computed for the example path in section 4.2. The maximum acceleration is set to $|u(t)| < 1.5 \text{m/s}^2$. Two speed profiles are determined: One with no bounds on the liquid angles, see Figure 4.5, and one with the bounds set to $\pm 15 \text{deg}$, see Figure 4.6.

For the case without constraints on the fluid angles, the well-known result for time-optimal trajectories with bounded inputs applies: The optimal input is
bang-bang, that is maximum acceleration during the first half of the maneuver and maximum de-acceleration during the second half [5]. For this case, only the last two equations of (4.9) have to be considered and an analytical solution can be found using Pontryagin’s Maximum Principle. Note that the proposed speed profile by the trajectory generation routine slightly differs from bang-bang. Because the cost function $\tilde{g}(q_{TG}, u, t) = 1 + \dot{u}(t)^2$ contains a penalty for the derivative of $u(t)$, the input changes smoothly in the middle of the maneuver.

In the case with constrained liquid angles, the speed profile looks very similar in the beginning and in the end as in the case without bounds. However, in the middle part, when the vehicle is beginning to steer, the two speed profiles diverge. The angle $\phi(t)$ is dependent on the lateral acceleration (4.16). As the vehicle enters the curve and $R(t)$ decreases (recall $R(t) \to \infty$ on a straight line), the velocity needs to be reduced in order to keep $\phi(t)$ bounded. Note the complicated speed profile that is required to keep $\phi(t)$ at $\phi_{min}$ for most of the time in the curve.
Figure 4.6: Time-optimal speed profile with constraints on fluid surface angles for curved path. Top: Speed profile for the WMR. Bottom: Angles $\theta(t)$ (green) and $\phi(t)$ (blue). Maximum acceleration is $|u(t)| < 1.5 \, \text{m/s}^2$, maximum angles are $|\theta(t)| < 15 \, \text{deg}$, $|\phi(t)| < 15 \, \text{deg}$ (red).
Chapter 5

Trajectory Tracking

A control system is designed and implemented that allows the WMR to track the trajectories found with the trajectory generation routine described in section 4. Trajectory tracking for WMRs is a well-known problem and a large body of literature is devoted to the problem. Common methods include feedback linearization [15] and sliding mode control [9] [23]. Both of these approaches produce satisfying tracking results. However, the focus is on the vehicle’s position and heading. Especially for the sliding mode approach, there is typically a transient phase in the beginning during which large accelerations occur. For the goal of this work, these techniques are therefore not suitable, because the main constraint of not spilling the liquid demands very close tracking of the given speed profile. The increasing computational power of personal computers in recent years has led to approaches using model predictive control techniques for trajectory tracking [8]. These approaches are very powerful, bounds on states and inputs can be taken into account. However, due to the computational burden, at this point model predictive control techniques are only applicable for large and relatively slow vehicles: In [8] the sampling time is chosen as large as 0.2 s.

Only close tracking of the given speed and steering profiles allows to draw conclusions with respect to the quality of the pendulum model and the trajectory generation routine. For this reason, the position and orientation of the vehicle are not considered explicitly. Instead, this project focuses on the speed and steering by using two separate SISO control loops. Iterative learning control is used to further improve the tracking performance. An observer is implemented to estimate the current state of the vehicle, measurements are acquired by a motion capturing system. The approach is described in more detail below.

5.1 Observer

In order to estimate the current state of the WMR

\[ q_r(t) = (x(t), y(t), \psi(t), v(t), \gamma(t))^T \]  

introduced in section 2.1, the vehicle is tracked with a motion capturing system. The system comprises eight OptiTrack S250e cameras by NaturalPoint. It returns measurements of the position and the yaw angle of the vehicle every
\[ T_s = \frac{1}{120} \text{s} \]

\[ z_m(kT_s) = (x_m(kT_s), y_m(kT_s), \psi_m(kT_s))^T. \tag{5.2} \]

For the remainder of this document the shorthand notation \( \cdot(k) \) is used instead of \( \cdot(T_s,k) \).

The measurements are used to design an asynchronous extended Kalman filter for the WMR system (2.10). The motion capturing system returns yaw angles in the interval \( (kT_s)_m(\pi, 2\pi] \). If the vehicle is turning such that \( \psi(t) \) exceeds these bounds, the returned angle wraps around by jumping by \( +\pi/2 \). Passing the measurements directly to the observer leads to erroneous behavior, because the vehicle appears to have turned by \( 2\pi \) within one time step. To overcome this problem, wrap-arounds are detected and \( (kT_s)_m \) is corrected such that it is continuous.

In every time step the system (2.10) is linearized around the current operating point

\[
A_o(k) = \frac{\partial f_o(k)}{\partial q_o} = \begin{pmatrix}
0 & 0 & -v(k) \sin(\psi(k)) & \cos(\psi(k)) & 0 \\
0 & 0 & v(k) \cos(\psi(k)) & \sin(\psi(k)) & 0 \\
0 & 0 & 0 & -\frac{L_{wb}}{\mu} & 0 \\
0 & 0 & 0 & 0 & -\frac{a_v}{\tau_v} \\
0 & 0 & 0 & 0 & -\tan(\psi(k))L_{wb} \cos(\gamma(k))^2
\end{pmatrix} \tag{5.3}
\]

and the discrete-time state-space matrix \( F_o \),

\[ q_o(k+1) = F_o(k)q_o(k) \tag{5.4} \]

is calculated as

\[ F_o(k) = e^{A_o(k)T_s}, \tag{5.5} \]

where \( e^M \) is the matrix exponential. The measurement matrix \( H_o \) is given by

\[ H_o = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0
\end{pmatrix}, \tag{5.6}
\]

and the weighting matrices \( Q_o \) and \( R_o \) are chosen as

\[ Q_o = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 10^3 & 0 & 0 \\
0 & 0 & 0 & 10^3 & 0
\end{pmatrix}, \quad R_o = \begin{pmatrix}
2 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 1
\end{pmatrix}. \tag{5.7}
\]

As \( \psi_m(t) \) wraps around, spurious measurements are observed sometimes. To reject this erroneous data, the difference between the current measurement \( \psi_m(k) \) and the average of the past three measurements \( \psi_m(k-1), \psi_m(k-2), \psi_m(k-3) \) is calculated and compared with a predefined threshold. If the difference is bigger than the threshold, the measurement \( \psi_m(k) \) is discarded and \( H_o \) and \( R_o \) are redefined for the current time step as

\[ \tilde{H}_o = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0
\end{pmatrix}, \quad \tilde{R}_o = \begin{pmatrix}
2 & 0 & 0 \\
0 & 2 & 0
\end{pmatrix}. \tag{5.8} \]
Chapter 5. Trajectory Tracking

The prediction of the WMR’s state for the current time step $\hat{q}(k|k-1)$ is calculated using the estimate of the last time step $\hat{q}(k-1|k-1)$ and the WMR’s model (2.10)

$$\hat{q}(k|k-1) = \hat{q}(k-1|k-1) + T_s \cdot f(\hat{q}(k-1|k-1)).$$

The prediction of the variance $P_o(k|k-1)$ is computed using the variance of the last time step $P_o(k-1|k-1)$ as

$$P_o(k|k-1) = F_o(k) \cdot P_o(k-1|k-1) \cdot F_o(k)^T + Q_o.$$ (5.10)

The measurement update is given by

$$K_o(k) = P_o(k|k-1) \cdot H_o \cdot H_o^T \cdot P_o(k|k-1) + R_o \cdot H_o^T \cdot P_o(k|k-1) \cdot H_o \cdot H_o^T \cdot P_o(k|k-1).$$ (5.11)

$$\hat{q}(k|k) = \hat{q}(k|k-1) + K_o(k) \cdot [z_m(k) - H_o \cdot \hat{q}(k|k-1)].$$ (5.13)

5.2 Control Loop

The WMR (2.10) is a MIMO system with inputs $u_v(t)$ and $u_\gamma(t)$ and, depending on the task at hand, any of the states as outputs. For trajectory tracking, this MIMO system is split up into two separate SISO systems: One for the velocity, with input $u_v(t)$ and output $v(t)$ and one for the steering, with input $u_\gamma(t)$ and output $\gamma(t)$. The actual position $(x(t), y(t))$ and heading $\psi(t)$ are not considered explicitly.

The two SISO systems are given by Eq. (2.7) and (2.8). P-Controllers with feed-forward are designed for both systems. The feed-forward commands are chosen as the commands that produce the desired outputs $v_{\text{des}}(t)$ and $\gamma_{\text{des}}(t)$ as the steady state response

$$u_{\text{ff}}(t) = a_v \cdot v_{\text{des}}(t)$$

$$u_{\text{ff}}(t) = a_\gamma \cdot \gamma_{\text{des}}(t)$$

see Figure 5.1. The reference signal for the feedback controller is delayed by $T_{\text{lat}} = 70$ ms. This corresponds to the delay between points 1 and 2 and comprises the delays of the wireless communication, the generation of the PWM signal on board of the WMR and of the motion capturing system. If the reference signal for the feedback controller is not delayed, the feedback controller counteracts the feedforward controller.

Figure 5.1: Control loop for velocity control: Feedback with P-controller and feedforward. The reference signal for the feedback controller is delayed.
Results are depicted in Figure 5.2. It is evident that the tracking is not satisfying, both for the velocity and the steering angle. Because of the limited quality of the measurements and frequent dropouts, the feedback action cannot be made arbitrarily aggressive. To further improve the tracking performance, iterative learning control is added to the control system.

Figure 5.2: Top: Desired speed profile (black) and actual speed (blue). Bottom: Desired steering angle profile (black) and actual steering angle (blue). Both obtained with feedback and feedforward control action.

5.3 Iterative Learning Control

Iterative learning control (ILC) is a method to improve tracking performance of dynamical systems that execute a repetitive task subject to repeatable disturbances. Thereby the error of past iteration is used to improve tracking in the next iteration. In this project, ILC is used to alter the feed-forward commands of the control loop depicted in Figure 5.2. The basic ILC-scheme can be derived as follows [6]:

Consider the discrete-time, linear, time-invariant SISO system

\[
\begin{align*}
x(k+1) &= Fx(k) + Gu(k) \\
y(k) &= Cx(k),
\end{align*}
\]

with the state \(x(k) \in \mathbb{R}^{n \times 1}\), output \(y(k) \in \mathbb{R}\), \(F \in \mathbb{R}^{n \times n}\), \(G \in \mathbb{R}^{n \times 1}\) and \(C \in \mathbb{R}^{1 \times n}\). Introducing the system’s discrete-time transfer function

\[
P(z) = C(zI - F)^{-1}G,
\]

the system (5.16) can be written as

\[
y_j(k) = P(z)u_j(k) + d(k),
\]

where \(k\) is the time index, \(j\) is the iteration index, \(z\) is the forward time-shift operator and \(d(k)\) is the disturbance, which is iteration independent.
The goal is to track a given output $y_d(k)$ with the system output $y_j(k)$. The classic ILC formulation then is

$$u_{j+1}(k) = Q(z) [u_j(k) + L(z)e_j(k + 1)] ,$$  \hspace{1cm} (5.19)

where

$$e_j(k + 1) = y_d(k + 1) - y_j(k + 1)$$  \hspace{1cm} (5.20)

and $Q(z)$ is the so-called Q-filter and $L(z)$ is the learning filter

$$Q(z) = ... + q_{-2}z^2 + q_{-1}z + q_0 + q_1z^{-1} + q_2z^{-2} + ...$$  \hspace{1cm} (5.21)

$$L(z) = ... + l_{-2}z^2 + l_{-1}z + l_0 + l_1z^{-1} + l_2z^{-2} + ... .$$  \hspace{1cm} (5.22)

The input of the next iteration $u_{j+1}(k)$ is given by the input of the current iteration $u_j(k)$ corrected by the filtered error $e_j(k + 1)$. Typically the resulting control input is low pass filtered by applying the Q-filter.

Considering $N$ time steps, this system can be written in the so-called lifted system description as

$$\begin{pmatrix}
y_j(1) \\
y_j(2) \\
\vdots \\
y_j(N)
\end{pmatrix} =
\begin{pmatrix}
p_1 & 0 & \ldots & 0 \\
p_2 & p_1 & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
p_N & p_{N-1} & \ldots & p_1
\end{pmatrix} \cdot
\begin{pmatrix}
u_j(0) \\
u_j(1) \\
\vdots \\
u_j(N-1)
\end{pmatrix} +
\begin{pmatrix}
d(1) \\
d(2) \\
\vdots \\
d(N)
\end{pmatrix},$$  \hspace{1cm} (5.23)

where $p_i = CF^{-1}G$ are Markov parameters and $P \in \mathbb{R}^{N \times N}$ is the lifted system description of $P(z)$. Furthermore denote the error signal

$$\begin{pmatrix}
e_j(1) \\
e_j(2) \\
\vdots \\
e_j(N)
\end{pmatrix} =
\begin{pmatrix}
y_d(1) \\
y_d(2) \\
\vdots \\
y_d(N)
\end{pmatrix} -
\begin{pmatrix}
y_j(1) \\
y_j(2) \\
\vdots \\
y_j(N)
\end{pmatrix}.$$

Then, (5.19) can be written as

$$\begin{pmatrix}
u_j+1(0) \\
u_j+1(1) \\
\vdots \\
u_j+1(N-1)
\end{pmatrix} =
\begin{pmatrix}
q_0 & q_{-1} & \ldots & q_{-(N-1)} \\
q_1 & q_0 & \ldots & q_{-(N-2)} \\
\vdots & \vdots & \ddots & \vdots \\
q_{N-1} & q_{N-2} & \ldots & q_0
\end{pmatrix} \cdot
\begin{pmatrix}
u_j(0) \\
u_j(1) \\
\vdots \\
u_j(N-1)
\end{pmatrix} +
\begin{pmatrix}
l_0 & l_{-1} & \ldots & l_{-(N-1)} \\
l_1 & l_0 & \ldots & l_{-(N-2)} \\
\vdots & \vdots & \ddots & \vdots \\
l_{N-1} & l_{N-2} & \ldots & l_0
\end{pmatrix} \cdot
\begin{pmatrix}
e_j(1) \\
e_j(2) \\
\vdots \\
e_j(N)
\end{pmatrix}.$$

(5.25)
where \( Q, L \in \mathbb{R}^{N\times N} \) refer to the lifted system descriptions of the filters \( Q(z) \) and \( L(z) \).

The ILC system (5.18), (5.19) is said to be asymptotically stable if and only if all entries of \( u_j(k) \) are bounded for \( j = \{0, 1, \ldots, \} \) and if for all \( k \in \{0, \ldots, N-1\} \)

\[
\lim_{j \to \infty} u_j(k) \text{ exists.}
\]

Inserting the definition of the error in (5.25) and subsequently inserting \( y_j = P u_j \) yields

\[
u_{j+1} = Q \left[ u_j + L(y_j - Pu_j) \right]
\]

\[
= Q \left[ u_j - LPu_j + Ly_d \right]
\]

\[
= Q \left[ I - LP \right] u_j + QLy_d
\]

(5.26)

The last term in (5.26) is constant and the well-known stability result is recovered:

The ILC system (5.18), (5.19) is asymptotically stable if and only if

\[
\rho(Q(I - LP)) < 1,
\]

(5.27)

where \( \rho(M) \) is the spectral radius of the square matrix \( M \).

The stability result (5.27) holds if the filters \( Q(z) \) and \( L(z) \) are chosen to be time-varying. The matrices \( Q \) and \( L \) have different parameters \( q_i \) and \( l_i \) on each line in that case.

**Design of the Filters**

Because the filters \( Q(z) \) and \( L(z) \) are applied after each iteration, the complete signals \( u_j(k) \) and \( e_j(k) \) are available and offline filtering techniques can be applied. As opposed to online filtering, zero-phase filters can be implemented (that is the upper right triangles of \( Q \) and \( L \) can contain non-zero elements). That is especially rewarding for the \( Q \)-filter, which is usually chosen as a low pass filter. The difference between standard low pass filtering and zero-phase low pass filtering is highlighted in Figure 5.3.

![Figure 5.3: Input signal (black) and signal filtered with a zero-phase low pass filter (blue) and a standard low pass filter (green).](image)
Separate pairs of filters are designed for the two SISO systems: \( \{Q_v(z), L_v(z)\} \) for the velocity and \( \{Q_\gamma(z), L_\gamma(z)\} \) for the steering angle.

The \( Q \)-filters are chosen as a Gaussian filters. These are weighted moving average filters that have weights that follow a normal distribution. To construct a zero-phase low pass filter \( Q(z) \) a standard low pass filter \( Q_0(z) \) is designed first. The signal is then filtered twice with \( Q_0(z) \), once in forward and once in backward direction. The resulting filter action is the sought zero-phase low pass filter and is given by

\[
Q(z) = Q'(z)Q'(z^{-1}).
\]  

(5.28)

Note that the order of \( Q(z) \) is twice the order of \( Q'(z) \).

\( Q_v'(z) \) is chosen as a 9th order Gaussian filter

\[
Q_v'(z) = b_0 + b_1 z^{-1} + \ldots + b_9 z^{-9},
\]

with

\[
\begin{pmatrix}
b_0 \\
b_1 \\
b_2 \\
b_3 \\
b_4 \\
b_5 \\
b_6 \\
b_7 \\
b_8 \\
b_9 \\
\end{pmatrix} = \begin{pmatrix} 
0.0098 \\
0.0336 \\
0.0849 \\
0.1574 \\
0.2143 \\
0.2143 \\
0.1574 \\
0.0849 \\
0.0336 \\
0.0098
\end{pmatrix}
\]

and \( Q_v(z) \) is calculated according to Eq. (5.28). This procedure is repeated for \( Q_\gamma(z) \), however, the order of the filter is doubled.

Zeroth-order transfer functions are chosen for the learning filters \( L_v(z) \) and \( L_\gamma(z) \). That is in \( L_v \) and \( L_\gamma \) all non-diagonal elements are zero:

\[
l_i = 0, \quad \text{for } i \neq 0.
\]

The learning filter for the steering is chosen as

\[
L_\gamma(z) = l_0 = \frac{1}{2}
\]

(5.30)

throughout the experiments. This renders a diagonal \( L_\gamma \) matrix

\[
L_\gamma = \begin{pmatrix} 
 l_0 & 0 & 0 \\
 0 & \ddots & 0 \\
 0 & 0 & l_0 \\
\end{pmatrix} = \begin{pmatrix} 
 \frac{1}{2} & 0 & 0 \\
 0 & \ddots & 0 \\
 0 & 0 & \frac{1}{2} \\
\end{pmatrix}.
\]

(5.31)

The learning filter for the velocity is chosen time-varying depending on the desired speed profile

\[
L_v = \begin{pmatrix} 
 l_{01} & 0 & 0 \\
 0 & \ddots & 0 \\
 0 & 0 & l_{0N} \\
\end{pmatrix}.
\]

(5.32)
In sections of the profile where the speed changes more rapidly and more detailed trajectories need to be learned, larger gains $l_0$ are chosen. Overall, values between 1 and 3 have proven to be well suited. In general it is desirable to choose small gains to prevent the algorithm from overlearning: When non-repeatable disturbances occur (for example due to erroneous wireless transmission of the control signal) the ILC scheme will try to reject these disturbances in the next iteration even though they will no longer be present. This leads to another tracking error in the next iteration. Less aggressive learning will minimize the negative effects of these non-repeatable disturbances.

Figure 5.4 shows results from applying the ILC scheme to the speed profile for the example path in section 4.2. The sums of the absolute errors of the velocity and steering are decreased from $128.2$ to $9.15 \text{m/s}$ and $16.25$ to $3.28 \text{rad}$, respectively, within 30 iterations. The learning for the steering saturates after about 10 iterations, while the error can be further decreased for the velocity even after 30 iterations.

The trajectory tracking is implemented in the script `main_ILC.m`.

![Figure 5.4: Evolutions of the sum of the absolute errors for the velocity (top) and steering (bottom).](image_url)

![Figure 5.5: Desired velocity $v_{\text{des}}(t)$ and actual speed profiles of different iterations.](image_url)
5.4 Results

Circular Path

The speed profile for the circular path described in section 4.3 is learned and executed. The learning filter for the steering is given by (5.31). The learning filter for the velocity is chosen as

\[
L_v = \begin{pmatrix}
l_0 & 0 & 0 \\
0 & \ddots & 0 \\
0 & 0 & l_0
\end{pmatrix} = \begin{pmatrix}
1.2 & 0 & 0 \\
0 & \ddots & 0 \\
0 & 0 & 1.2
\end{pmatrix}.
\] (5.33)

The trajectory generation algorithm proposes the final time \(T_f = 5.54\) s. Adding a second of standstill in the beginning and in the end, the number of time steps for the maneuver is \(N = 905\). The ILC stability condition (5.27) is satisfied

\[
\rho(Q_v(I - L_vP_v)) = 0.75 < 1
\] (5.34)

\[
\rho(Q_v(I - L_vP_v)) = 0.88 < 1.
\] (5.35)

Results of the trajectory tracking are depicted in Figure 5.6. The tracking for both SISO systems is good, with average absolute errors of 0.0076 m/s for the velocity and 0.0085 rad for the steering angle. The steps in the steering profile in the beginning and in the end of the maneuver cannot be tracked perfectly, because of the physical limitations of the servo mechanism. This could be remedied by allowing the maneuver to start with the desired steering angle. Note the fine ripples in the actual velocity profile that are present throughout the maneuver.

![Graph of steering angle and velocity](image)

Figure 5.6: Experimental results for a circular path with radius 1 m. Desired and actual steering profile (top) and velocity profile (bottom). Note the added second of standstill in the beginning and in the end.

The measured fluid sloshing is depicted in Figure 5.7. In general, the measured fluid angles correspond to the simulated ones. However, overshoots and phase
shifts are present throughout the experiment. Errors are especially striking in sections where the liquid angles are supposed to remain constant, see time interval $t \in [1.5, 3.25]$. In simulation the liquid is regulated such that it approaches these values smoothly. In practice, even minor errors in trajectory tracking or small disturbances alter the behavior of the fluid such that it oscillates around the desired values instead of remaining constant.

Figure 5.7: Experimental results for a circular path with radius 1 m. Desired and actual fluid angles and imposed bounds on the angles (red).

Curved Path

The speed profile for the the example path in section 4.2 is tested. Again, the learning filter for the steering is given by (5.31). A time-varying learning filter is chosen

$$L_v = \begin{pmatrix} l_{0_1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & l_{0_N} \end{pmatrix},$$

(5.36)

where

$$l_{0_i} = \begin{cases} 1 & \text{if } i \in [1, 242] \lor i \in [373, 717] \\ 2.5 & \text{if } i \in [243, 372] \end{cases},$$

(5.37)

which corresponds to more aggressive learning in the middle part where the desired speed profile has finer details. The number of steps is $N = 717$ which corresponds to the proposed final time of $T_f = 3.97 \, \text{s}$. The ILC stability condition (5.27) is satisfied

$$\rho(Q_v(I - L_v P_v)) = 0.75 < 1$$

(5.38)

$$\rho(Q_\gamma(I - L_\gamma P_\gamma)) = 0.9 < 1.$$  

(5.39)
Results of the trajectory tracking are depicted in Figure 5.9. Both the steering and velocity profile are tracked well, with average absolute errors of 0.0123 m/s and 0.0047 rad, respectively. As in the case of the circular path described above, the actual speed profile is not as smooth as the desired one, small ripples can be observed.

The corresponding measurement data of the fluid sloshing is depicted in Figure 5.9. Again, the simulated and actual angles are in general very similar. However, slight differences in the amplitude and phase can be observed. Note how the angle $\phi(t)$ is already excited during the first second, when the vehicle is still going straight. This highlights a general problem of the used WMR: Vibrations caused by the uneven ground or irregularities in the chassis are strong enough to excite the liquid. Also, the basic problem observed in case of the circular path persists: The angle $\phi(t)$ is not kept at the boundary in the interval $t \in [1.5, 2.1]$ as proposed by the trajectory generation algorithm.

The conducted experiments show that the pendulum model approximates the fluid sloshing well. The water’s simulated and actual behaviors are in general very similar, even though there are some imperfections during trajectory tracking. The vehicle and its simple model are identified as the biggest sources of error: The vehicle’s wheels are irregular and out-of-center, causing constant vibrations during the maneuvers. Also the whole body rolls, pitches and yaws when the vehicle is steering. All of these effects are not accounted for by the model (4.9) used in trajectory generation and therefore lead to errors. Because there is no feedback of the fluid’s current state, typically it is not possible to smoothly regulate the pendulum angles such that they saturate at the given bounds as proposed by the trajectory generation algorithm in some parts of the maneuvers. Also, if there is a disturbance in the beginning of the maneuver that affects the water inside the container, its behavior is often completely
overturned during the rest of the maneuver. The liquid angles are typically saturated at the given bounds in the trajectories proposed by the trajectory generation routine. Hence, slight errors during tracking will cause the water to slosh higher than the proposed limits and the goal of not spilling the liquid is essentially not met. To account for the imperfect tracking, the bounds for the liquid angles can be lowered during trajectory generation to include a margin of error.
Chapter 6

Conclusion and Outlook

In the first part of this work models for the vehicle and the liquid sloshing inside the container are developed. The dynamics of liquid in a cylindrical container are best described using partial differential equations. However, in light of controls tasks, this type of model is far too complex and computationally demanding. Instead, a much simpler model is introduced: The liquid dynamics are modeled as a damped spherical pendulum. This entails the assumption that the liquid surface is always planar and always contains a certain point that corresponds to the pendulum’s pivot. The current state of the liquid can therefore be fully described by the two angles that define the liquid surface’s orientation. Experiments are carried out to estimate the equivalent length and damping of the spherical pendulum.

Two measuring systems are proposed to estimate the current state of the liquid. The camera based measuring system is comprised of two cameras facing the vessel. The liquid angles are estimated by determining the highest point of the liquid surface. The voltage divider based measuring system features five copper rods that are partially submerged into the water and a copper disk on the bottom of the container. The rods are connected in series with a known resistor and the pole of a battery. By connecting the disk with the other pole of the battery voltage dividers are created. The potential differences between the rods and the disk are functions of the liquid levels at the rods. A least squares plane is fitted through these five measurement points to get an estimate for the current posture of the liquid’s surface.

The voltage divider based measuring system is found to clearly outperform the camera based system in terms of sample rate and precision. A basic problem of the camera based measuring system is that the precision with which the position of the liquid surface’s highest point can be localized decreases rapidly as the liquid approaches its unexcited state.

Trajectory generation is split up into two separate subproblems: Paths that satisfy the WMR’s kinematic constraints are chosen manually; suitable speed profiles along these paths are then calculated numerically. The proposed algorithm computes speed profiles that satisfy bounds on the acceleration and on the two liquid angles. At the same time the algorithm minimizes a cost function comprising the final time and a penalty on the derivative of the acceleration.
The used method is mathematically equivalent to Pontryagin’s Minimum Principle, and hence satisfies necessary but not sufficient conditions of optimality. Ultimately, the trajectory generation routine proved to work flawlessly.

Experiments are carried out to validate the found trajectories on a real WMR. The voltage divider based measuring system is used to measure the sloshing. An observer is designed that estimates the current state of the vehicle based on measurements of the vehicle’s position and yaw angle acquired by a motion capturing system. The desired steering and velocity profiles are learned using iterative learning control. The liquid angles are found to emulate the simulated behavior in general. That being said, finer details cannot be replicated. This is mainly due to the hobby-grade quality of the used vehicle and the thereby introduced vibrations and inaccuracies. A basic problem lies in the chassis of the WMR, which rolls, pitches and yaws when the vehicle is steering. All of these effects are not accounted for by the model and lead to errors. Nevertheless, considering the strong simplifications introduced by the pendulum assumption and the low quality hardware, the experimental and the simulated data correspond extraordinary well.

The trajectory tracking in this work focuses on closely following the calculated speed and steering profiles, the position and orientation of the vehicle are not taken into account explicitly. This allows to draw conclusions regarding the quality of the trajectory generation routine and of the pendulum model for the fluid sloshing at the expense of allowing drift in both position and orientation. Further research could deal with extensions of existing trajectory tracking methods for WMRs to include feedback of the current state of the fluid as well as position and orientation of the vehicle. An interesting question is how to handle the absolute bounds on the fluid sloshing online, while minimizing the tracking error. Because the steering has a big influence on the fluid sloshing, follow-up work for the trajectory generation algorithm could tackle a major challenge: The overall performance of the system might be increased by solving the full trajectory generation problem instead of separating it into path and speed-profile generation. Due to the complexity of the model, analytical approaches might not be applicable. However, the proposed numerical trajectory generation routine could be extended to include path generation as well.
Appendix A

Technical Drawings and Pictures
Figure A.1: The main part of the voltage divider based slosh measuring system.
Figure A.2: Front view (top) and side view of the vehicle (bottom).
Appendix B

List of Submitted Code

Camera Based Slosh Measuring
Main File: main_CameraBasedSloshMeasuring.m
Additional Files:
- calcAngles.m
- calcPendulumPosition.m
- fitLine.m
- getSteadyStateParams.m
- intersectOpticalRayWithCylinder.m
- loadParameters.m
- plotPendulum.m

Voltage Divider Based Slosh Measuring
Main Files:
- main_VoltDivBasedSloshMeasuring.m
- main_performCalibration.m
Additional Files:
- calcAngles.m
- decodeData.m
- receiveData.m
- SloshMeasuring.ino

Trajectory Generation
Main File: main_CalculateSpeedProfile.m
Additional Files:
- analyzePath.m
- calcSteerAngle.m
- calcYawAngle.m
- completeFields.m
- generatePlots.m
- generateTOMLABProblem.m
- getTfVec.m
setUpPhase.m
setUpPhaseClothCirc.m
setUpPhaseClothoid.m

Trajectory Tracking
Main File    main_ILC.m
Additional File    VehicleControl.ino
Bibliography


Title of work:
Trajectory Generation for an Ackermann Vehicle Carrying a Dynamic Load

Thesis type and date:
Master Thesis, October 2012

Supervision:
Prof. Dr. John Baillieu
Prof. Dr. Raffaello D'Andrea
Dr. Angela Schöllig

Student:
Name: Benjamin Troxler
E-mail: troxlebe@student.ethz.ch
Legi-Nr.: 06-918-544
Semester: 10

Statement regarding plagiarism:
By signing this statement, I affirm that I have read the information notice on plagiarism, independently produced this paper, and adhered to the general practice of source citation in this subject-area.

Information notice on plagiarism:
http://www.ethz.ch/students/semester/plagiarism_s_en.pdf

Zurich, 12. 10. 2012: [Signature]