Control of Outdoor Robots at Higher Speeds on Challenging Terrain

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

M.Sc. Goran Huskić

aus Sarajevo, Bosnien-Herzegowina

Tübingen

2018

Tag der mündlichen Qualifikation:	24.07.2018
Dekan:	Prof. Dr. Wolfgang Rosenstiel
1. Berichterstatter:	Prof. Dr. rer. nat. Andreas Zell
2. Berichterstatter:	Prof. Dr. rer. nat. Andreas Schilling

For my loved ones

Abstract

This thesis studies the motion control of wheeled mobile robots. Its focus is set on high speed control on challenging terrain. Additionally, it deals with the general problem of path following, as well as path planning and obstacle avoidance in difficult conditions.

First, it proposes a heuristic longitudinal control for any wheeled mobile robot, and evaluates it on different kinematic configurations and in different conditions, including laboratory experiments and participation in a robotic competition.

Being the focus of the thesis, high speed control on uneven terrain is thoroughly studied, and a novel control law is proposed, based on a new model representation of skid-steered vehicles, and comprising of nonlinear lateral and longitudinal control. The lateral control part is based on the Lyapunov theory, and the convergence of the vehicle to the geometric reference path is proven. The longitudinal control is designed for high speeds, taking actuator saturation and the vehicle properties into account. The complete solution is experimentally tested on two different vehicles on several different terrain types, reaching the speeds of $\approx 6\,\mathrm{m\,s^{-1}}$, and compared against two state-of-the-art algorithms.

Furthermore, a novel path planning and obstacle avoidance system is proposed, together with an extension of the proposed high speed control, which builds up a navigation system capable of autonomous outdoor person following. This system is experimentally compared against two classical obstacle avoidance methods, and evaluated by following a human jogger in outdoor environments, with both static and dynamic obstacles.

All the proposed methods, together with various different state-of-the-art control approaches, are unified into one framework. The proposed framework can be used to control any wheeled mobile robot, both indoors and outdoors, at low or high speeds, avoiding all the obstacles on the way. The entire work is released as open-source software.

Kurzfassung

Diese Arbeit befasst sich mit Bewegungsregelung von fahrenden Robotern. Der Schwerpunkt liegt auf der Regelung bei hohen Geschwindigkeiten auf schwierigem Terrain. Dabei wird die Pfadverfolgung, sowie Pfadplanung und Hindernisvermeidung unter anspruchsvollen Bedingungen behandelt.

Zuerst wird ein Algorithmus zur Längsregelung von fahrenden Robotern vorgestellt, der auf verschiedenen kinematischen Antrieben und unter verschiedenen Bedingungen evaluiert wurde, inklusive Laborexperimenten und der Teilnahme an einem Roboterwettbewerb.

Als Schwerpunkt der Arbeit wurde Fahrregelung bei hohen Geschwindigkeiten auf unebenem Terrain gründlich erforscht, wobei als Ergebnis ein neuartiges Regelungsgesetz enwickelt wurde, welches auf einer neuen Modelldarstellung von Kettenantrieb basiert, und aus einer nichtlinearen Längs- und Querregelung gebildet wird. Die Querregelung basiert auf der Lyapunov-Theorie, wobei die Konvergenz des Fahrzeugs zum geometrischen Referenzpfad bewiesen wurde. Die Längsregelung wurde für hohe Geschwindigkeiten entwickelt, wobei die Sättigung der Aktoren, sowie die Eigenschaften des Fahrzeugs berücksichtigt sind. Die Gesamtlösung wurd experimentell auf zwei verschiedenen Fahrzeugen und auf mehreren verschiedenen Terraintypen ausgewertet. Der Ansatz wurde mit zwei anderen Algorithmen aus dem aktuellen Stand der Technik verglichen, wobeei Geschwindigkeiten von $\approx 6\mathrm{m\,s^{-1}}$ erreicht wurden.

Weiterhin wird ein neues System für Pfadplanung und Hindernisvermeidung vorgestellt, zusammen mit einer Erweiterung des vorgestellten Regelungsalgorithmus für hohe Geschwindigkeiten, was insgesamt ein Navigationssystem zur autonomen Personenverfolgung im Außenbereich bildet. Dieses System wurde experimentell mit zwei klassischen Hindernisvermeidungsmethoden verglichen, und durch Verfolgung eines Läufers im Außenbereich mit sowohl statischen als auch dynamischen Hindernissen ausgewertet.

Alle vorgestellten Methoden, zusammen mit zahlreichen Regelungsalgorithmen aus dem aktuellen Stand der Technik, sind in ein System vereinigt. Dieses System kann jeden fahrenden Roboter ansteuern, im Innen- und Außenbereich, bei niedrigen oder hohen Geschwindigkeiten, wobei allen Hindernissen ausgewichen wird. Die ganze Arbeit ist quelloffen verfügbar.

Acknowledgments

I would like to thank Prof. Dr. Andreas Zell for being my PhD supervisor, and for supporting and leading me through the entire studies.

I would also like to thank Prof. Dr. Andreas Schilling for his evaluations and recommendations of my work through the past years, making the continuity of my studies possible. I am grateful to the German Academic Exchange Service (DAAD) for providing me their PhD scholarship, organizing many events and helping in many ways. I would also like to thank Vita Serbakova for her support and help from the very beginning, as well as Klaus Beyreuther for his friendly approach and competent assistance.

My special thanks goes to Sebastian Buck, my friend and colleague, who has been highly valuable in both academic and private life. The most of the underlying work we have done together. Many thanks to Dr. Nedim Šrndić, my great friend, for his unselfish help and support. I am also grateful to Dr. Sebastian Scherer, Dr. Sebastian Otte, Adrian Zwiener, Richard Hanten, Julian Jordan, and Radouane Ait Jellal, as well as other colleagues and friends.

To Dr. Simon Lacroix and Matthieu Herrb I am grateful for their scientific feedback and a very friendly and fruitful collaboration. To the research institute LAAS-CNRS in Toulouse, France, I am grateful for being open to academic exchange and making my research there possible.

The most important gratitude for everything goes to my parents, my brother and Marina, and my better half Lamija.

Contents

Chapter 1

Introduction

The rapid technology development is changing our lives every day. Robotic systems are increasingly employed in many applications that were initially meant only for humans. Apart from stationary robots, which revolutionized the ways of production, mobile robots are becoming increasingly popular. In industrial plants, mobile robots have become a cheaper and a much more flexible solution than the classic conveyor belts. Automated vehicles are undertaking agricultural tasks, while planetary exploration can hardly be imagined without space rovers. Inspection of hazardous areas with mobile robots saves human operators from the risk of being exposed. Many mobile robots find their role in surveillance, or search and rescue applications, and service robots are being employed in museums, restaurants, or retirement homes. Transportation in general is in a highly dynamic and changing phase. Many car manufacturers, together with the suppliers and research institutes, are working towards the common goal of making motor vehicles autonomous. In this case, a car would not be a machine operated exclusively by a human, but rather an autonomous mobile robot capable of solving complex tasks and making decisions by itself. Obviously, this would be a revolution, and it would completely change the ways of transportation, and the very concept of a motor vehicle.

One of the main tasks for mobile robots is motion control. It is crucial to have a safe, stable and smooth motion, regardless of the task. If the robot has a large deviation from the planned path, not only the task would be jeopardized, but a failure of unpredictable proportions could happen. If a planetary rover would deviate from the path, and fall into a ditch, many years of research and development, as well as a huge financial investment would fail. If an autonomous car would be in a situation which requires reducing its speed and braking in front of a pedestrian, it would be of utmost importance for the longitudinal control to work properly. In order to act properly in such critical situations, and to meet the performance requirements, a careful design of motion control laws needs to be made, taking the desired motion into account.

When designing a robust and reliable motion control law to deal with dynamic and cluttered environments, a number of parameters and constraints need to be considered. Usually, both kinematic and dynamic model need to be employed. Modelling the dynamics of the vehicle and of the environment usually includes a complex and tedious procedure. Such a modelling needs to be made for each different robot configuration

individually, and the resulting model applies only for the specific vehicle. The question which arises here is: *Is there a control approach applicable to all kinematic configurations, without having the need of complex modelling and control design, while still providing satisfactory results?*

Furthermore, motion control gets increasingly complex and challenging as the driving speed increases, and as the terrain gets more uneven and slippery. This is a common case in agricultural, exploration, search and rescue, mining, or surveillance tasks. Such off-road scenarios are impossible for some kinematic configurations, such as omnidirectional drives, since they can only move on flat ground. Vehicles with Ackermann steering are capable of coping with such conditions, but the complexity of the steering system makes them expensive to produce and prone to defect. Another, special type of kinematics, the so called *skid steering*, has proven to be very robust, maneuverable, and relatively cheap to produce. There is no mechanical steering system, and there are no moving parts. In order for the vehicle to steer, the individual wheels need to be controlled with different speeds. Such vehicles are very well suited for outdoor scenarios, and make the majority of off-road solutions on the mobile robots market.

However, even though skid-steered robots offer many advantages, they are difficult to model and to control. The kinematic model includes dynamics implicitly, and the motion is terrain-dependent. To model the vehicle motion, complex interactions between the wheels and the ground need to be taken into account, which include e.g. friction and torsion. Most of the work on modelling and control of such vehicles deals with low or moderate speeds. The question which arises here is: *Is it possible to accurately and robustly control skid-steered vehicles at high speeds on rough terrain?*

Off-road applications are usually highly dynamic, including static and dynamic obstacles, as well as rough terrain. Additional difficulty in these conditions is human-robot interaction. An example application incorporating the main challenges of off-road conditions would be a jogger-following scenario, in which a robot accompanies a human jogger, keeping the desired distance and speed, while avoiding all the obstacles on the way. Here the question is: *Is there a navigation system allowing a robot to follow a human at higher speeds in all-terrain conditions, while avoiding both static and dynamic obstacles?*

Another important issue in the field of mobile robots is having a unified navigation framework. The first question we have posed deals with the issue of having a control law applicable to all kinematic configurations. Having a simple and robust solution is beneficial, but having different advanced control solutions for all kinematic configurations unified in one system would be of a great value, whether it is for research or educational purposes. In this way, for each special configuration, there would already exist a model and at least one control law dealing with robust motion control. The final question we pose is: *Is there a unified navigation system for all wheeled mobile robots?*

1.1 Contributions

The initial goal of this thesis was to develop a navigation system which would allow a wheeled mobile robot to follow a jogger in rough terrain conditions, while simultaneously avoiding static and dynamic obstacles on the way. A solution to this problem is proposed here, as well as solutions to the rest of the previously posed questions. The contributions can be listed as follows:

- Proposing a new longitudinal control for any wheeled mobile robot, which provides a robust and straightforward solution for safe and stable navigation in cluttered and dynamic environments. The proposed control is fused with the approach proposed in (Mojaev and Zell, 2004) and experimentally evaluated in two different manners: in the robotic competition SICK Robot Day 2014, and in laboratory conditions.
  - In the competition, a warehouse scenario was simulated. The goal was to fetch and deliver special objects during a specified time window, while driving in an arena together with three other robots at the same time. At this competition, among 14 international teams, our team Attempto Tübingen won the 2nd place.
  - In the laboratory conditions, an Ackermann-steering, a bi-steerable, and an omnidirectional robot were used to evaluate the approach on different kinematic configurations. The results are compared against the approach proposed in (Mojaev and Zell, 2004), and further developed in (Li *et al.*, 2007).
- Proposing a new control approach for skid-steered vehicles at high speeds on challenging terrain types. The approach consists of two nonlinear control laws: a lateral and a longitudinal one. The individual contributions are as follows:
  - A new representation of skid-steered vehicles in path coordinates is introduced.
  - A new kinematic control law based on the Lyapunov theory is proposed, which deals with the convergence of the vehicle to the path.
  - A new longitudinal control scheme based on the kinematic parameters is introduced, taking into account actuator saturation, convergence to the path, and the asymmetry of steering.
  - The complete solution is thoroughly tested on two different skid-steered vehicles, a Robotnik Summit XL robot, and a Segway RMP 440 robot, on several different terrain types and at speeds up to $6\,\mathrm{m\,s^{-1}}$, comparing the performance against two other state-of-the-art algorithms.
- Developing a fully autonomous system for person following in all-terrain conditions, at higher speeds, using skid-steered vehicles. The system consist of:

  - a new path planning approach, which is able to find locally optimal paths, taking into account kinematic constraints and obstacles on the way, even at higher speeds,
  - a control approach, which is an extension of the previously mentioned high speed control approach for skid-steered vehicles.

  The whole system is evaluated in two different ways:

  - static obstacle avoidance,
  - jogger following, while avoiding both static and dynamic obstacles.

  Both experimental parts were conducted outdoors, while the second one includes all-terrain conditions. In the experiment with unknown static obstacles, the proposed approach is compared against two classical obstacle avoidance methods. In the jogger following scenario, the goal was to show human-robot interaction in rough conditions. The terrain conditions include flat, uneven, dry, wet, or even snowy and icy terrain. The environment conditions include cluttered and very dynamic situations with pedestrians, dogs, cars, etc.

- Implementing twelve different path following algorithms dealing with different kinematic configurations and integrating them in a unified navigation framework for any wheeled mobile robot. The framework is written in the Robot Operating System (ROS) framework, which is described in (Quigley *et al.*, 2009), and offers many more features than the standard ROS Navigation Stack, described in (Marder-Eppstein, 2017b). With the proposed framework, it is possible to control any wheeled mobile robot using ROS, regardless of its kinematic configuration. The framework offers motion control at low and high speeds with obstacle avoidance, both indoors and outdoors. All the above listed contributions are a part of this framework and the complete code is open-source available.

1.2 Structure of the Thesis

After Chapter 1, the remainder of this thesis is organized as follows. In Chapter 2 some basics on motion control of mobile robots are given. First, motion control of mobile robots is defined and classified. Then, some basic Lyapunov theory is introduced, where the terms like autonomous system, stability or equilibrium are introduced. Furthermore, the backstepping technique is introduced as an important nonlinear control design method. These theoretical basics are mostly based on (Khalil and Grizzle, 1996) and (Slotine *et al.*, 1991). In Chapter 3, we describe the details of the first contribution listed above. The chapter is based on the following publications:

- Huskić, G., Buck, S. and Zell, A., 2016, July. A Simple and Efficient Path Following Algorithm for Wheeled Mobile Robots. In International Conference on Intelligent Autonomous Systems (pp. 375-387). Springer, Cham.

- Buck, S., Hanten, R., Huskić, G., Rauscher, G., Kloss, A., Leininger, J., Ruff, E., Widmaier, F. and Zell, A., 2015, July. Conclusions from an object-delivery robotic competition: Sick robot day 2014. In Advanced Robotics (ICAR), 2015 International Conference on (pp. 137-143). IEEE.

In Chapter 4 a novel high speed control approach for skid-steered vehicles on challenging terrain is introduced. Furthermore, two state-of-the-art algorithms are described, used for the comparison against the proposed approach. The first state-of-the-art algorithm is proposed in (Soetanto *et al.*, 2003), and extended in (Indiveri *et al.*, 2007). The second one is proposed in (Pentzer *et al.*, 2014b), based on (Pentzer *et al.*, 2014a). Additionally, an extension of the proposed approach is described, utilizing both the kinematic and the dynamic model of skid-steered vehicles. The control design is based on backstepping. The chapter is mostly based on the work described in:

- Huskić, G., Buck, S. and Zell, A., 2017, May. Path following control of skid-steered wheeled mobile robots at higher speeds on different terrain types. In Robotics and Automation (ICRA), 2017 IEEE International Conference on (pp. 3734-3739). IEEE.

- Huskić, G., Buck, S., Herrb, M., Lacroix, S. and Zell, A., High-Speed Path Following Control of Skid-Steered Vehicles. In International Journal of Robotics Research. (submitted)

In Chapter 5 we propose our autonomous system for tracking dynamic objects (e.g. humans) in all-terrain conditions, at higher speeds, using skid-steered vehicles. Furthermore, two classical obstacle avoidance methods are described, against which the proposed approach is compared. These classical obstacle avoidance methods are proposed in (Koren and Borenstein, 1991) and (Fox *et al.*, 1997). The chapter is mostly based on the publication:

- Huskić, G., Buck, S. and Zell, A., 2017, September. Outdoor Person Following at Higher Speeds using a Skid-Steered Mobile Robot. In Intelligent Robots and Systems (IROS), 2017 IEEE International Conference on. IEEE.

In Chapter 6 we describe a unified and generic navigation framework for wheeled mobile robots. The framework contains all the approaches described in the previously chapters, and in this chapter, the remaining approaches are described, as well as the framework in general. The chapter is mostly based on:

- Huskić, G., Buck, S. and Zell, A. GeRoNa: Generic Robot Navigation. In Journal of Intelligent & Robotic Systems. (submitted)

In Chapter 7, the work is summarized and the results are discussed.

Research described in this thesis included practical work with different robotic systems. All of these robots are presented in Figure 1.1.

Figure 1.1: *All the robots used for the research described in this thesis.*

Chapter 2

Motion Control of Mobile Robots

2.1 Motion Control Classification

Motion control of mobile robots deals with the basic issue of converting ideal motion plans into motion execution. One approach would be to directly set the planned motion as command. However, it is well known that such open-loop control is not robust to modelling errors, nor to any external disturbances. This is why the more reasonable approach would be to use methods based on feedback control. These methods suppose that the variables involved in the control loop are measurable by real-time sensors. Usually, those variables are the position and the orientation, and they represent the state of the robot, as measured by proprioceptive (e.g. odometry), or exteroceptive sensors (e.g. laser scanner). Control inputs of wheeled mobile robots are usually linear and rotational velocities, when controlling a kinematic model. If dealing with a dynamic model, usually wheel torques are used.

Motion planning strategies have a feasible reference state trajectory as an output, and motion control answers the question how to make the physical mobile robot track exactly this reference trajectory by using the actuators with which the vehicle is equipped. The reference state trajectory can consist of only one pose, a set of poses, or a set of poses with a timing law assigned to it.

Depending on this, motion of wheeled mobile robots can be classified into three tasks:

- Point stabilization: Starting from a given initial configuration, the robot needs to reach a desired goal configuration.
- Path following: Starting from a given initial configuration, the robot needs to follow a geometric path.
- Trajectory tracking: Starting from a given initial configuration, the robot needs to follow a trajectory (geometric path with an associated timing law).

Discussions on motion control of wheeled mobile robots and its classification can be found in (De Luca *et al.*, 1998), (Morin and Samson, 2008), (Aguiar and Hespanha, 2007), and (Soetanto *et al.*, 2003).

2.1.1 Point Stabilization

Point stabilization is the problem of controlling the robot to reach the desired configuration by using only the current error information, without the need of planning a trajectory between the initial and the desired point. Point stabilization for nonholonomic mobile robots is considerably more difficult than path following and trajectory tracking, since the problem cannot be solved with a smooth pure state feedback, as stated in Brockett *et al.* (1983). This problem can be understood by considering a parallel parking maneuver of a car, i.e. bringing the error to zero in the lateral direction, along which the car cannot move directly. As a solution, there are two main approaches: discontinuous and hybrid feedback laws, as e.g. (Aguiar *et al.*, 2000) and (Hespanha and Morse, 1999), and smooth time-varying control laws, as e.g. (de Wit *et al.*, 1993) and (Godhavn and Egeland, 1997).

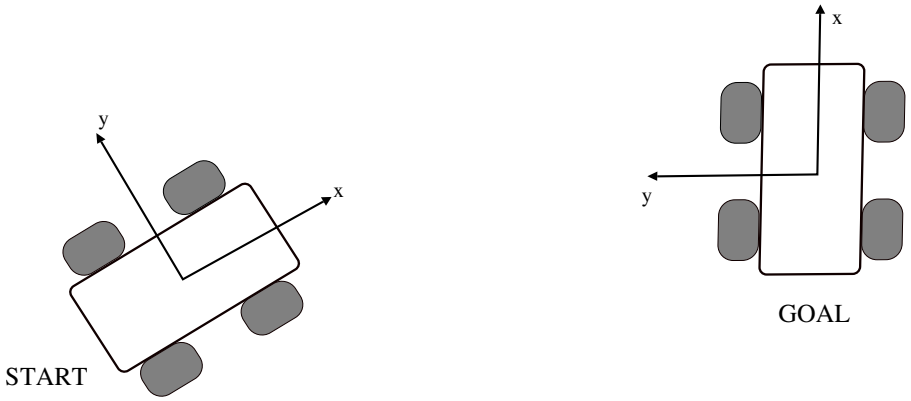


Figure 2.1: *Point stabilization: moving from one configuration to another.*

2.1.2 Trajectory Tracking

Tracking a geometric reference parametrized with time becomes especially challenging in the case of underactuated vehicles, i.e. the ones that have less actuators than state variables. As stated in (Aguiar and Hespanha, 2007), the classical approach for trajectory tracking of underactuated vehicles uses local linearization and decoupling of the multivariable model to steer the same number of degrees of freedom as the number of available control inputs. This can be done by using standard linear, or nonlinear control methods. Alternative approaches use linearization of the vehicle error dynamics around trajectories that lead to LTI (linear time-invariant) systems. This is combined with gain scheduling and linear parameter varying design methods, which can be found in e.g. (Kaminer *et al.*, 1998). These approaches guarantee stability only in a neighbourhood of the selected operating points. Another approach is to use feedback linearization methods, such as in e.g. (Walsh *et al.*, 1994). Furthermore, Lyapunov-based methods, such as the one

in (Aguiar and Hespanha, 2007), can overcome some of the limitations of the already mentioned approaches.

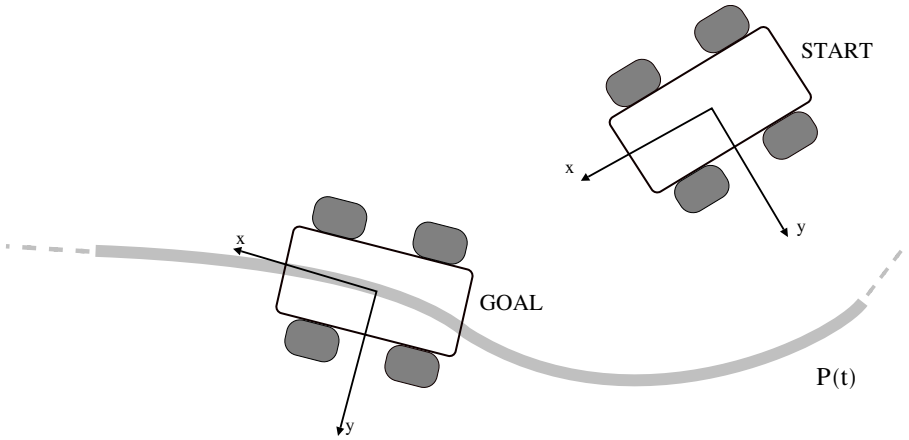


Figure 2.2: *Trajectory tracking: tracking a geometric path P parametrized with time t.*

2.1.3 Path Following

In path following, the robot needs to converge to and follow a geometric path that is specified without a temporal law. Pioneer work can be found in (Samson, 1995), (Samson, 1992), (de Wit *et al.*, 1993) and (Kanayama *et al.*, 1990). Further work can be found in (Jiang and Nijmeijer, 1999) and (Micaelli and Samson, 1993). There is an underlying assumption in the path following problem, which states that the robot's forward speed tracks a desired speed profile, while the controller acts on the robot's orientation to drive it to the path. When compared to trajectory control laws, in path following smoother convergence to the path is achieved, and the control signals are less likely pushed to saturation. The fundamental difference between path following and trajectory tracking can be found in (Aguiar *et al.*, 2005) and (Aguiar *et al.*, 2004).

This thesis deals with the problem of path following, since in dynamic and outdoor environments it is the most reasonable choice. For instance, while driving outdoors and avoiding obstacles on the way, it is only important to have a smooth and safe navigation, i.e. geometric following. The timing law does not have to be pre-specified. It is more reasonable if the speed is a free control parameter. More about the related work on path following using different mobile robot configurations can be found in Section 3.1.

The geometric reference path that needs to be followed is parametrized by some arbitrary path parameter. This parameter is usually chosen to be the arc length of the path. To describe the relationship between the robot and the reference path, the most commonly used approach is the one based on the Serret-Frenet frame moving along the reference path, consisting of three vectors: tangent, normal and binormal to the curve. Since the

motion of a wheeled mobile robot is mostly considered planar, the Serret-Frenet frame consists of the tangent and normal vectors.

By using this approach, it is possible to define the Serret-Frenet frame either in the orthogonal projection from the robot to the path, or on an arbitrary point on the path in front of the robot. The latter one is then considered to be a *virtual vehicle*, i.e. a desired configuration on the path which needs to be reached and tracked. The difference between these methods can be found in e.g. (Płaskonka, 2015). In this thesis, several control algorithms are described, using both of the Serret-Frenet constructions.

After the relationship between the robot and the reference path is modelled in such a way, usually a Lyapunov-based approach is employed, since the model described in the newly constructed Serret-Frenet frame is exactly the error model. This allows a straightforward construction of a Lyapunov function candidate, where the equilibrium is reached when the robot converges to the path.

The methods that use the Serret-Frenet framework together with Lyapunov theory usually include kinematic modelling. This can be extended by modelling the full dynamic behaviour. Furthermore, it is possible to design a pure geometric controller, without explicitly modelling the vehicle. In this case, the control becomes generic and can be used for any model.

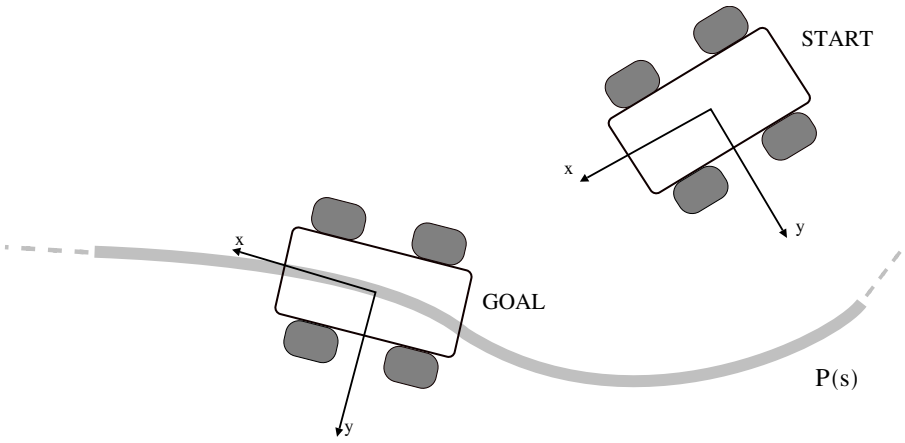


Figure 2.3: *Path following: following a geometric path P parametrized with arc length s.*

2.2 Lyapunov Stability Basics

In order to deal with motion control of mobile robots, especially path following, some basics of nonlinear control theory need to be introduced. These basics and further details can be found in (Khalil and Grizzle, 1996) and (Slotine *et al.*, 1991).

Stability theory is a key concept in systems theory and engineering in general, especially in motion control of mobile robots. Fur the purpose of the work presented in this

thesis, some basic notions in stability of equilibrium points are given. In control theory in general, there are other kinds of stability, such as input-output stability, or stability of periodic orbits.

In the case of stability of equilibrium points, it is usually meant in the sense of Lyapunov, a Russian mathematician and engineer who developed the concept. An equilibrium point can be understood as a point of balance. In the mathematical sense, it is a solution which is stable, if all solutions starting at nearby points are stable as well. Otherwise, it is unstable. Furthermore, it is asymptotically stable if all solutions starting at neighbouring points not only stay in the neighbourhood, but also tend to the equilibrium point as time approaches infinity.

2.2.1 Fundamental Terms

A finite-dimensional dynamical system can be described with a so-called unforced state equation

$$\dot{x} = f(t,x), \tag{2.1}$$

where x is the state, and t is the time. A special case of system (2.1) arises when the function f does not explicitly depend on time t, so it becomes an autonomous, or time invariant system

$$\dot{x} = f(x). \tag{2.2}$$

An important concept in dealing with the state equation is the *equilibrium point*. An equilibrium point is the point $x = x^*$ in state space, if it has the property that if the system starts at x^*, it will remain at x^* for all future time. For the system (2.2), the equilibrium points are the real roots of the equation

$$f(x) = 0. \tag{2.3}$$

A constraint is imposed to the system defined in Eq. (2.1), named the *Lipschitz condition*, where $f(t,x)$ satisfies the inequality

$$\|f(t,x) - f(t,y)\| \leqslant L\|x - y\|, \tag{2.4}$$

for all (t,x), and (t,y) in some neighbourhood of the initial values (t_0,x_0). The function $f(t,x)$ is said to be Lipschitz in x, and the positive constant L is called a *Lipschitz constant*.

Lemma 1. *If $f(t,x)$ and $[\partial f/\partial x](t,x)$ are continuous on $[a,b] \times D$, for some domain $D \subset \mathbb{R}^n$, then f is locally Lipschitz in x on $[a,b]xD$.*

Lemma 2. *If $f(t,x)$ and $[\partial f/\partial x](t,x)$ are continuous on $[a,b] \times \mathbb{R}^n$, then f is globally Lipschitz in x on $[a,b] \times \mathbb{R}^n$ if and only if $[\partial f/\partial x]$ is uniformly bounded on $[a,b] \times \mathbb{R}^n$.*

2.2.2 Stability of Autonomous Systems

Let us consider the autonomous system from Eq. (2.2), where $f : D \to \mathbb{R}$ is a locally Lipschitz map from a domain $D \subset \mathbb{R}^n$ into $\mathbb{R}^n$. Let us suppose that the equilibrium is at the origin $x = 0$, and $f(x)$ satisfies $f(0) = 0$. This can be done without a loss in generality, because any equilibrium point can be shifted to the origin by changing the variables.

Definition 1. *The equilibrium point $x = 0$ of Eq. (2.2) is*

- *stable if, for each $\varepsilon > 0$, there is a $\delta = \delta(\varepsilon) > 0$ such that*

$$\|x(0)\| < \delta \Rightarrow \|x(t)\| < \varepsilon, \forall t \geqslant 0 \tag{2.5}$$

- *unstable if it is not stable.*
- *asymptotically stable if it is stable and δ can be chosen such that*

$$\|x(0)\| < \delta \Rightarrow \lim_{t\to\infty} x(t) = 0. \tag{2.6}$$

To show that the origin is stable, for any value of ε, a value of δ needs to be produced, possibly dependent on ε, such that a trajectory starting in a δ neighbourhood of the origin will never leave the ε neighbourhood. Now we can state Lyapunov's stability theorem.

Theorem 1. *Let $x = 0$ be an equilibrium point for (2.2) and $D \subset \mathbb{R}^n$ be a domain containing $x = 0$. Let $V : D \to \mathbb{R}$ be a continuously differentiable function such that*

$$V(0) = 0 \ \text{ and } \ V(x) > 0 \ \text{ in } \ D - \{0\} \tag{2.7}$$

$$\dot{V}(x) \leqslant 0 \ \text{ in } \ D \tag{2.8}$$

Then, $x = 0$ is stable. Moreover, if

$$\dot{V}(x) < 0 \ \text{ in } \ D - \{0\} \tag{2.9}$$

then $x = 0$ is asymptotically stable.

A continuously differentiable function $V(x)$ satisfying (2.7) and (2.8) is called a *Lyapunov function*. A function $V(x)$ satisfying condition (2.7), i.e. $V(0) = 0$ and $V(x) > 0$ for $x \neq 0$ is said to be *positive definite*. If it satisfies a weaker condition $V(x) \leqslant 0$ for $x \neq 0$, it is *positive semidefinite*. A function $V(x)$ is said to be *negative definite* or *negative semidefinite* if $-V(x)$ is a positive definite or positive semidefinite, respectively. Now Lyapunov's theorem can be rephrased.

Theorem 2. *The origin is stable if there is a continuously differentiable positive definite function $V(x)$ so that $\dot{V}(x)$ is negative semidefinite, and it is asymptotically stable if $\dot{V}(x)$ is negative definite.*

An important feature of Lyapunov's stability theorem is that its conditions are only sufficient. If a Lyapunov function candidate cannot satisfy the conditions for stability or asymptotic stability, it does not mean that the equilibrium is not stable or asymptotically stable. It only means that this particular Lyapunov function cannot establish such stability property.

When the origin $x = 0$ is asymptotically stable, it is often of interest to see how far from the origin the trajectory can be and still converge to the origin as t approaches infinity. This region can be defined as *region of attraction*.

Definition 2. *Let $\phi(t;x)$ be a solution of* (2.2) *that starts at initial state x at time $t = 0$. Then, the region of attraction is defined as the set of all points x such that $\phi(t;x)$ is defined for all $t \leqslant 0$ and $\lim_{t\to\infty} \phi(t,x) = 0$.*

If, for any initial state x, the trajectory $\phi(t;x)$ approaches the origin as $t \to \infty$, no matter how large $\|x\|$ is, then the asymptotically stable equilibrium point at the origin is said to be *globally asymptotically stable*. In this case, the region of attraction is the whole space $\mathbb{R}$.

Theorem 3. *Let $x = 0$ be an equilibrium point for* (2.2). *Let $V : \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function such that*

$$V(0) = 0 \ \text{ and } \ V(x) > 0, \ \ \forall x \neq 0 \tag{2.10}$$

$$\|x\| \to \infty \Rightarrow V(x) \to \infty \tag{2.11}$$

$$\dot{V}(x) < 0, \ \ \forall x \neq 0 \tag{2.12}$$

then $x = 0$ is globally asymptotically stable.

A function satisfying the condition $\|x\| \to \infty \Rightarrow V(x) \to \infty$ is said to be *radially unbounded*.

2.2.3 The Invariance Principle

If in a domain about the origin there is a Lyapunov function whose derivative along the trajectories of the system is negative semidefinite, and if it can be established that no trajectory can stay identically at points where $\dot{V}(x) = 0$, except at the origin, then the origin is asymptotically stable. This idea follows from LaSalle's *invariance principle*.

Lemma 3. *If a solution $x(t)$ of* (2.2) *is bounded and belongs to D for $t \geqslant 0$, then its positive limit set L^+ is a nonempty, compact, invariant set. Moreover, $x(t)$ approaches L^+ as $t \to \infty$.*

Now LaSalle's theorem can be stated.

Theorem 4. *Let $\Omega \subset D$ be a compact set that is positively invariant with respect to* (2.2). *Let $V : D \to \mathbb{R}$ be a continuously differentiable function such that $\dot{V}(x) \leqslant 0$ in Ω. Let E be the set of all points in Ω where $\dot{V}(x) = 0$. Let M be the largest invariant set in E. Then every solution starting in Ω approaches M as $t \to \infty$.*

Unlike Lyapunov's theorem, LaSalle's theorem does not require the function $V(x)$ to be positive definite. Furthermore, the construction of the set Ω does not have to be tied in with the construction of the function $V(x)$.

2.2.4 Stability of Nonautonomous Systems

Let us consider the nonautonomous system (2.1), where $f : [0,\infty) \times D \to \mathbb{R}^n$ is piecewise continuous in t and locally Lipschitz in x on $[0,\infty) \times D$, and $D \subset \mathbb{R}^n$ is a domain that contains the origin $x = 0$. The origin is an equilibrium point for (2.1) at $t = 0$ if

$$f(t,0) = 0, \ \ \forall t \geqslant 0. \tag{2.13}$$

Definition 3. *The equilibrium point $x = 0$ of* (2.1) *is*

- *stable if, for each $\varepsilon > 0$, there is $\delta = \delta(\varepsilon, t_0) > 0$ such that*

$$\|x(t_0)\| < \delta \ \ \Rightarrow \|x(t)\| < \varepsilon, \ \ \forall t \geqslant t_0 \geqslant 0 \tag{2.14}$$

- *uniformly stable if, for each $\varepsilon > 0$, there is $\delta = \delta(\varepsilon) > 0$, independent of t_0, such that* (2.14) *is satisfied.*

- *unstable if it is not stable.*

- *asymptotically stable if it is stable and there is a positive constant $c = c(t_0)$ such that $x(t) \to 0$ as $t \to \infty$, for all $\|x(t_0)\| < c$.*

- *uniformly asymptotically stable if it is uniformly stable and there is a positive constant c, independent of t_0, such that for all $\|x(t_0)\| < c$, $x(t) \to 0$ as $t \to \infty$, uniformly in t_0, i.e. for each $\eta > 0$, there is $T = T(\eta) > 0$ such that*

$$\|x(t)\| < \eta, \ \ \forall t \leqslant t_0 + T(\eta), \ \ \forall \|x(t_0)\| < c \tag{2.15}$$

- *globally uniformly asymptotically stable if it is uniformly stable, $\delta(\varepsilon)$ can be chosen to satisfy $\lim_{\varepsilon \to \infty} \delta(\varepsilon) = \infty$, and, for each pair of positive numbers η and c, there is $T = T(\eta, c) > 0$ such that*

$$\|x(t)\| < \eta, \ \ \forall t \geqslant t_0 + T(\eta, c), \ \ \forall \|x(t_0)\| < c. \tag{2.16}$$

For autonomous systems, the invariant set theorems are very useful when it comes to stability analysis, while the conclusions about asymptotic stability can be drawn even when $\dot{V}(x)$ is only negative semidefinite. On the other hand, the invariant set theorems are not applicable to nonautonomous systems. This makes the stability analysis of nonautonomous systems more difficult in general. An important results which helps in this case is the so-called *Barbalat's lemma*. It is a purely mathematical result concerning the asymptotic properties of functions and their derivatives, but when properly used for dynamic systems, especially for nonautonomous systems, it solves many problems. Now we can state Barbalat's lemma.

Lemma 4. *If the differentiable function $f(t)$ has a finite limit as $t \to \infty$, and if $\dot{f}$ is uniformly continuous, then $\dot{f}(t) \to 0$ as $t \to \infty$.*

Definition 4. *A function f is said to be uniformly continuous on $[0,\infty)$ if*

$$\forall \Lambda > 0,\ \ \exists \eta(\Lambda) > 0,\ \ \forall t_1 \geqslant 0,\ \ \forall t \geqslant 0,\ \ |t - t_1| < \eta \Rightarrow |f(t) - f(t_1)| < \Lambda. \qquad (2.17)$$

In other words, f is uniformly continuous if one can always find an η which does not depend on the specific point t_1, and in particular, such that η does not shrink as $t_1 \to \infty$. A convenient approach to analyse continuity is to examine the function's derivative. A sufficient condition for a differentiable function to be uniformly continuous is that its derivative is bounded. Now a practical corollary of Barbalat's lemma can be stated.

Corollary 1. *If the differentiable function $f(t)$ has a finite limit as $t \to \infty$, and is such that $\ddot{f}$ exists and is bounded, then $\dot{f}(t) \to 0$ as $t \to \infty$.*

When analysing dynamic systems, especially when designing a controller for mobile robots, the next *Lyapunov-like* corollary is used.

Lemma 5. *If a scalar function $V(x,t)$ satisfies the following conditions*

- *$V(x,t)$ is lower bounded,*
- *$\dot{V}(x,t)$ is negative semidefinite,*
- *$\dot{V}(x,t)$ is uniformly continuous in time,*

then $\dot{V}(x,t) \to 0$ as $t \to \infty$.

Using this kind of a *Lyapunov-like* analysis differs from the classical Lyapunov analysis. The function V can simply be a lower bounded function of x and t, instead of a positive definite function, and the derivative $\dot{V}$ must be uniformly continuous, in addition to being negative or zero. This is usually being done by proving that $\ddot{V}$ is bounded. In such analysis, the primary difficulty is to properly choose the scalar function V.

2.3 Backstepping

In order to understand the control structure introduced in Section 4.6, the main idea of the backstepping technique needs to be clarified. It is a technique for designing stabilizing control laws for a special class of nonlinear dynamic systems in a recursive way. As described in (Khalil and Grizzle, 1996), let us consider the system

$$\begin{aligned} \dot{\eta} &= f(\eta) + g(\eta)\xi, \\ \dot{\xi} &= u, \end{aligned} \tag{2.18}$$

where $[\eta^T, \xi]^T \in \mathbb{R}^{n+1}$ is the state and $u \in \mathbb{R}$ is the control input. The functions $f : D \to \mathbb{R}^n$ and $g : D \to \mathbb{R}^n$ are smooth in a domain $D \subset \mathbb{R}^n$ that contains $\eta = 0$ and $f(0) = 0$. The goal is to design a state feedback control law to stabilize the origin $(\eta = 0, \xi = 0)$. We assume that both f and g are known. Let us suppose that $\dot{\eta} = f(\eta) + g(\eta)\xi$ can be stabilized by a smooth state feedback control law $\xi = \phi(\eta)$, where $\phi(0) = 0$, which means that the origin of

$$\dot{\eta} = f(\eta) + g(\eta)\phi(\eta) \tag{2.19}$$

is asymptotically stable. Furthermore, let us suppose that there is a smooth and positive definite Lyapunov function $V(\eta)$ that satisfies

$$\frac{\partial V}{\partial \eta}\big[f(\eta) + g(\eta)\phi(\eta)\big] \leqslant -W(\eta), \quad \forall \eta \in D, \tag{2.20}$$

where $W(\eta)$ is positive definite. Now, the following expression holds

$$\begin{aligned} \dot{\eta} &= \big[f(\eta) + g(\eta)\phi(\eta)\big] + g(\eta)\big[\xi - \phi(\eta)\big] \\ \dot{\xi} &= u. \end{aligned} \tag{2.21}$$

With a change of variables

$$z = \xi - \phi(\eta) \tag{2.22}$$

we get a system

$$\begin{aligned} \dot{\eta} &= \big[f(\eta) + g(\eta)\phi(\eta)\big] + g(\eta)z, \\ \dot{z} &= u - \dot{\phi}. \end{aligned} \tag{2.23}$$

Since f, g, and ϕ are known, the derivative $\dot{\phi}$ can be computed using the expression

$$\dot{\phi} = \frac{\partial \phi}{\partial \eta}\big[f(\eta) + g(\eta)\xi\big]. \tag{2.24}$$

By defining $v = u - \dot{\phi}$ the system is reduced to

$$\begin{aligned} \dot{\eta} &= \big[f(\eta) + g(\eta)\phi(\eta)\big] + g(\eta)z, \\ \dot{z} &= v, \end{aligned} \tag{2.25}$$

which is similar to the initial system, except that the first component has an asymptotically stable origin when the input is zero.

Using the Lyapunov function candidate

$$V_c(\eta, \xi) = V(\eta) + \frac{1}{2}z^2, \tag{2.26}$$

the first time derivative is

$$\begin{aligned} \dot{V}_c &= \frac{\partial V}{\partial \eta}\big[f(\eta) + g(\eta)\phi(\eta)\big] + \frac{\partial V}{\partial \eta}g(\eta)z + zv \\ &\leqslant -W(\eta) + \frac{\partial V}{\partial \eta}g(\eta)z + zv. \end{aligned} \tag{2.27}$$

By choosing

$$v = -\frac{\partial V}{\partial \eta}g(\eta) - kz, \;\; k > 0, \tag{2.28}$$

we get

$$\dot{V}_c \leqslant -W(\eta) - kz^2, \tag{2.29}$$

which shows that the origin $(\eta = 0, z = 0)$ is asymptotically stable. Since $\phi(0) = 0$, then the origin $(\eta = 0, \xi = 0)$ is asymptotically stable as well. By employing all the substitutes, we get the feedback control law

$$u = \frac{\partial \phi}{\partial \eta}\big[f(\eta) + g(\eta)\xi\big] - \frac{\partial V}{\partial \eta}g(\eta) - k\big[\xi - \phi(\eta)\big]. \tag{2.30}$$

The idea of the procedure can be summarized in the next lemma.

Lemma 6. *Let us consider the system* (2.18)*, with $\phi(\eta)$ as a stabilizing state feedback control law with the property $\phi(0) = 0$. Let $V(\eta)$ be a Lyapunov function that satisfies* (2.20)*, with some positive definite function $W(\eta)$. Then, the state feedback control law* (2.30) *stabilizes the origin of* (2.18) *with $V(\eta) + \big[\xi - \phi(\eta)\big]^2/2$ as a Lyapunov function. Furthermore, if all the assumptions hold globally and $V(\eta)$ is radially unbounded, the origin will be globally asymptotically stable.*

The backstepping technique is very important when it comes to incorporating the mobile robot's kinematic model into the dynamic model. The principle of the above introduced equations is visualized in Figure 2.4.

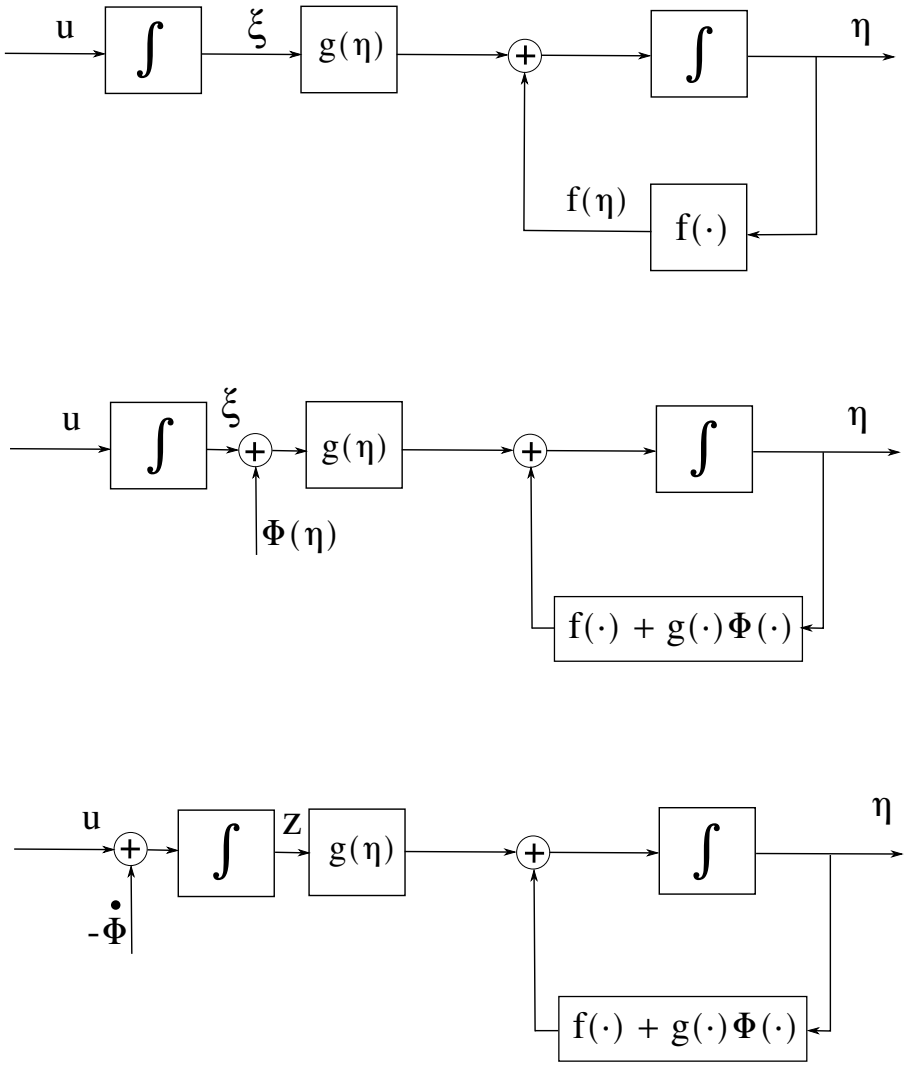


Figure 2.4: *The block diagram of system* (2.18) *in the original form, after introducing* $\phi(\eta)$ *and after backstepping* $-\phi(\eta)$ *through the integrator.*

Chapter 3

Generic Path Following Control of Wheeled Mobile Robots

3.1 Introduction

Most of the path following solutions existing in the literature are developed for a specific mobile robot configuration. A comprehensive classification of different mobile robot configurations was done by Campion et al. in (Campion *et al.*, 1996), detailing all possibilities of structural properties of the wheels.

Path following for omnidirectional drives is tackled by Kanjanawanishkul and Zell in (Kanjanawanishkul and Zell, 2009), where the authors employ model predictive control (MPC) to design the control law, by using a virtual vehicle formulation. Given the error derived from the state vectors of the robot and the path, the MPC control law explicitly deals with the virtual vehicle dynamics. Pioneer work on the virtual vehicle concept is presented in (Kanayama *et al.*, 1990), where a stable, Lyapunov-based path following control law is proposed for nonholonomic vehicles, i.e. for all wheeled vehicles which are not omnidirectional.

A thorough study on feedback control of Ackermann (car-like) vehicles, including path following, is made by De Luca et al. in (De Luca *et al.*, 1998). In (Coulter, 1992) and (Thrun *et al.*, 2006) two geometric path following control algorithms are proposed, which in principle do not rely on any kinematic model. However, both of the algorithms assume the steering characteristic to be Ackermann vehicles. Path following of car-like vehicles in the presence of sliding is tackled in (Lenain *et al.*, 2006), using a special kinematic model accounting for sliding, and a model predictive control law. In (Nizard *et al.*, 2016) a path following controller for bi-steerable vehicles is presented. The reference path is transformed to two synchronized paths, and the approach is evaluated on an electrical public transport vehicle EZ10.

Unicycle motion control is a popular example in control theory applications, which is the reason for many research results in this area. Some examples can be found in (Samson, 1995), (Lapierre *et al.*, 2006), (Morro *et al.*, 2011), and (Ghommam *et al.*, 2010). Path following solution for skid-steered vehicles is proposed in (Lucet *et al.*, 2008) using sliding mode control. More work on path following of skid-steered vehicles

can be found in (Huskić *et al.*, 2017d), (Pentzer *et al.*, 2014b) and (Rajagopalan *et al.*, 2016).

Even though all the above mentioned algorithms are interesting and efficient solutions, they are model-dependent, and designed for a certain type or a group of mobile robots. There are, however, some algorithms which are model-independent and in that sense generic. They use pure geometric tracking and can be utilized by any kinematic configuration. One such algorithm can be found in (Oftadeh *et al.*, 2014). The authors introduce a concept of a generalized wheel, and simplify the kinematic and nonholonomic constraints. Because of this advantage, it is possible to derive a time-optimal solution for the speed of the robot, such that the actuator velocities stay within specified bounds. In (Egerstedt *et al.*, 2001) two model-independent path following solutions are proposed. Both of the solutions are based on the virtual vehicle principle. This approach was extended by Maček et al. in (Maček *et al.*, 2005) with an additional velocity control.

Mojaev and Zell have developed a generic path following algorithm in (Mojaev and Zell, 2004), and evaluated it on a differential drive. This approach was utilized for omnidirectional drives by Li et al. in (Li *et al.*, 2007) for the development of a ball dribbling robot. In this chapter, we are proposing an approach to provide safer, more accurate and stable path following performance, outperforming the algorithm proposed in (Mojaev and Zell, 2004), and offering more features in the linear velocity control than the algorithm proposed in (Maček *et al.*, 2005). The contributions of this chapter are as follows:

- proposing a novel heuristic linear velocity control,
- introducing this linear velocity control to the approach proposed in (Mojaev and Zell, 2004),
- utilizing the approach for Ackermann and bi-steerable drives,
- experimentally evaluating the algorithm performance on
  - an omnidirectional drive (in a robotic competition and in laboratory conditions),
  - Ackermann and bi-steerable drives (laboratory conditions).

3.2 Orthogonal Projection Based Control

To understand the principle of the presented approach, Figure 3.1 should be observed. The world frame is defined with the axes X_w, Y_w and the origin O. The robot's center of mass can be seen as point R. The geometric reference path is denoted with P, and the robot's orthogonal projection to the path is point T. The moving reference frame $\{P\}$ is defined with its origin at point T. Abscissa X_t and ordinate X_n are unit tangent and unit normal vectors at point T, respectively. Along the axis X_n, the orthogonal distance from

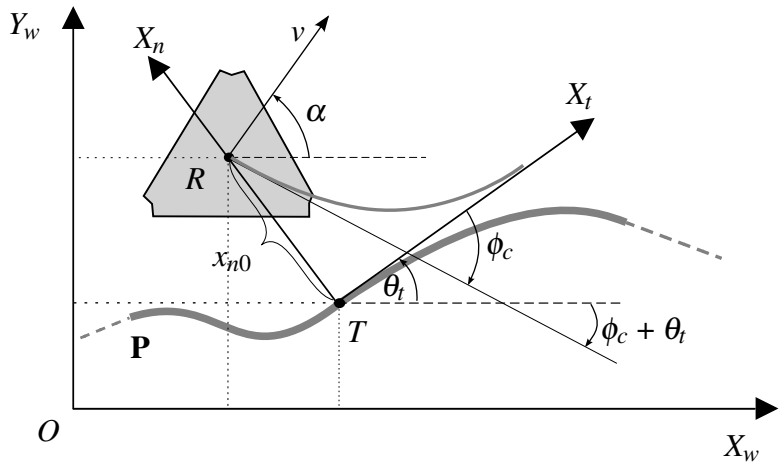


Figure 3.1: *Orthogonal projection based path following*

the robot to the path is denoted with x_n, and along the axis X_t, the tangential distance with x_t. The tangential angle of the path at point T is denoted with θ_t. This is, in fact, the angle between the moving reference frame and the world frame. The robot's translational velocity is represented with v, and the robot's moving direction angle is α. If we denote x_{n0} as the distance from the robot to the path at point T, it is possible to define the following exponential law

$$x_n = x_{n_0} e^{-kx_t}, \tag{3.1}$$

where k is a positive constant which regulates the speed of convergence. The orthogonal distance x_n is now an exponential function of the tangential distance, with an initial value x_{n_0} at point T. If we denote the angle of the exponential function's tangent as ϕ_c, the following applies: $\tan\phi_c = \frac{dx_n}{dx_t}$. Now, this angle can be expressed as

$$\phi_c = \arctan(-kx_n). \tag{3.2}$$

If we denote the desired velocity with v_n, the robot's model in the moving reference frame is then

$$\begin{aligned} \dot{x}_t &= v_n \cos\phi_c, \\ \dot{x}_n &= v_n \sin\phi_c. \end{aligned} \tag{3.3}$$

The model in Eq. (3.3) can be used to control any robot, regardless of the kinematics, since the robot is modelled just as a geometric point. In order for the robot to converge to the reference path, the angle ϕ_c needs to converge to zero. In (Mojaev and Zell, 2004), a differential drive was controlled using PD control with ϕ_c as error input. In their work, control rotational velocity is expressed as $\omega = K_p\phi_c + K_d\frac{d\phi_c}{dt}$, where K_p and K_d are proportional and derivative gains, respectively.

Omnidirectional drives are able to move in any direction, regardless of their orientation. This means that they can independently move and simultaneously change their orientation, i.e. rotate. Because of this property, it was possible for Li et al. in (Li *et al.*, 2007) to directly use ϕ_c as direction angle, and use separate PD control for rotation. In this case, using ϕ_c as direction angle, the robot will always converge to the path by following the exponential function in the local coordinate frame. The robot's control model in the world frame is then

$$\begin{aligned} v_x &= v_n \cos\alpha, \\ v_y &= v_n \sin\alpha, \end{aligned} \tag{3.4}$$

where $\alpha = \phi_c + \theta_t$, and v_x, v_y are the robot's translational velocity components. Rotation control is then $\omega = k_1(e_\theta + k_2\dot{e}_\theta)$, where k_1, k_2 are proportional and derivative gains, respectively, and $e_\theta = \theta_d - \theta$ the error between the desired orientation θ_d and the measured orientation θ. In the case of nonholonomic robots, θ corresponds with α, and there is no independent rotation. A stability analysis of this algorithm can be found in (Li *et al.*, 2007).

By using this formulation, the inherent singularity, which appears when $x_n = \frac{1}{c_{max}}$, can be avoided, where c_{max} is the maximum curvature of the path. This problem was analysed by Samson in (Samson, 1992).

3.3 Longitudinal Control

The main limitation of the described algorithm is its lack of linear velocity control, since the desired velocity v_n is a positive constant. At higher velocities, the performance would greatly decrease in the following ways:

- at highly curved parts of the path, the tracking accuracy decreases,
- in cluttered and/or dynamic environments, danger of colliding with obstacles increases,
- abruptly going from full velocity to zero at the goal position increases the final positioning error, the danger of tipping over, and may damage the actuators,
- driving fast and rotating at the same time introduces unwanted dynamic effects (applies only to omnidirectional drives).

These problems can be solved with a heuristic approach described in the following text.

3.3.1 Path Curvature

The following idea is inspired by the one proposed in (Maček *et al.*, 2005). First, we define path curvature.

If a 2D curve P is parametrized by the arc length s, then $x_d = p(s)$, and $y_d = q(s)$ are the x and y coordinates of the desired path, respectively. If we denote $p' = \frac{\partial p}{\partial s}$, $q' = \frac{\partial q}{\partial s}$, and $p'' = \frac{\partial^2 p}{\partial s^2}$, $q'' = \frac{\partial^2 q}{\partial s^2}$, then the curvature κ can be computed as

$$\kappa = \frac{p'q'' - p''q'}{(p'^2 + q'^2)^{\frac{2}{3}}}, \tag{3.5}$$

which is further detailed, e.g., by Toponogov in (Toponogov, 2006).

Now, we define a free control parameter which we name *path-look-ahead distance*, an arbitrarily chosen distance in path coordinates. It starts at the orthogonal projection T, and ends between T and the goal position, at most exactly at the goal position. So, if we denote the path-look-ahead distance with l_κ, then $l_\kappa \in [0, s_G - s_T]$. Here, s_T and s_G are the point T and the goal position in path coordinates, respectively. The proposed penalty factor using the curvature measure for a discretized path is then

$$\xi_\kappa = e^{-K_\kappa \sum_{i=i_T}^{i_{l_\kappa}} |\kappa_i|}, \tag{3.6}$$

where $K_\kappa > 0$ is a free parameter which determines the speed of convergence, i_T is the index of orthogonal projection T, i_{l_κ} is the index of the point positioned at the path-look-ahead distance from the projected point, and κ_i is the curvature in the current point with the index i. By using this penalty factor to scale the linear velocity as $v = v_n \xi_\kappa$, tracking accuracy remains high even in the areas of high path curvature.

3.3.2 Distance to Obstacles

To be sure that the robot will avoid collisions with static and dynamic obstacles in its environment, a safety measure should be considered, similar to the proposed curvature penalty factor. This penalty factor is proposed as

$$\xi_o = e^{-\frac{K_o}{d_o}}, \tag{3.7}$$

where d_o is the distance to the obstacles of interest, and K_o is a convergence parameter. This distance to the obstacles of interest can be chosen in different ways. In this work, we use the distance to the nearest obstacle, since it provides satisfying results. Using this penalty factor as $v = v_n \xi_o$, danger of colliding with obstacles is greatly reduced, and the navigation has a much higher safety level.

3.3.3 Distance to the Goal Position

In order to avoid dangerous sudden braking at the end of the path, it is necessary to introduce asymptotic deceleration, as the robot approaches the goal. This penalty factor

can be defined as

$$\xi_g = e^{-\frac{K_g}{d_g}}, \tag{3.8}$$

where $K_g > 0$ is a parameter which determines the convergence of the asymptote, and d_g is the distance to the goal in path coordinates. By using this penalty factor as $v = v_n \xi_g$, there is no danger of tipping over, tracking accuracy remains preserved at the goal position, and the actuators are not stressed with extreme values.

3.3.4 Angular Velocity

An omnidirectional robot's linear and angular velocity are kinematically independent, but at higher velocities, different dynamic effects occur. If these effects are not explicitly accounted for, they might lead to unwanted behaviour.

We then propose the following rotation penalty factor

$$\xi_\omega = e^{-K_\omega |\omega|}, \tag{3.9}$$

where $K_\omega > 0$ regulates the convergence, and the linear velocity is then $v = v_n \xi_\omega$. If the robot needs to change its orientation during the drive, it will shortly decelerate to keep the tracking accuracy, until the rotation is complete. This way, the tracking accuracy is preserved, and there are no unwanted dynamic effects.

3.3.5 Final Heuristic Longitudinal Control

By using all of the proposed penalty factors for the linear velocity, the final expression is then given as

$$v = v_n exp\left(-\left(K_\kappa \sum_{i=i_P}^{i_\kappa} |\kappa_i| + K_\omega |\omega| + \frac{K_o}{d_o} + \frac{K_g}{d_g}\right)\right). \tag{3.10}$$

This expression gives a compact and efficient solution for linear velocity control, which allows the robot to cope with the problems of complex paths in cluttered and dynamic environments.

3.4 Orthogonal-Exponential Approach

Now we introduce the proposed longitudinal control solution to the path following control presented in Section 3.2. This integral control algorithm we address as *Orthogonal-exponential approach*. The robot's model in the moving reference frame is then

$$\begin{aligned} \dot{x}_t &= v \cos \phi_c, \\ \dot{x}_n &= v \sin \phi_c, \end{aligned} \tag{3.11}$$

where v is defined as in Eq. (3.10), and $\phi_c = \arctan(-kx_n)$, as in Eq. (3.2).

Omnidirectional robot can now be directly controlled, similar as in (Li *et al.*, 2007). We use the error angle ϕ_c in robot coordinates, such that ${}^R\phi_c = \phi_c + \theta_t - \theta$.

For the Ackermann and bi-steerable drive we propose a PID control with ${}^R\phi_c$ as error input: $\varphi = K_P {}^R\phi_c + K_I \int_0^T {}^R\phi_c dt + K_D \frac{d^R\phi_c}{dt}$, where K_P, K_I, K_D are proportional, integral and derivative gains, respectively. The bi-steerable drive is assumed to be symmetrical: $\varphi_r = -\varphi_f$, where φ_f and φ_r are front and rear axle steering angles, respectively. The resulting steering is then: $\varphi = \varphi_f - \varphi_r$. With $\{K_P = 1, K_I = 0, K_D = 0\}$, i.e. using a P controller, the steering angle control becomes $\varphi = {}^R\phi_c$. This choice of parameters showed satisfying results and was used in the experiments.

To analyse the stability of the whole system, we follow the work of (Li *et al.*, 2007) and choose the Lyapunov candidate function as

$$V = \frac{1}{2} {x_n}^2 + \frac{1}{2}\phi_c^2, \tag{3.12}$$

and then the derivative can be found as

$$\begin{aligned} \dot{V} &= x_n \dot{x}_n + \phi_c \dot{\phi}_c \\ &= x_n v \sin(\arctan(-kx_n)) + \arctan(-kx_n) \frac{v \sin(\arctan(-kx_n))}{1 + (kx_n)^2}. \end{aligned} \tag{3.13}$$

Using the properties of the functions arctan and sin, we conclude that $\dot{V} < 0$ and that the equilibrium $(x_n = 0, \phi_c = 0)$ is globally asymptotically stable.

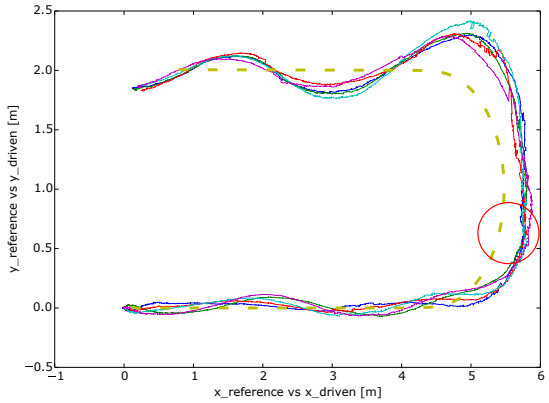
The robot's linear velocity v, computed as in Eq. (3.10), is positive and doesn't vanish with time. The velocity stays in the interval $v \in (0, v_n)$, regardless of the control parameter values. This implies that the global asymptotic stability is preserved by using the longitudinal control proposed in Section 3.3.

3.5 Experimental Results

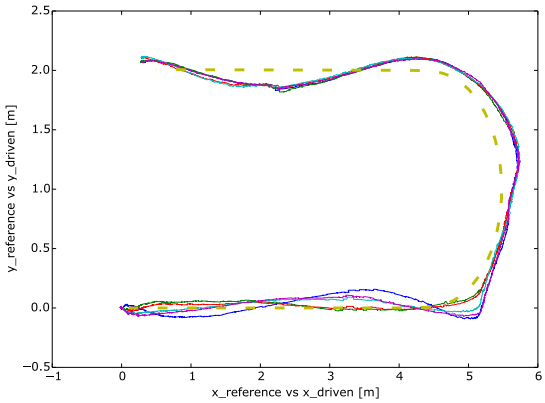
The proposed orthogonal-exponential approach was experimentally evaluated in two different manners:

- with an omnidirectional drive, we participated as Team Attempto Tübingen in the international robotics competition SICK Robot Day 2014 and won the 2nd place,
- in our laboratory, path following experiments were conducted using an omnidirectional drive, Ackermann and bi-steerable drive.

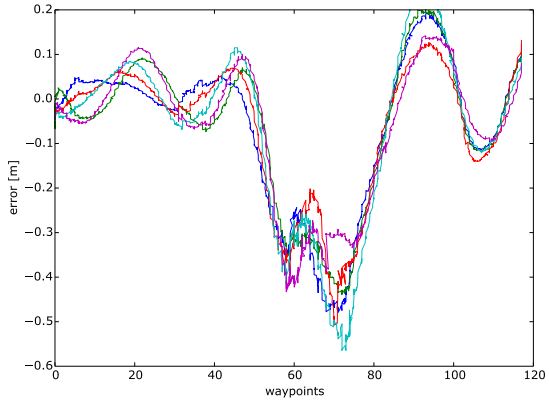
The robots we used can be seen in Figure 3.3. The omnidirectional robot Arnie is based on hardware used at RoboCup competitions, with details described in (Li, 2009). The computer configuration consists of an Intel Core2Duo P8700 dual-core processor with



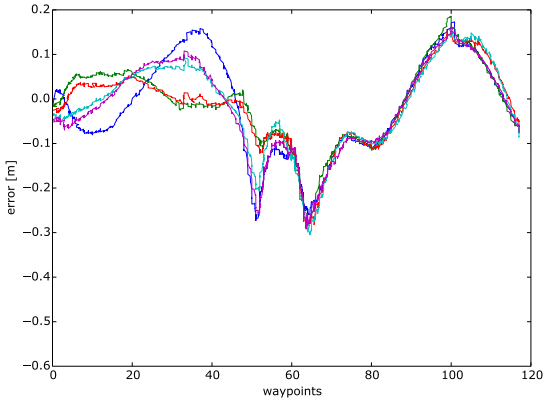
(a) *Path following using the old approach*



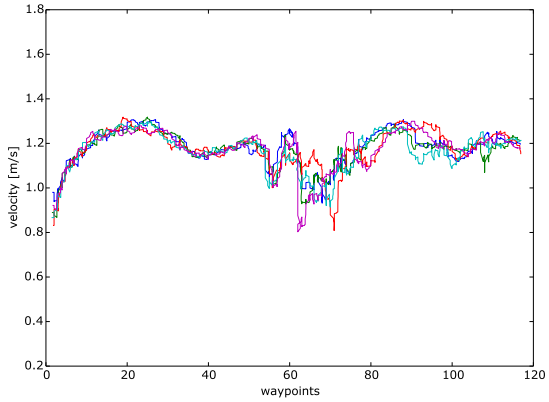
(b) *Path following using the proposed approach*



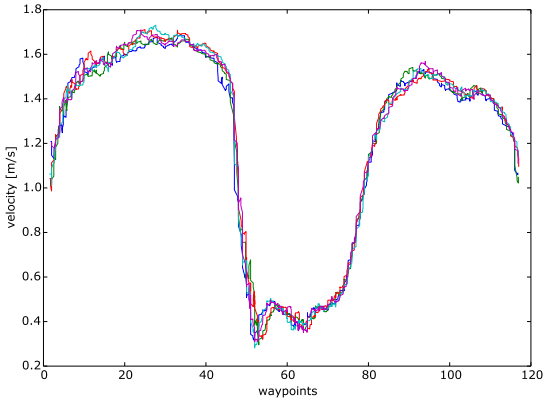
(c) *Following error of the old approach*



(d) *Following error of the proposed approach*



(e) *Actual velocity using the old approach*

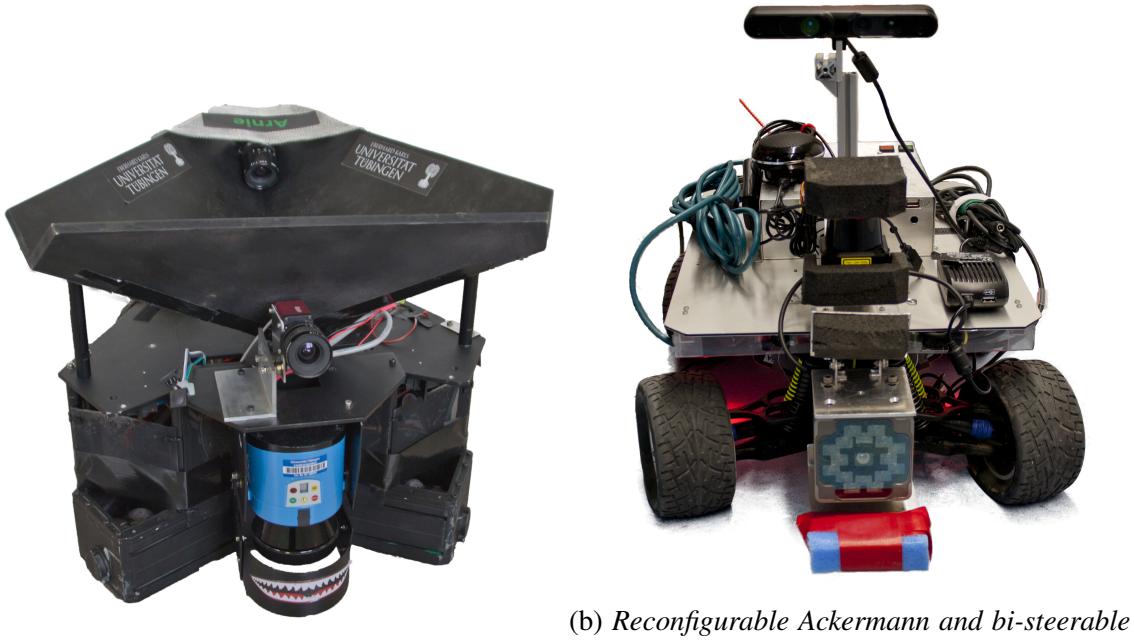


(f) *Actual velocity using the proposed approach*

Figure 3.2: *Five runs (different colors) of following a U-curve with an omnidirectional robot, where* $v_n = 1.3\mathrm{m\,s^{-1}}$. *The golden dashed line in (a) and (b) represents the desired path.*

2.53GHz clock speed and 2GB of RAM. The pyramid-shaped construction on top of the robot is designed for collecting the objects in SICK Robot Day 2014. Inside this construction there is a PointGrey Firefly camera, used for reading the bar codes from the objects. In front of the robot there is a SICK LMS100 laser scanner, and in the back, a SICK TiM551, both used for obstacle detection and mapping. Furthermore, there is an Allied Vision Marlin camera for sign detection.

The reconfigurable Ackermann and bi-steerable robot Gloin is equipped with a i7-4785T Intel CPU with 4 real and 8 virtual cores, 2.20 GHz each, and 8GB of RAM. The sensors used for the experiments consist of a Hokuyo UTM-30LX laser scanner, and internal wheel encoders.



(a) *Omnidirectional robot Arnie*

(b) *Reconfigurable Ackermann and bi-steerable robot Gloin*

Figure 3.3: *Robots used for the experiments presented in this chapter.*

3.5.1 SICK Robot Day 2014

The SICK Robot Day is a bi-annually hosted robotic competition by the well known sensor producer SICK AG, in Waldkirch, Germany. In the SICK Robot Day 2014, where 14 international teams participated, the competition scenario resembled the one of an automated warehouse. There were four robots competing at the same time to collect and deliver special objects in a limited time window. The robots were fully autonomous, and had to detect filling and delivery stations, as well as navigate in the arena, while avoiding collisions with other robots. The goal was to collect and deliver the most objects in

the specified time window, while correctly delivered objects were awarded with positive points, and wrong deliveries would lead to penalty points.

Having such a dynamic environment and a strict time limit required the motion of the robot to be suitable for cluttered environments, avoiding all the collisions, and driving fast when the area is clear. Furthermore, in order to detect the special stations, the robot would need to drive through the arena and look around for the special number signs. In order to do so while driving, the natural solution was to use an omnidirectional drive and to rotate while driving. These requirements were met with the Orthogonal-exponential approach. This approach allows independent translation and rotation of an omnidirectional drive, and the linear velocity control in Eq. (3.10) deals with dynamic and cluttered environments, while the speed increases, if there is no danger ahead.

A typical path following scenario in the training arena can be observed in Figure 3.4, visualized in ROS/RViz. The black and white number signs stand above the special delivery stations. After picking up a special object from one of the fetching stations in the middle of the arena (visualized as green cylinders), the goal is to read a bar code from the fetched object, to find the corresponding number sign, and to deliver the object to the station located underneath the correct number sign. In this example, the bar code was corresponding to the number sign 2, so the robot delivered the object to the proper station located under this sign. Thick blue line represents the driven path, and thin red line represents the planned reference path. Blue and red dots represent the walls detected by the laser scanner, and the blue square in the middle represents the central point for the fetching stations.

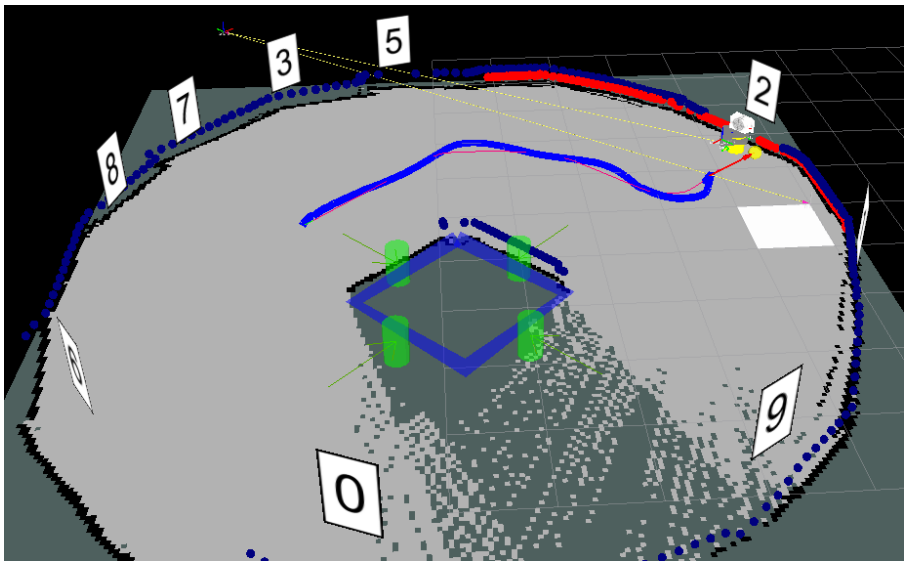


Figure 3.4: *A representative example of path following inside the training arena.*

Our team won the 2nd place, and was able to quickly solve the tasks without any collision. More details about the competition and the complete system can be found in

(Buck *et al.*, 2015).

3.5.2 Laboratory Experiments

In this section, results from the laboratory experiments with a U-shaped curve will be explained in detail. All the results in the tables were averaged over 5 experiments. The errors and the velocities are plotted relative to the path. In the area of the highest curvature, there was a planted obstacle near the path, such that the robot would hit it, if the deviation from the path was too large. This area is illustrated with a red circle in the following figures. The obstacle was hit every time while driving with the old approach, and avoided every time using the proposed approach. This emphasizes the advantage of the inherent safety measures of the proposed approach. A typical path following scenario in laboratory conditions, visualized in ROS/RViz, can be observed in Figure 3.5. The thick blue line represents the actual path driven by the robot, and the thin green line the planned reference. The rectangles along the driven path represent the safety area around the robot in each pose. The red dot is the point to which the robot had to adjust its orientation, and it is placed on a rectangular obstacle. While driving, the omnidirectional robot rotates independently, in order to ”look” at this point. We refer to this point as *look-at point*. This way the independent translational and rotational control were tested.

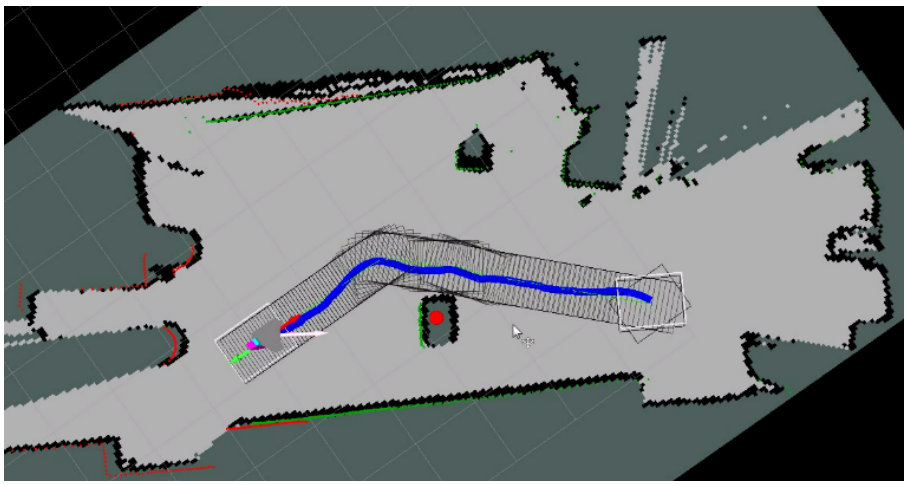


Figure 3.5: *A representative example of path following inside the laboratory.*

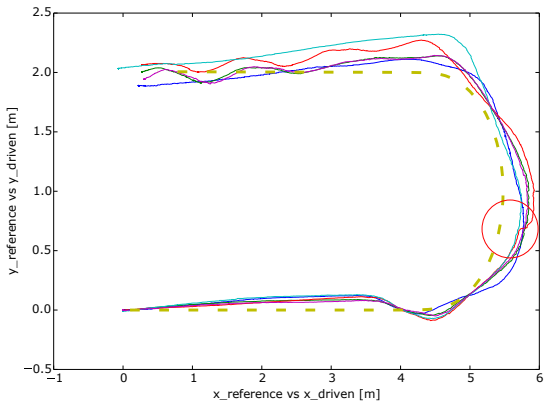
Omnidirectional robot Similar as in Figure 3.5, a look-at point was set on the planted obstacle, such that the robot had to change its orientation accordingly during driving. The desired velocity was chosen as the maximum at which the old approach could still follow the path ($v_n = 1.3\mathrm{m\,s^{-1}}$). A comparison between the old and the proposed approach can be seen in Figure 3.2. Different colors of the paths represent individual experiments repeated 5 times. As it can be observed in Figure 3.2a and Figure 3.2b, the driven path

Table 3.1: *Proposed vs. original algorithm: omnidirectional drive*

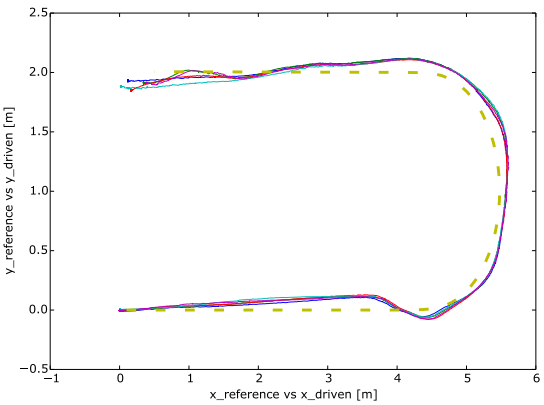
omni	e_{max}[cm]	e_{avg}[cm]	σ[cm]	$v_{max}[\frac{\mathrm{m}}{\mathrm{s}}]$	$v_{avg}[\frac{\mathrm{m}}{\mathrm{s}}]$
original	48.44	−7.7	17.14	1.32	1.16
proposed	28.73	−6.23	10.71	1.73	0.93
Δ[%]	40.7	19.06	37.51	23.69	20.12

deviates more from the reference (golden dashed line), when driving with the old approach, and the motion is oscillating more, which is especially visible in the upper part of Figure 3.2a. In Figure 3.2c and Figure 3.2d the tracking errors of the both approaches are presented. In Figure 3.2c, in the area between the waypoints ≈ 55 and 80, a large error can be observed, which corresponds to the area of the highest curvature in Figure 3.2a and Figure 3.2b, where the obstacle was planted. At this location the obstacle was hit in every experiment, using the old approach, and there was no collision at all by using the proposed approach. In Figure 3.2e and Figure 3.2f the speed profiles of both approaches are shown. By using the old approach, the speed is approximately constant. The desired speed was $1.3\mathrm{m\,s^{-1}}$, but the low level control of the robot could keep the average speed at $1.16\mathrm{m\,s^{-1}}$. By using the proposed approach, the robot would accelerate along the straight segments, and reach the speed of $1.73\mathrm{m\,s^{-1}}$, while the average speed is then $0.93\mathrm{m\,s^{-1}}$. Table 3.1 shows the improvement the proposed approach offers. The maximum tracking error with the old approach is 48.44cm, and with the proposed approach 28.73cm. This means that the tracking error with the proposed approach is 40.7% smaller than in the case of the old approach. The average error with the proposed approach is then 19.06% smaller than with the old approach. The proposed approach is also more reliable with a 37.51% decrease in standard deviation. The velocities are comparable.

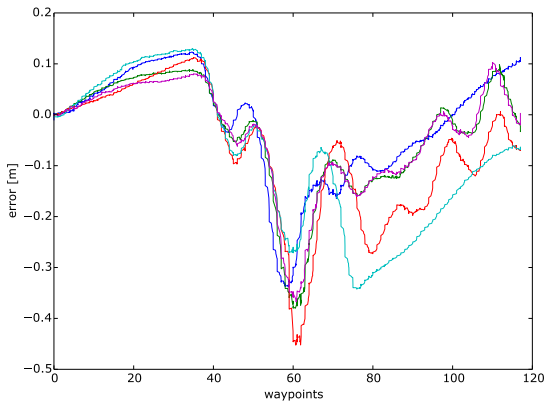
Bi-steerable robot A comparison between the old and the proposed approach for the bi-steerable configuration, with the command velocity $v_n = 0.7\mathrm{m\,s^{-1}}$, can be seen in Figure 3.6. As it can be seen in Figure 3.6a and Figure 3.6b, using the proposed approach, the path following performance is more accurate and repeatable. In the upper part of Figure 3.6a oscillating motion can be observed when using the old approach. Furthermore, in the area denoted with the red circle, there is a collision in every experiment, since the deviation from the reference path is too large. In Figure 3.6c and Figure 3.6d the tracking errors of the both approaches are shown. In Figure 3.6c there is a peak at around the waypoint number 60, which corresponds to the large error in the curve, while hitting the obstacle with the old approach. Between the waypoints 60 and 120 the above mentioned oscillating motion can be observed. The speed profiles are shown in Figure 3.6e and Figure 3.6f. The old approach has an average speed of $0.42\mathrm{m\,s^{-1}}$, while



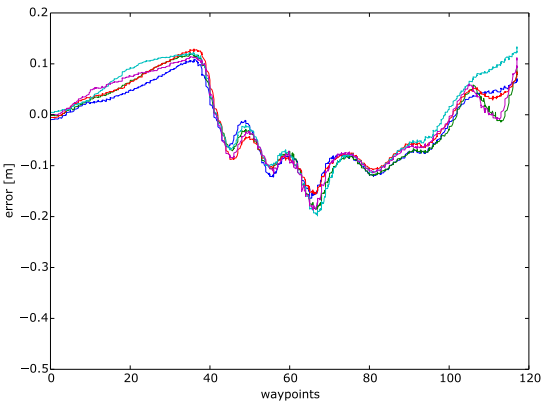
(a) *Path following using the old approach*



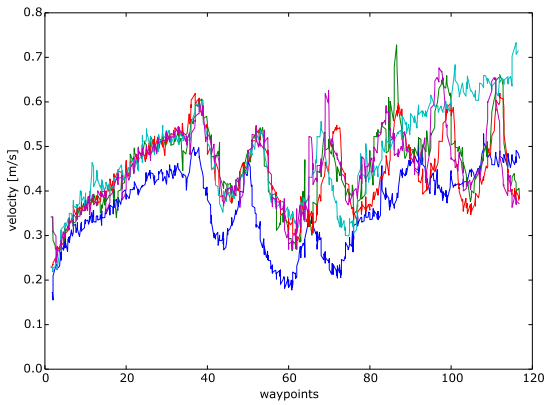
(b) *Path following using the proposed approach*



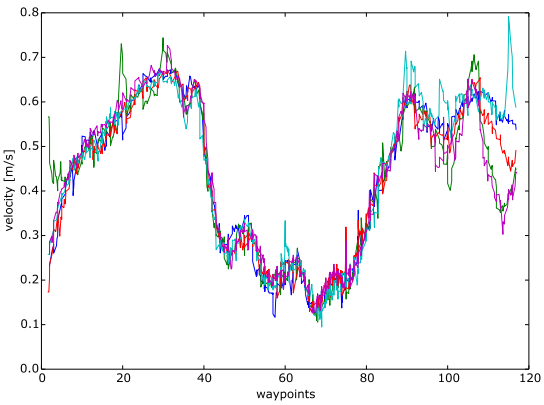
(c) *Following error of the old approach*



(d) *Following error of the proposed approach*



(e) *Actual velocity using the old approach*



(f) *Actual velocity using the proposed approach*

Figure 3.6: *Five runs of following a U-curve with a 2-axle-steering robot, where* $v_n = 0.7\,\mathrm{m\,s^{-1}}$

Table 3.2: *Proposed vs. original algorithm: bi-steerable drive*

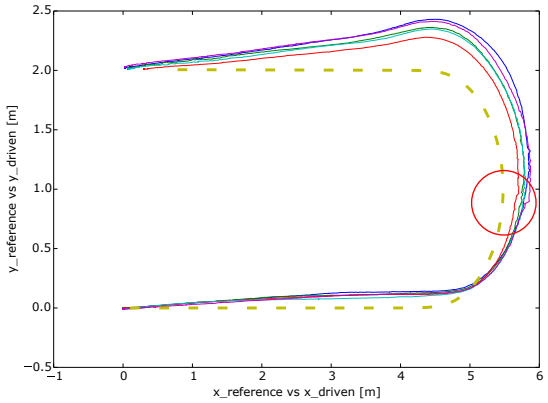
2-steer	e_{max}[cm]	e_{avg}[cm]	σ[cm]	$v_{max}[\frac{\text{m}}{\text{s}}]$	$v_{avg}[\frac{\text{m}}{\text{s}}]$
original	37.62	−5.2	12.34	0.66	0.42
proposed	17.83	−3.74	7.93	0.72	0.37
Δ[%]	52.6	28.13	35.74	8.61	13.68

Table 3.3: *Proposed vs. original algorithm: Ackermann-steering drive*

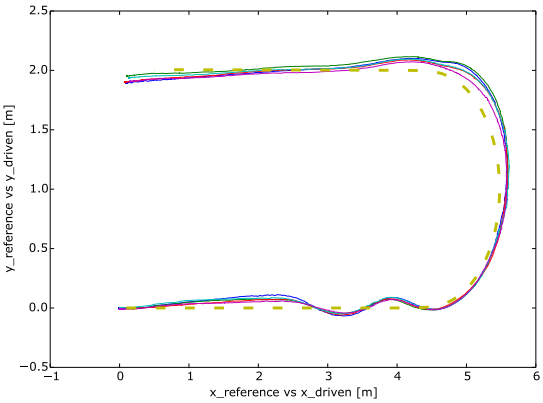
Ackermann	e_{max}[cm]	e_{avg}[cm]	σ[cm]	$v_{max}[\frac{\text{m}}{\text{s}}]$	$v_{avg}[\frac{\text{m}}{\text{s}}]$
original	38.28	−8.42	17.06	0.55	0.4
proposed	18.43	−3.618	7.57	0.8	0.41
Δ[%]	51.85	57.03	55.63	30.85	−2.08

the proposed approach reaches the speed of $0.72\text{m}\,\text{s}^{-1}$ in the straight segments, with an average speed of $0.37\text{m}\,\text{s}^{-1}$. Table 3.2 shows that the maximum tracking error with the old approach reaches 37.62cm, while with the proposed approach, the maximum error is 17.83cm. This means that the proposed approach has a 52.6% smaller maximum error. The average error with the proposed approach is 28.13% smaller than with the old approach, and the standard deviation decreases by 35.74%.

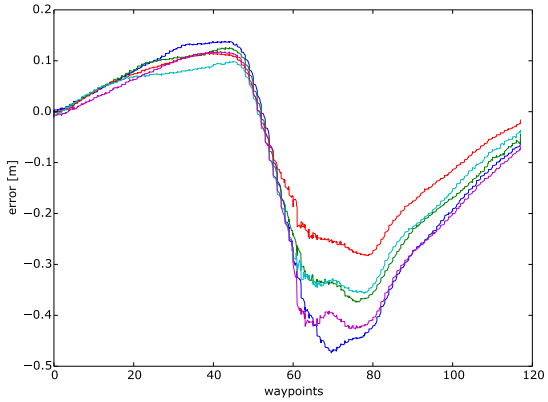
Ackermann-steering robot The results with the Ackermann-steering configuration can be seen in Figure 3.7. It is clear from Figure 3.7a and Figure 3.7b that the proposed approach offers more accurate motion. The motion in Figure 3.7a is not oscillating, as in the case of Figure 3.2a and Figure 3.6a, but a certain drift seems to be present, and the deviation from the path is large. This comes from the fact that the actuators are slow, and have a certain delay before they start moving. If driving with a constant speed and not decelerating before the curve, as in the case of the old approach, there will always be understeering present. The red circle represents the collision area, similar as in the previous experiments. In Figure 3.7c and Figure 3.7d the error profiles are shown. Similar as before, in Figure 3.7c, in the area between waypoints 60 and 80 there is a large error peak, when using the old approach, which corresponds with the collision area. As presented in Table 3.3, the proposed approach has a 51.85% smaller maximum error, a 57.03% smaller average error and a 55.63% smaller standard deviation. When driving with the old approach, the maximum speed is $0.55\text{m}\,\text{s}^{-1}$, and the average speed is $0.4\text{m}\,\text{s}^{-1}$. With the proposed approach, the maximum speed reaches $0.8\text{m}\,\text{s}^{-1}$, while the average speed is $0.41\text{m}\,\text{s}^{-1}$.



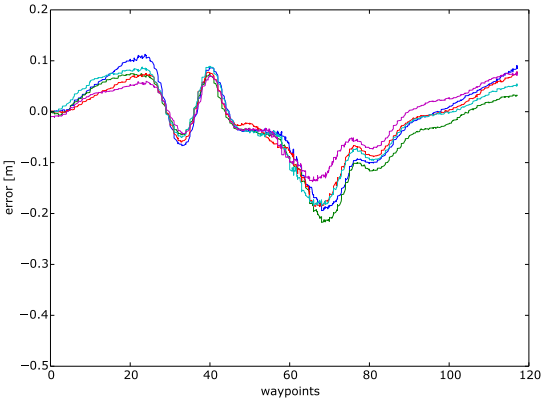
(a) *Path following using the old approach*



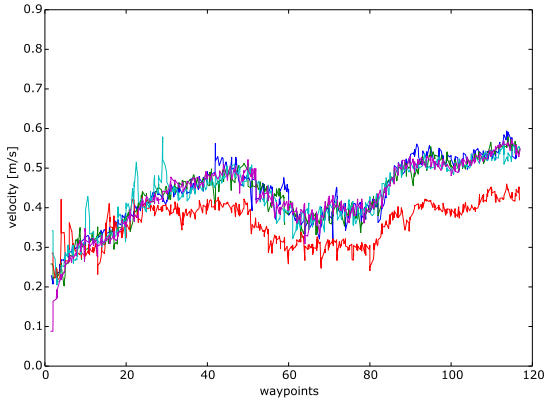
(b) *Path following using the proposed approach*



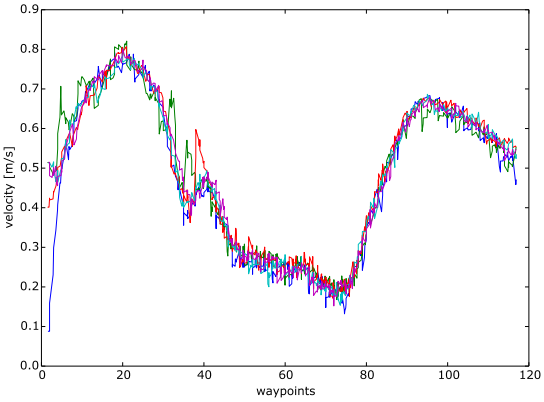
(c) *Following error of the old approach*



(d) *Following error of the proposed approach*



(e) *Actual velocity using the old approach*



(f) *Actual velocity using the proposed approach*

Figure 3.7: *Five runs of following a U-curve with an Ackermann-steering robot, where* $v_n = 0.6\,\mathrm{m\,s^{-1}}$

3.6 Conclusions

In this chapter, we propose a new heuristic linear velocity control for wheeled mobile robots for safe, stable and accurate path following in dynamic and cluttered environments. It is an alternative to complex model-based and predictive controllers which model the dynamics of the robot and the environment. The proposed longitudinal control can be used in addition to any path following controller which assumes constant linear velocity. It is coupled with the approach proposed in (Mojaev and Zell, 2004), and then utilized for Ackermann and bi-steerable drives. Its performance is experimentally evaluated using an omnidirectional robot and a reconfigurable Ackermann and bi-steerable robot. The proposed solution can be used on any wheeled mobile robot, regardless of the kinematic configuration. The performance was also evaluated in the international robotics competition SICK Robot Day 2014, where it coped with a highly dynamic environment and led our team to win the 2nd place (detailed in (Buck *et al.*, 2015)).

Chapter 4

High Speed Path Following Control of Skid-Steered Vehicles

4.1 Introduction

Navigation of autonomous or automated vehicles should be safe, accurate, and robust, which is especially challenging at higher speeds and on rough terrain. After a collision-free path has been planned in the surrounding environment, the vehicle should be able to exactly follow the planned route.

Path following at higher speeds on rough terrain is still an ongoing and challenging research topic. Some work on aggressive driving with car-like vehicles based on Model Predictive Control (MPC) is presented in (Williams *et al.*, 2016) and (Williams *et al.*, 2017). Off-road control of car-like vehicles at high speeds using observer-based control can be found in (Lenain *et al.*, 2011), (Lenain *et al.*, 2010) and (Deremetz *et al.*, 2017). Similar work on double-steering vehicles can be found in (Lucet *et al.*, 2009). Reactive high speed navigation on rough terrain is tackled in (Shimoda *et al.*, 2007) and (Spenko *et al.*, 2004). A comprehensive study on mobile robots on rough terrain can be found in (Iagnemma and Dubowsky, 2004). High speed path following based on pure pursuit and receding strategy is proposed in (Elbanhawi *et al.*, 2016), while backstepping steering control can be found in (Xin and Minor, 2012). All of these approaches offer interesting solutions for driving at higher speeds on rough terrain, but none of them deals with skid-steered vehicles, which are especially suitable for outdoor applications, but difficult to model and to control.

Trajectory tracking control of skid-steered vehicles can be found in (Caracciolo *et al.*, 1999), (Kozłowski and Pazderski, 2004), or (Yi *et al.*, 2007). Similar work can be found in (Kozłowski and Pazderski, 2006), (Yi *et al.*, 2009), and (Miller and Murphey, 2012). However, these approaches do not offer experimental evaluation at higher speeds. One solution to path following using skid-steered vehicles is proposed in (Pentzer *et al.*, 2014b), where a unicycle control strategy is used to control a skid-steered robot with the help of a kinematic mapping. This mapping is based on the parameter estimation proposed in (Pentzer *et al.*, 2014a). Their proposed path following solution is experimentally evaluated at speeds up to $1\,\mathrm{m\,s^{-1}}$. Path following which takes wheel slip into

account was proposed in (Rajagopalan *et al.*, 2016), where the maximum speed of the robot in the experiments reaches $1\,\mathrm{m\,s^{-1}}$. In (Ostafew *et al.*, 2016), tracking of manually defined paths with constraints using a learning-based Nonlinear Model Predictive Control (NMPC) is done with skid-steered vehicles at speeds up to $2\,\mathrm{m\,s^{-1}}$. In (Indiveri *et al.*, 2007) a unicycle control strategy from (Soetanto *et al.*, 2003) is extended for higher speeds and experimentally evaluated on a skid-steered vehicle.

In (Huskić *et al.*, 2017d) and (Huskić *et al.*, 2017b) a robust path following algorithm for skid-steered vehicles is proposed, and experimentally compared against the approaches proposed in (Pentzer *et al.*, 2014b) and (Indiveri *et al.*, 2007) at speeds up to $6\,\mathrm{m\,s^{-1}}$. This chapter is based on the work proposed in (Huskić *et al.*, 2017d) and (Huskić *et al.*, 2017b). The proposed algorithm is compared against the two already mentioned state-of-the-art algorithms on two different robots: Robotnik Summit XL and Segway RMP 440. With the Robotnik Summit XL, the comparison is made on three different terrain types, and on two different paths. Using the Segway RMP 440, the comparison is made on three different terrain types and two different path types.

Before going into mathematical and experimental details, a short description of the technology show *DLR SpaceBot Camp 2015* is given, during which a part of the mathematical framework presented in this chapter was developed.

4.2 DLR SpaceBot Camp 2015

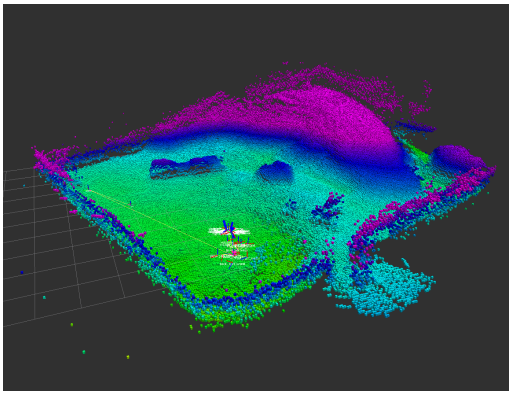
The DLR SpaceBot Camp 2015 was a robotics technology show organized by the German Aerospace Center (DLR), which replicated a scenario of exploring an unknown planet surface with a robotic system. The robotic system could have been one robot, or a team of robots, as long as the whole system weighed under 100kg. Ten different teams from different German universities and research institutes were selected and financed by the DLR.

The tasks can be divided in three main categories:

- **Finding and identifying** three special objects (a blue cup, a yellow cuboid-formed battery, and a red box as a base object);
- **Grasping** the cup and the battery, and transporting them to the base object;
- **Assembling** the three objects to a complete system.

The robotic system was supposed to solve the tasks as autonomously as possible. There was a replicated control station ”on Earth”, which was allowed to communicate with the robotic system only in special time windows. This communication had constant delays and occasional blackouts. The simulated planet surface was an arena consisting of a small hill, two rocks and a terrain mixture of gravel, macadam and sand. In Figure 4.1 the photo of the arena is shown, together with a map of it constructed by our

(a) *Simulated planet surface in the exhibition arena*



(b) *Elevation map of the arena*

Figure 4.1: *The terrain in the exhibition arena and the constructed elevation map*

robot. Since our robotic system consisted of two Robotnik Summit XL robots, it was important to study the properties of skid-steered vehicles on a challenging terrain. One of the Robotnik Summit XL robots in the exhibition arena can be seen in Figure 4.2. The default hardware was modified in order to fit the needs of the exhibition, and besides the Hall-effect sensors and the internal gyroscope, the robot was equipped with four RGB-D Asus Xtion Pro Live cameras, one 9DOF Razor Inertial Measurement Unit, one 3D laserscanner Velodyne Puck VLP-16, and two 2D laser scanners SICK Tim 5xx. For the manipulation tasks, there was a 6DOF Crustcrawler Pro-Series robotic arm made of different Dynamixel MX and AX sensors. The computer configuration consists of an Intel Core i7-4790S processor with 4 cores and 16 GB of RAM. Even though high speeds are not pursued in planet exploration with rovers, examining the behaviour of skid-steered vehicles in rough conditions resulted in a controller which is not only capable of coping with rough terrain, but also with high speeds.

4.3 Skid Steering

A skid steering mechanism is similar to the one of a differential drive, but not identical. While a differential drive has two non-steerable wheels on one axle, a skid steered vehicle has $2 \cdot k$ non-steerable wheels, where k is an integer greater than 1. Instead of wheels, a skid-steered vehicle can also have tracks, while the principle stays the same. In the rest of the text, one track, or one side of the wheels, will be referred to as a *tread*. Furthermore, it is assumed that differential drives have only contact points between the wheels and the ground, while skid-steered vehicles have contact patches between the treads and the ground.

In order for such a vehicle to steer, one tread has to move faster than the other, which means the other one needs to skid, hence the name of the mechanism. Since these vehicles always skid, it means that the nonholonomic constraint, which assumes pure rolling,

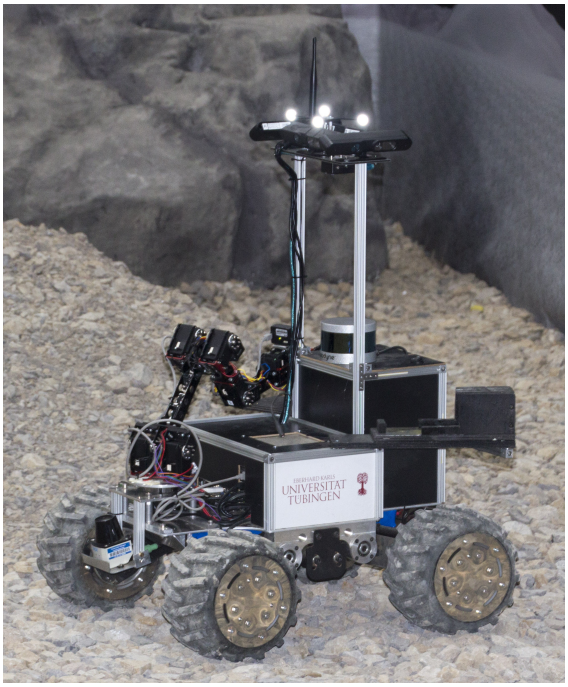


Figure 4.2: *Robotnik Summit XL in the exhibition arena.*

is violated. Such a motion highly depends on the contacts between the treads and the ground, which involves complex interactions such as torsion of both the treads and the ground surface. This makes it difficult to model such a behaviour without having a full dynamical model which includes the friction between the treads and the ground. This kind of models can be very complex and their parameters difficult to estimate. Such models can be found in (Wong, 2001) and (Yu *et al.*, 2010). Furthermore, the majority of commercial robots only offers velocities as control commands, while a real dynamical model would need torques (which are usually assumed to be linearly proportional to the motor currents). Controlling a dynamic model with velocity inputs would then require transformations such as the one proposed in (Martins *et al.*, 2017).

Driven by these reasons, for the work presented in this chapter we have embraced a type of kinematic modelling which implicitly takes dynamics into account, and can be easily configured for different robots with different dynamic characteristics. The idea for such a modelling comes from (Martínez *et al.*, 2005), and it was further developed in (Mandow *et al.*, 2007).

4.3.1 ICR Kinematics

If a skid-steered vehicle is observed as a rigid body with planar motion, there is a point in the plane in which the motion of the vehicle can be represented by pure rotation. This point is the Instantaneous Center of Rotation (ICR). It can be seen in Figure 4.3, where

the ICR is expressed in the vehicle's local coordinates (x, y). The angular velocity is denoted with ω, and the linear velocity vector is $v = (v_x, v_y)$. Control inputs for this system are the speeds of the left and the right tread, V_l and V_r.

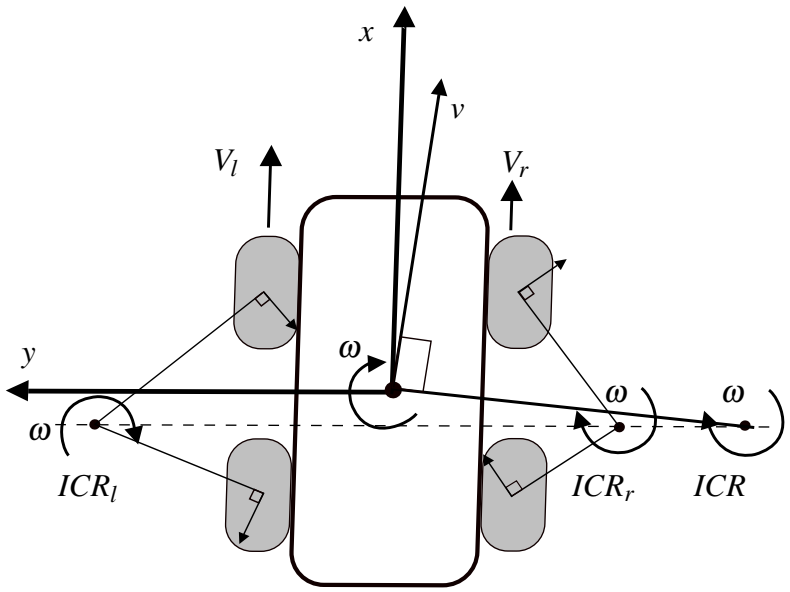


Figure 4.3: *Skid steering kinematics based on Instantaneous Centres of Rotation (ICRs). All three ICRs are estimated by solving an optimization problem whose solutions are made of ICR coordinates.*

The motion of a particle on the tread is composed of the input speed of the tread and the component that comes from the motion of the entire vehicle. This means that the treads have their own ICRs, different from the ICR of the entire vehicle. As opposed to differential drives, where the ICRs of the left and the right wheel coincide with the ground contact points, in the case of skid-steered vehicles, the tread ICRs usually lie outside of the treads. This offset is a direct consequence of the slippage between the treads and the ground, and it is terrain-dependent. In Figure 4.3, the ICRs of the left and the right tread are denoted as ICR_l and ICR_r, respectively.

As stated in (Martínez *et al.*, 2005), because of Kennedy's ICR theorem, all the three ICRs lie on the same line, and have the same x-coordinate x_{ICR}. The ICR y-coordinates of the vehicle, the left, and the right tread, are denoted as y_{ICR}, y_{ICRl}, and y_{ICRr}, respectively. Furthermore, all three ICRs have the same angular velocity.

Now, the following expressions can be derived:

$$\begin{aligned} v_x &= y_{ICR}\omega, \\ v_y &= -x_{ICR}\omega, \end{aligned} \tag{4.1}$$

which describe the motion of the entire body. The speeds of the left and the right tread

can be written as

$$\begin{aligned} V_l &= \frac{v_x - y_{ICRl}\omega}{\alpha_l}, \\ V_r &= \frac{v_x - y_{ICRr}\omega}{\alpha_r} \end{aligned} \tag{4.2}$$

where α_l and α_r are fuzzy parameters which describe mechanical issues such as tire inflation or belt transmission, as proposed in (Mandow *et al.*, 2007).

Using Eq. (4.1) and Eq. (4.2), the kinematic equations of skid-steered vehicles based on ICR parameters can be expressed as:

$$\begin{aligned} v_x &= \frac{\alpha_l V_l y_{ICRr} - \alpha_r V_r y_{ICRl}}{y_{ICRr} - y_{ICRl}}, \\ v_y &= x_{ICR}\frac{\alpha_r V_r - \alpha_l V_l}{y_{ICRr} - y_{ICRl}}, \\ \omega &= \frac{\alpha_l V_l - \alpha_r V_r}{y_{ICRr} - y_{ICRl}}. \end{aligned} \tag{4.3}$$

If $\alpha_l = \alpha_r = 1$, $x_{ICR} = 0$, $y_{ICRl} = -y_{ICRr} = w$, where $2w$ is the distance between the right and the left tread, then Eq. (4.3) represents the kinematic model of a differential drive. The ICR parameters $(x_{ICR}, y_{ICRl}, y_{ICRr}, \alpha_l, \alpha_r)$ of skid-steered vehicles can be seen as a deviation from an ideal differential drive. This allows a control law developed for skid-steered vehicles to be applied to differential drives, when the ICR parameters are adjusted accordingly.

Following the work of Caracciolo et al. in Caracciolo *et al.* (1999), the relation between the local and global velocities can be expressed as

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix}. \tag{4.4}$$

Furthermore, a nonholonomic constraint for the skid-steered vehicle is defined in Caracciolo *et al.* (1999), expressed as

$$x_{ICR}\dot{\theta} + v_y = 0. \tag{4.5}$$

This constraint can be expressed in Pfaffian form as

$$\begin{bmatrix} -\sin\theta & \cos\theta & x_{ICR} \end{bmatrix} \begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\theta} \end{bmatrix} = \boldsymbol{A}(\boldsymbol{q})\dot{\boldsymbol{q}} = \boldsymbol{0}. \tag{4.6}$$

Now, defining $\boldsymbol{\eta} = \begin{bmatrix} v_x & \omega \end{bmatrix}^T$ as a pseudo-velocity control input vector, it is possible to

express the generalized velocities as

$$\dot{\boldsymbol{q}} = \boldsymbol{S}(\boldsymbol{q})\boldsymbol{\eta}, \tag{4.7}$$

where $\boldsymbol{S}(\boldsymbol{q})$ is a full rank matrix with columns in the null space of $\boldsymbol{A}(\boldsymbol{q})$. We choose the matrix $\boldsymbol{S}(\boldsymbol{q})$ as proposed by Kozłowski et al. in Kozłowski and Pazderski (2004). This way, it is possible to use the longitudinal and angular velocity as control inputs, and not the lateral velocity, as proposed in Caracciolo *et al.* (1999). Now it is possible to describe the vehicle's motion in global coordinates as

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & x_{ICR}\sin\theta \\ \sin\theta & -x_{ICR}\cos\theta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ \omega \end{bmatrix}. \tag{4.8}$$

Here, $\dot{X}$ and $\dot{Y}$ are the linear velocity components in global coordinates, while $\dot{\theta} = \omega$, where θ is the global orientation of the vehicle. The boundedness of x_{ICR} is discussed in (Caracciolo *et al.*, 1999), (Kozłowski and Pazderski, 2004) and (Kozłowski and Pazderski, 2006). In our work, we experimentally identify the ICR parameters $(x_{ICR}, y_{ICRl}, y_{ICRr}, \alpha_l, \alpha_r)$ for different terrain types, by using evolutionary algorithms, similar as in (Mandow *et al.*, 2007). This kind of identification provides realistic values for x_{ICR} which are always bounded.

4.4 Path Following

In order to define and solve the path following problem, one practical approach is to transform the model of the vehicle to path coordinates.

4.4.1 Transformed Kinematic Model

If we observe Figure 4.4, a skid-steered vehicle is depicted similarly as in Figure 4.3, having its centre Q and orientation θ in the world frame. The geometric reference path is denoted with **P** and parametrized by the path parameter $s : [0,\infty) \rightarrow [0,\infty)$, a free control parameter which can be arbitrarily specified. For the path following problem, this parameter is usually chosen as the arc length of the path. The world frame $\{W\}$ is defined by the coordinates (X_w, Y_w) and zero O, while there is a Serret-Frenet frame $\{F\}$ moving along the path, defined by its coordinates (x_e, y_e), and having its origin in the point P. This moving reference frame is rotated in the world frame by the angle θ_t, which is the angle of the path tangent in the point P. The curvature along the path is defined as $c(s) = d\theta_t/ds$, which means that $\dot{\theta}_t = c(s)\dot{s}$. The path following problem can now be defined as making the (x_e, y_e) coordinates, as well as the difference $\theta_e = \theta - \theta_t$, converge to zero.

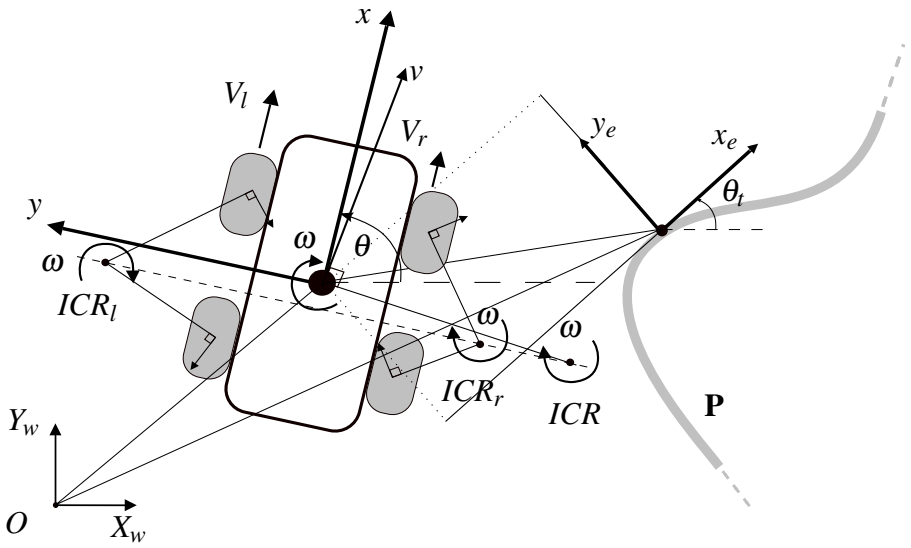


Figure 4.4: *Path following geometry with a skid-steered vehicle.*

Let us now define a vector $\mathbf{q}$, which is the robot's position vector in world coordinates, pointing from the point O to the point Q, and a vector $\mathbf{p}$, pointing from O to P, which is the position vector of the Serret-Frenet frame. If we now define a vector $\mathbf{r}$ in the Serret-Frenet frame, pointing from P to Q, the following relation holds

$$\dot{\mathbf{q}} = \dot{\mathbf{R}}(\theta_t)\mathbf{r} + \mathbf{R}(\theta_t)\dot{\mathbf{r}} + \dot{\mathbf{p}}, \tag{4.9}$$

where $\mathbf{R}(\theta_t)$ is the rotation matrix from $\{F\}$ to $\{W\}$, around the angle θ_t.

If we consider that

$$\mathbf{r} = \begin{bmatrix} x_e \\ y_e \end{bmatrix}, \mathbf{R}(\theta_t)\dot{\mathbf{p}} = \begin{bmatrix} \dot{s} \\ 0 \\ 0 \end{bmatrix}, \mathbf{q} = \begin{bmatrix} X_w \\ Y_w \\ 0 \end{bmatrix}, \tag{4.10}$$

we can derive the dynamics of the error coordinates:

$$\begin{aligned} \dot{x}_e &= \begin{bmatrix} \cos\theta_t & \sin\theta_t \end{bmatrix} \begin{bmatrix} \dot{X}_w \\ \dot{Y}_w \end{bmatrix} - \dot{s}(1 - c(s)y_e), \\ \dot{y}_e &= \begin{bmatrix} -\sin\theta_t & \cos\theta_t \end{bmatrix} \begin{bmatrix} \dot{X}_w \\ \dot{Y}_w \end{bmatrix} - c(s)\dot{s}x_e. \end{aligned} \tag{4.11}$$

Using the global velocities from Eq. (4.8), we derive the skid steering model in path

coordinates:

$$\begin{aligned}
\dot{x}_e &= v_x \cos\theta_e + x_{ICR}\omega \sin\theta_e - \dot{s}(1 - c(s)y_e), \\
\dot{y}_e &= v_x \sin\theta_e - x_{ICR}\omega \cos\theta_e - c(s)\dot{s}x_e, \\
\dot{\theta}_e &= \omega - c(s)\dot{s}, \;\; \dot{\theta}_t = c(s)\dot{s}.
\end{aligned} \tag{4.12}$$

4.4.2 Kinematic Control

The kinematic model of skid-steered vehicles in path coordinates from Eq. (4.12) can now be used to develop a path following control law. Since the model is consisting of error coordinates, it is convenient to use the Lyapunov stability theory, and to propose the Lyapunov function candidate

$$V = \frac{1}{2}\left(x_e^2 + y_e^2 + \frac{1}{\sigma}\left|\sin\left(\theta_e - \psi(y_e, v_x)\right)\right|\right), \tag{4.13}$$

where $\psi(y_e, v_x)$ is a function for smooth transient maneuvers, having the property $\psi(0, v_x) = 0$, similarly chosen as in (Micaelli and Samson, 1993) and (Soetanto *et al.*, 2003). Here, σ and k_ψ are positive parameters.

By deriving the Lyapunov function candidate, we get

$$\begin{aligned}
\dot{V} = &\frac{1}{\sigma}\frac{\sin(\theta_e - \psi(y_e, v_x))\cos(\theta_e - \psi(y_e, v_x))}{|\sin(\theta_e - \psi(y_e, v_x))|}(\dot{\theta}_e - \dot{\psi}(y_e, v_x)) + \\
&x_e(v_x \cos\theta_e + x_{ICR}\cos\theta_e + x_{ICR}\omega\sin\theta_e - \dot{s}) + \\
&y_e v_x \sin\theta_e - y_e x_{ICR}\omega\cos\theta_e.
\end{aligned} \tag{4.14}$$

In order for $\dot{V}$ to be less or equal to zero, the following expressions need to be defined:

$$\begin{aligned}
\dot{s} &= v_x \cos\theta_e + x_{ICR}\omega\sin\theta_e + \gamma x_e, \\
\dot{\theta}_e &= \dot{\psi}(y_e, v_x) + \frac{|\sin(\theta_e - \psi(y_e, v_x))|}{\sin(\theta_e - \psi(y_e, v_x))\cos(\theta_e - \psi(y_e, v_x))} \cdot \\
&\left(-\sigma y_e v_x \sin\theta_e + \sigma y_e x_{ICR}\omega\cos\theta_e - \zeta(\theta_e - \psi(y_e, v_x))^2\right),
\end{aligned} \tag{4.15}$$

where γ and ζ are positive parameters. With these expressions it is now possible to control the dynamics of the moving reference frame on the path, and the rotation of the vehicle in path coordinates, i.e. Eq. (4.15) presents the path following control for skid-steered vehicles, based on the ICR kinematic model in path coordinates.

4.4.3 Stability Analysis

In this section we analyse the convergence properties of the closed loop system, when Eq. (4.15) is applied to Eq. (4.12).

Proposition 1. *Let us assume that the linear and angular velocities follow the desired profiles, do not tend to zero, as time t tends to infinity, and are bounded, together with their first derivatives. The path curvature is also bounded, where* $c(s) \in [c_{min}, c_{max}]$*, with* c_{max} *being the maximum, and* c_{min} *the minimum allowed curvature. By applying the control from Eq.* (4.15) *to the model in Eq.* (4.12)*, the error coordinates* $x_e(t)$*,* $y_e(t)$*, and* $\theta_e(t)$ *asymptotically tend to zero, as t goes to infinity. This means that the vehicle will converge to the geometric reference path.*

Proof. The first derivative of the proposed Lyapunov function candidate defined in Eq. (4.14) will become negative semi-definite when the control from Eq. (4.15) is applied, which yields

$$\dot{V} = -\gamma x_e^2 - \frac{\zeta}{\sigma}(\theta_e - \psi(y_e, v_x))^2 \leqslant 0. \tag{4.16}$$

This means that the Lyapunov function candidate defined in Eq. (4.13) is positive, bounded, and monotonically decreasing, hence x_e, y_e and $\theta_e - \psi(y_e, v_x)$ are also bounded. By experimentally identifying a bounded x_{ICR}, the lateral velocity v_y is also bounded. Since v_x and $\psi(y_e, v_x)$ are also chosen to be bounded, and θ_e is normalized and bounded between $[-\frac{\pi}{2}, \frac{\pi}{2}]$, $\dot{s}$ and $\dot{\theta}_e$ are also bounded. This implies that $\dot{x}_e$ and $\dot{y}_e$ are bounded as well. It can be seen that

$$\ddot{V} = -2\gamma x_e \dot{x}_e - 2\frac{\zeta}{\sigma}(\theta_e - \psi(y_e, v_x))(\dot{\theta}_e - \dot{\psi}(y_e, v_x)) \tag{4.17}$$

is bounded, since $\dot{\psi}(y_e, v_x)$ is bounded as well. Because of $\ddot{V}$ being bounded, $\dot{V}$ is uniformly continuous. Now, by using Barbalat's lemma, $\dot{V}$ converges to zero, as t goes to infinity, hence $\theta_e - \psi(y_e, v_x)$ and x_e tend to zero as well. Since the boundedness of $\ddot{\theta}_e - \ddot{\psi}(y_e, v_x)$ can be derived, $\dot{\theta}_e - \dot{\psi}(y_e, v_x)$ is uniformly continuous and converges to zero. This means that the right hand side of equation

$$\begin{aligned} \dot{\theta}_e - \dot{\psi}(y_e, v_x) = \frac{|\sin(\theta_e - \psi(y_e, v_x))|}{\sin(\theta_e - \psi(y_e, v_x))\cos(\theta_e - \psi(y_e, v_x))} \cdot \\ \Big(-\sigma y_e v_x \sin\theta_e + \sigma y_e x_{ICR} \omega \cos\theta_e - \\ \zeta(\theta_e - \psi(y_e, v_x))^2 \Big), \end{aligned} \tag{4.18}$$

needs to tend to zero as well. The two-sided limit

$$\lim_{\theta_e-\psi(y_e,v_x)\to 0} \frac{|\sin(\theta_e-\psi(y_e,v_x))|}{\sin(\theta_e-\psi(y_e,v_x))\cos(\theta_e-\psi(y_e,v_x))} \tag{4.19}$$

does not exist, in this case the convergence needs to be analysed from the left and from the right. To simplify the notation, if we replace $\theta_e-\psi(y_e,v_x)$ with u, we will have

$$\begin{aligned}\lim_{u\to 0^-} \frac{|\sin u|}{\sin u\cos u} &= \lim_{u\to 0^-} \frac{1}{\cos u} \lim_{u\to 0^-} \frac{|\sin u|}{\sin u} \\ &= \lim_{u\to 0^-} \frac{|\sin u|}{\sin u} = -1,\end{aligned} \tag{4.20}$$

and similarly

$$\lim_{u\to 0^+} \frac{|\sin u|}{\sin u\cos u} = 1. \tag{4.21}$$

Since $\theta_e-\psi(y_e,v_x)$ tends to zero, the expression $\zeta(\theta_e-\psi(y_e,v_x))^2$ tends to zero as well. Now the expressions $\sigma y_e x_{ICR}\omega\cos\theta_e$ and $-\sigma y_e v_x\sin\theta_e$ need to tend to zero. Taking into account the assumptions about v_x, x_{ICR}, and ω, it is clear that y_e converges to zero. This implies that $\psi(y_e,v_x)$ tends to zero as well, which further implies that θ_e also tends to zero. Now that all the error coordinates (x_e,y_e,θ_e) converge to zero, as t tends to infinity, the vehicle will asymptotically converge to the reference path. □

4.5 Speed Control

The reachable curvature for a skid-steered vehicle can be expressed as

$$c = \frac{\alpha_r V_r - \alpha_l V_l}{\sqrt{(\alpha_r y_{ICRl} V r - \alpha_l y_{ICRr} V_l)^2 + (-\alpha_r x_{ICR} V_r + \alpha_l x_{ICR} V_l)^2}}. \tag{4.22}$$

If we want to forbid turning on the spot while driving, similarly as in (Indiveri *et al.*, 2007), we need to restrict the wheel speeds to be of a same sign, limited with a maximum value V_m, i.e. $V_l, V_r \in [0, V_m]$.

To find the curvature extrema, we need to observe two cases. In the first case we have $V_l = 0$, and $V_r = V_m$, and the maximum curvature can be derived as

$$c_{max} = \frac{1}{\sqrt{y_{ICRl}^2 + x_{ICR}^2}}. \tag{4.23}$$

In this case, the vehicle's speed can be computed as

$$v_x = \frac{\alpha_r y_{ICRl} V_m}{y_{ICRl} - y_{ICRr}}. \tag{4.24}$$

In the second case, we have $V_r = 0$, and $V_l = V_m$, and the minimum curvature is expressed as

$$c_{min} = -\frac{1}{\sqrt{y_{ICRr}^2 + x_{ICR}^2}}, \tag{4.25}$$

while the vehicle's speed is

$$v_x = -\frac{\alpha_l y_{ICRr} V_m}{y_{ICRl} - y_{ICRr}}. \tag{4.26}$$

When planning the reference path, it should be assured that $c(s) \in [c_{min}, c_{max}]$.

If the loop is closed with the control law from Eq. (4.15), the curvature in the closed loop can be expressed as

$$c_{cl} = \frac{\dot{\theta}_e + c\dot{s}}{\sqrt{v_x^2 + [x_{ICR}(\dot{\theta}_e + c\dot{s})]^2}}. \tag{4.27}$$

The speeds of the left and right tread in the closed loop are then

$$\begin{aligned} V_l &= \frac{v_x - y_{ICRl}(\dot{\theta}_e + c\dot{s})}{\alpha_l} \\ &= \frac{v_x - y_{ICRl} \cdot c_{cl} \cdot \sqrt{v_x^2 + \left[x_{ICR}(\dot{\theta}_e + c\dot{s})\right]^2}}{\alpha_l}, \\ V_r &= \frac{v_x - y_{ICRr}(\dot{\theta}_e + c\dot{s})}{\alpha_r} \\ &= \frac{v_x - y_{ICRr} \cdot c_{cl} \cdot \sqrt{v_x^2 + \left[x_{ICR}(\dot{\theta}_e + c\dot{s})\right]^2}}{\alpha_r}. \end{aligned} \tag{4.28}$$

The maximum speeds of the left and the right tread can now be expressed as

$$\begin{aligned} \max\{V_l\} &= \frac{v_x + |y_{ICRl} c_{cl}| \sqrt{v_x^2 + [x_{ICR}(\dot{\theta}_e + c\dot{s})]^2}}{\alpha_l}, \\ \max\{V_r\} &= \frac{v_x + |y_{ICRr} c_{cl}| \sqrt{v_x^2 + [x_{ICR}(\dot{\theta}_e + c\dot{s})]^2}}{\alpha_r}. \end{aligned} \tag{4.29}$$

Since the convergence of the closed system variables has been proven, the following

properties can be used:

$$\begin{aligned} \lim_{t\to\infty} c_{cl} &= \frac{c}{\sqrt{1+(x_{ICR}c)^2}}, \\ \lim_{t\to\infty} \dot{\theta}_e &= 0, \\ \lim_{t\to\infty} \dot{s} &= v_x, \end{aligned} \tag{4.30}$$

in order to compute Eq. (4.29) when $t \to \infty$. The maximum speeds of the left and the right tread are now

$$\begin{aligned} \max\left\{V_l\right\} &= \frac{v_x(1+|y_{ICRl}c|)}{\alpha_l}, \\ \max\left\{V_r\right\} &= \frac{v_x(1+|y_{ICRr}c|)}{\alpha_r}. \end{aligned} \tag{4.31}$$

Assuming that the vehicle's motors exhibit the same maximum speed, we can write $\max\left\{V_l\right\} = \max\left\{V_r\right\} = V_m$. The speed of the entire vehicle can be then expressed as

$$\begin{aligned} v_x &= \frac{\alpha_l V_m}{1+|y_{ICRl}c|}, \\ v_x &= \frac{\alpha_r V_m}{1+|y_{ICRr}c|}, \end{aligned} \tag{4.32}$$

depending whether we use the left or the right tread to compute it.

Now, the following speed control scheme can be introduced:

$$\begin{aligned} \omega \geqslant 0 : v_x &= \begin{cases} \frac{-\alpha_r y_{ICRl} V_m}{y_{ICRr}-y_{ICRl}}, & V \geqslant \varepsilon. \\ \frac{\alpha_r V_m}{1+|y_{ICRr}c|}, & V < \varepsilon. \end{cases} \\ \omega < 0 : v_x &= \begin{cases} \frac{\alpha_l y_{ICRr} V_m}{y_{ICRr}-y_{ICRl}}, & V \geqslant \varepsilon. \\ \frac{\alpha_l V_m}{1+|y_{ICRl}c|}, & V < \varepsilon, \end{cases} \end{aligned} \tag{4.33}$$

where ε is a positive parameter. As stated in (Huskić *et al.*, 2017d), since skid-steered vehicles behave asymmetrically, depending on the mechanical construction and on the terrain structure, the speed should be adjusted accordingly. This is the reason why a distinction is made whether the vehicle is turning right or left.

If the vehicle is turning left, the right tread speed V_r is dominant, and V_m can be applied, in order to maximize the speed, and to take the actuator saturation into account. The exhibited speed should never be greater than V_m. Furthermore, the Lyapunov function candidate from Eq. (4.13) is used here as a path following error measure. If the error is greater than a positive threshold ε , the expression from Eq. (4.24) is used. On the other side, if the error is smaller than ε, the speed of the entire vehicle is computed using Eq.

(4.32). Similar analysis can be made if the left tread speed V_l is dominant.

By using this speed control scheme, asymmetric behaviour of the vehicle is taken into account, together with the path following error, and the actuator saturation. The vehicle will not turn on spot while driving, and the speed is maximized.

The path following control law defined in Eq. (4.15), combined with the speed control defined in Eq. (4.33), presents an integral solution for path following of skid-steered vehicles at higher speeds on different terrain types. This algorithm we refer to as *HBZ*, by using the surname initials of the authors of the paper in which we initially proposed this approach (Huskić *et al.*, 2017d).

4.6 Dynamic Control

The control law (4.15) deals explicitly with the kinematic model of skid-steered vehicles, and the dynamics is only implicitly included. In order to account for the dynamics explicitly, let us consider a generalized dynamic model of a nonholonomic robot with generalized coordinates $(q_1, ..., q_n)$, subject to m constraints. This generalized model can be described as

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{R}(\dot{\boldsymbol{q}}) = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{\tau}, \tag{4.34}$$

where $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ is a symmetric, positive definite inertia matrix, $\boldsymbol{R}(\dot{\boldsymbol{q}}) \in \mathbb{R}^{n\times 1}$ represents generalized resistive forces, $\boldsymbol{B}(\boldsymbol{q}) \in \mathbb{R}^{n\times r}, r = n-m$, is the input transformation matrix, and $\boldsymbol{\tau} \in \mathbb{R}^{n\times 1}$ is the input vector.

As stated in (Kozłowski and Pazderski, 2004), the model (4.34) describes the dynamics of a free body and does not include the nonholonomic constraint (4.5). For this purpose, as proposed in (Caracciolo *et al.*, 1999), a vector of Lagrange multipliers $\boldsymbol{\lambda}$ is introduced such that

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{R}(\dot{\boldsymbol{q}}) = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{\tau} + \boldsymbol{A}^T(\boldsymbol{q})\boldsymbol{\lambda}. \tag{4.35}$$

Now Eq. (4.35) offers a dynamic model with the nonholonomic constraint. However, in order to control the system, it would be more suitable to express the model in terms of the velocity vector $\boldsymbol{\eta}$. In order to do so, Eq. (4.35) needs to be multiplied by $\boldsymbol{S}^T(\boldsymbol{q})$, which results in

$$\boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{q}\boldsymbol{R}(\dot{\boldsymbol{q}}) = \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{B}(\boldsymbol{q})\boldsymbol{\tau} + \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{A}^T(\boldsymbol{q})\boldsymbol{\lambda}. \tag{4.36}$$

By deriving Eq. (4.7) we get

$$\ddot{\boldsymbol{q}} = \dot{\boldsymbol{S}}(\boldsymbol{q})\boldsymbol{\eta} + \boldsymbol{S}(\boldsymbol{q})\dot{\boldsymbol{\eta}}. \tag{4.37}$$

Including Eq. (4.36) into Eq. (4.37), we get

$$\boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{S}}(\boldsymbol{q})\eta + \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{M}(\boldsymbol{q})\boldsymbol{S}(\boldsymbol{q})\dot{\boldsymbol{\eta}} + \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{R}(\boldsymbol{q}) = \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{B}(\boldsymbol{q})\boldsymbol{\tau} + \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{A}^T(\boldsymbol{q})\boldsymbol{\lambda}. \tag{4.38}$$

Now, by noticing that $\boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{A}^T(\boldsymbol{q})\boldsymbol{\lambda} = \boldsymbol{0}$, and by setting

$$\boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{M}(\boldsymbol{q})\boldsymbol{S}(\boldsymbol{q}) = \bar{\boldsymbol{M}},\ \ \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{S}}(\boldsymbol{q}) = \bar{\boldsymbol{C}},\ \ \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{R}(\boldsymbol{q}) = \bar{\boldsymbol{R}},\ \ \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{B}(\boldsymbol{q}) = \bar{\boldsymbol{B}}, \tag{4.39}$$

we get

$$\bar{\boldsymbol{M}}\dot{\boldsymbol{\eta}} + \bar{\boldsymbol{C}}\boldsymbol{\eta} + \bar{\boldsymbol{R}} = \bar{\boldsymbol{B}}\boldsymbol{\tau}. \tag{4.40}$$

The complete equations of motion of a nonholonomic mobile robot are now defined by

$$\begin{aligned} \dot{\boldsymbol{q}} &= \boldsymbol{S}(\boldsymbol{q})\boldsymbol{\eta}, \\ \bar{\boldsymbol{M}}\dot{\boldsymbol{\eta}} + \bar{\boldsymbol{C}}\boldsymbol{\eta} + \bar{\boldsymbol{R}} &= \bar{\boldsymbol{B}}\boldsymbol{\tau}. \end{aligned} \tag{4.41}$$

As suggested in (Fierro and Lewis, 1995), let $\boldsymbol{u}$ be an auxiliary input, and apply the nonlinear feedback

$$\boldsymbol{\tau} = \bar{\boldsymbol{B}}^{-1}(\boldsymbol{q})\left[\bar{\boldsymbol{M}}(\boldsymbol{q})\boldsymbol{u} + \bar{\boldsymbol{C}}\boldsymbol{\eta} + \bar{\boldsymbol{R}}\right], \tag{4.42}$$

the dynamic control problem can be converted into the kinematic control problem

$$\dot{\boldsymbol{q}} = \boldsymbol{S}(\boldsymbol{q})\boldsymbol{\eta}, \dot{\boldsymbol{\eta}} = \boldsymbol{u}. \tag{4.43}$$

Path following approaches usually deal with the kinematic part of the model, i.e. with (4.7). The path following control law from Eq. (4.15) deals with the same issue, only transformed to path coordinates. The velocity vector $\boldsymbol{\eta}$ is assumed to track the desired velocity profile perfectly. However, this perfect tracking is not assured by a pure kinematic control law. In order to guarantee that the velocity will track its desired profile, the above introduced procedure needs to be employed. This way, the prescribed control $\boldsymbol{\eta}$ is converted into a torque control $\boldsymbol{\tau}$, such that the desired velocity behaviour is exhibited, and parameters such as mass, friction, etc., are taken into account.

4.6.1 Backstepping Kinematics into Dynamics for Skid-Steered Vehicles

To lift the assumption of a perfect speed tracking, we need to employ the technique of backstepping. More details can be found in (Khalil and Grizzle, 1996), (Fierro and Lewis, 1995), and (Soetanto *et al.*, 2003). Let us define a virtual control law for $\dot{\theta}_e$, which represents the desired behaviour of $\dot{\theta}_e$ in (4.15) as

$$\begin{aligned} \phi = \dot{\psi}(y_e, v_x) + \frac{|\sin(\theta_e - \psi(y_e, v_x))|}{\sin(\theta_e - \psi(y_e, v_x))\cos(\theta_e - \psi(y_e, v_x))} \cdot \\ \left(-\sigma y_e v_x \sin\theta_e + \sigma y_e x_{ICR}\omega\cos\theta_e - \zeta(\theta_e - \psi(y_e, v_x))^2 \right). \end{aligned} \tag{4.44}$$

Now, let $\varepsilon = \dot{\theta}_e - \phi$ be the difference between the actual and desired values of $\dot{\theta}_e$. If we replace $\dot{\theta}_e$ by $\varepsilon + \phi$ in the expression for $\dot{V}$, we get

$$\dot{V} = -\gamma x_e^2 - \frac{\zeta}{\sigma}(\theta_e - \psi(y_e, v_x))^2 + \frac{1}{\sigma}\frac{|\sin(\theta_e - \psi(y_e, v_x))|}{\sin(\theta_e - \psi(y_e, v_x))\cos(\theta_e - \psi(y_e, v_x))}\varepsilon. \quad (4.45)$$

The function V can now be augmented in a way to take into account ε and a desired velocity profile v_d as

$$V_1 = V + \frac{1}{2}\left[\varepsilon^2 + (v_x - v_d)^2\right]. \quad (4.46)$$

The first time derivative of Eq. (4.46) is then

$$\dot{V}_1 = \dot{V} + \varepsilon\dot{\varepsilon} + (v_x - v_d)(\dot{v} - \dot{v}_d). \quad (4.47)$$

In order for the first derivative of the new Lyapunov candidate to be negative semidefinite, i.e. $\dot{V}_1 \leqslant 0$, the following expressions need to be defined

$$\begin{aligned} \dot{\varepsilon} &= -\frac{1}{\sigma}\frac{|\sin(\theta_e - \psi(y_e, v_x))|}{\sin(\theta_e - \psi(y_e, v_x))\cos(\theta_e - \psi(y_e, v_x))}\varepsilon - k_\varepsilon\varepsilon, \\ \dot{v}_x &= \dot{v}_d - k_{v_d}(v_x - v_d). \end{aligned} \quad (4.48)$$

Since $\theta_e = \theta - \theta_t$, then $\dot{\theta}_e = \dot{\theta} - \dot{\theta}_t = \omega - c(s)\dot{s}$, and $\ddot{\theta}_e = \dot{\omega} - c\ddot{s} - g\dot{s}^2$, where $g = \frac{dc(s)}{ds}$. Now, the derivative of ε can be written as

$$\dot{\varepsilon} = \ddot{\theta}_e - \dot{\phi} = \dot{\omega} - c(s)\ddot{s} - g\dot{s}^2 - \dot{\phi}. \quad (4.49)$$

The auxiliary control vector can be then defined as

$$\dot{\boldsymbol{\eta}} = \begin{bmatrix} \dot{v}_x \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} \dot{v}_d - k_{v_d}(v_x - v_d) \\ \dot{\phi} - \frac{1}{\sigma}\frac{|\sin(\theta_e - \psi(y_e, v_x))|}{\sin(\theta_e - \psi(y_e, v_x))\cos(\theta_e - \psi(y_e, v_x))}\varepsilon - k_\varepsilon\varepsilon + c(s)\ddot{s} + g\dot{s}^2 \end{bmatrix}. \quad (4.50)$$

The control from Eq. (4.50) represents the acceleration control vector of the entire system (4.43), making sure that the velocity vector tracks the desired profile, and that the robot follows the geometric reference path, taking all the dynamic parameters into consideration.

Proposition 2. *Let us consider the dynamic model* (4.43) *together with the control law* (4.50) *and the virtual vehicle dynamics from* (4.15). *Let v_d be a bounded desired velocity profile with bounded first and second order derivatives. Furthermore, v_d does not tend to zero with $t \to \infty$. Then x_e, y_e, θ_e and $v_x - v_d$ tend asymptotically to zero, as time goes to infinity.*

Proof. The proof is similar to the one in Section 4.4.3. Since $\dot{V}_1$ is negative semidefinite and bounded below, V_1 is positive, bounded and monotonically decreasing. Therefore,

x_e, y_e, θ_e, ε and v_x are also bounded, since $\psi(y_e, v_x)$ and v_d are assumed to be bounded. Then, $\dot{x}_e$, $\dot{y}_e$, $\dot{\varepsilon}$ and $\dot{v}_x$ are bounded as well. Since $\ddot{V}_1$ is bounded, $\dot{V}_1$ is uniformly continuous, and by Barbalat's lemma, $\dot{V}_1 \to 0$ as $t \to \infty$. As a consequence, the variables x_e, y_e, θ_e, $\psi(y_e, v_x)$, ε and $v_x - v_d$ tend to zero, as t tends to infinity. This means that the vehicle converges to the geometric reference path, while tracking the desired speed profile asymptotically. □

4.7 The SLPINL Controller

This controller is a state-of-the-art controller used for comparison with the proposed controller. It is based on the work presented in (Soetanto *et al.*, 2003), and its extension is presented in (Indiveri *et al.*, 2007). The name SLPINL comes from the surname initials of the authors of both contributions. Using a similar notation and reasoning, as for deriving

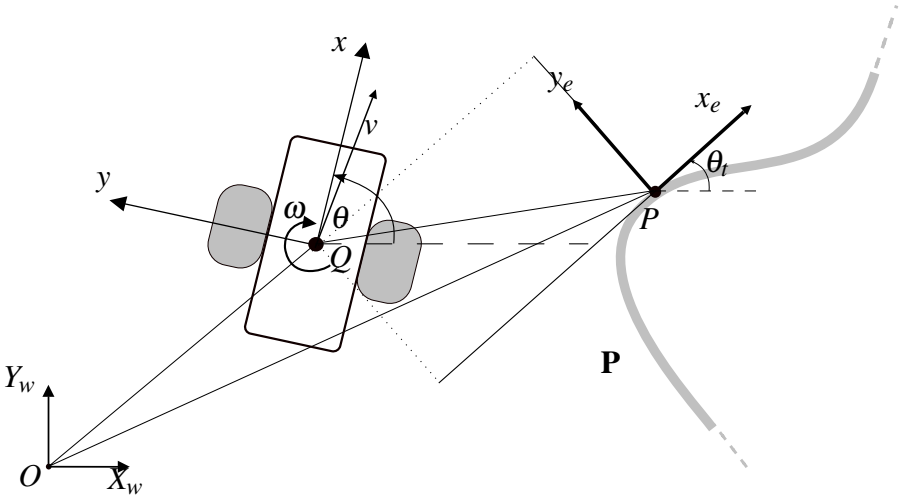


Figure 4.5: *Geometry of the SLPINL algorithm.*

Eq. (4.12), by observing Figure 4.5 we get a kinematic model of a unicycle in path coordinates as

$$\begin{aligned} \dot{x}_e &= -\dot{s}(1 - c(s)y_e) + v\cos\theta_e, \\ \dot{y}_e &= -c(s)\dot{s}x_e + v\sin\theta_e, \\ \dot{\theta}_e &= \omega - c(s)\dot{s}, \end{aligned} \tag{4.51}$$

where $c(s)$ is the path curvature, s an arbitrary path parameter, and (x_e, y_e, θ_e) are the error coordinates, as defined before.

By using the Lyapunov function candidate

$$V = \frac{1}{2}(x_e^2 + y_e^2) + \frac{1}{2\gamma}\big(\theta_e - \delta(y_e, v)\big)^2, \tag{4.52}$$

where $\lim_{t\to\infty} v(t) \neq 0$, $\delta(0,v) = 0$, and $y_e v \sin\delta(y_e,v) \leqslant 0, \forall y_e \forall v$, the following path following control law can be derived

$$\begin{aligned} \dot{s} &= v\cos\theta_e + k_1 x_e, \\ \dot{\theta}_e &= \dot{\delta} - \gamma y_e v \frac{\sin\theta_e - \sin\delta}{\theta_e - \delta} - k_2(\theta_e - \delta), \end{aligned} \tag{4.53}$$

where k_1, k_2 and γ are positive parameters.

This kinematic control law proposed in (Soetanto *et al.*, 2003) is extended in (Indiveri *et al.*, 2007) by proposing an additional speed control

$$v = \begin{cases} \frac{V_m}{2}, & V \geqslant \varepsilon \\ \frac{V_m}{1+b|\kappa(s)|}, & V < \varepsilon \end{cases}, \tag{4.54}$$

where V_m denotes the maximum speed of the vehicle, $\kappa(s)$ the reference path curvature, and b is a positive parameter. Similar as for Eq. (4.33), the speed is limited if the path following error measured by the Lyapunov function candidate is large, and the speed increases, when the path curvature decreases. Again, actuator saturation is taken into account, and the vehicle will not turn on the spot at higher speeds.

The SLPINL algorithm, initally proposed in (Soetanto *et al.*, 2003), and then extended in (Indiveri *et al.*, 2007), offers a very efficient path following control for unicycles, and can even be used on skid-steered vehicles at moderate speeds (see (Indiveri *et al.*, 2007)).

4.8 The PBR Controller

This controller is another state-of-the-art controller used for comparison. It is proposed in (Pentzer *et al.*, 2014b), and its name is an abbreviation of the authors' surname initials. It is a path following algorithm designed for skid-steered vehicles, where the control law is initially developed for unicycles, but the reference path is transformed in a way to account for the ICR coordinates of a skid-steered vehicle. This relies on the fact that a skid-steered vehicle can be seen as a differential drive with an offset equal to the ICR coordinates. Discussion on this can also be found in (Huskić *et al.*, 2017d) and (Mandow *et al.*, 2007). By observing Figure 4.6, the following kinematic equations can be derived

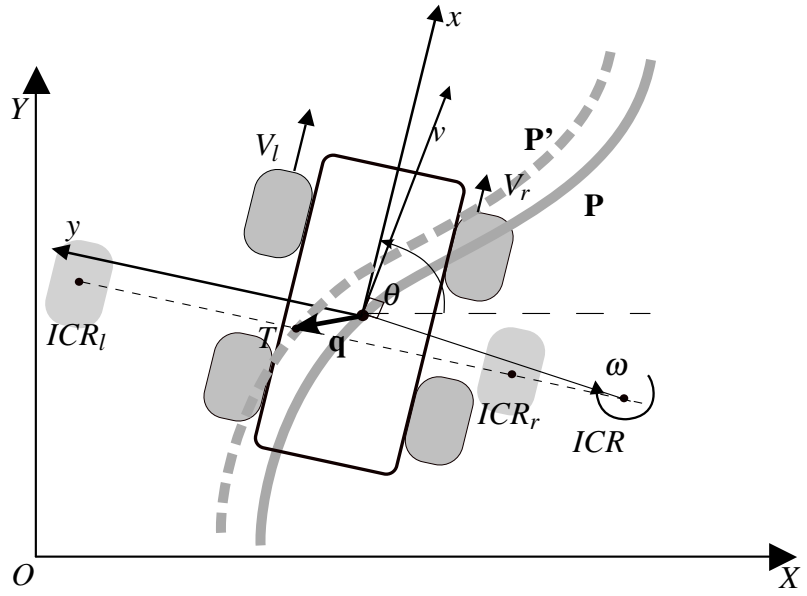


Figure 4.6: *Geometry of the PBR algorithm.*

$$
\begin{aligned}
v_x &= \frac{V_l y_{ICRr} - V_r y_{ICRl}}{y_{ICRr} - y_{ICRr}}, \\
v_y &= \frac{\left(V_r - V_l\right) x_{ICR}}{y_{ICRr} - y_{ICRl}}, \\
\omega &= \frac{V_l - V_r}{y_{ICRr} - y_{ICRl}},
\end{aligned}
\tag{4.55}
$$

which is very similar to the ICR model presented in Eq. (4.3), just without the additional parameters α_l and α_r. Both of these models are based on the work proposed in (Martínez *et al.*, 2005) and (Mandow *et al.*, 2007).

In order to estimate the ICR coordinates, this approach uses an online Extended Kalman Filter (EKF), as described in (Pentzer *et al.*, 2014b), with more details in (Pentzer *et al.*, 2014a). The equations of motion for the EKF are expressed as

$$
\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\theta} \\ \dot{y}_{ICRr} \\ \dot{y}_{ICRl} \\ \dot{x}_{ICR} \end{bmatrix} = \begin{bmatrix} v_x \cos\theta - v_y \sin\theta + w_X \\ v_x \sin\theta - v_y \cos\theta + w_Y \\ \frac{V_l - V_r}{y_{ICRr} - y_{ICRl}} + w_\omega \\ w_r \\ w_l \\ w_x \end{bmatrix},
\tag{4.56}
$$

where w_X, w_Y, w_ω, w_r, w_l, w_x are additive zero-mean Gaussian process noises, such that the ICR coordinates are assumed to be constant. Global localization and measurements of wheel or track speeds are required to be fused with Eq. (4.56) for the EKF to produce

estimates of the ICR coordinates. Further details and convergence analysis can be found in (Pentzer *et al.*, 2014a).

Now that the ICR coordinates are estimated, the next control law is used

$$\begin{aligned} v &= u(t), \\ \omega &= -k_1 v d \frac{\sin\theta_e}{\theta_e} - k_2 |v| \theta_e, \end{aligned} \tag{4.57}$$

where the function $u(t)$ must satisfy $\lim_{t\to\infty} u(t) \neq 0$, and k_1 and k_2 are positive parameters. Similar as before, $\theta_e = \theta - \theta_t$ is the robot orientation in path coordinates, with θ being the global robot orientation, and θ_t the tangent angle in the nearest point on the path, and d is the length of the orthogonal projection to the path. More details on this control law can be found in (de Wit *et al.*, 1993).

The control in Eq. (4.57) is developed for unicycles, but in order to use it for skid-steered vehicles, the ICR coordinates need to be taken into account. As it can be seen in Figure 4.6, the left and the right ICR can be considered as the left and the right wheels of a virtual differential drive, whose center is translated from the center of the skid-steered vehicle by the vector

$$\mathbf{q} = \begin{bmatrix} q_x \\ q_y \end{bmatrix} = \begin{bmatrix} x_{ICR} \\ \frac{y_{ICRl} - y_{ICRr}}{2} \end{bmatrix}. \tag{4.58}$$

So, the vector $\mathbf{q}$ actually represents an offset between the original skid-steered vehicle and the virtual differential drive. If the original reference path **P** is translated by this vector $\mathbf{q}$, such that the virtual differential drive tracks the new path **P'**, by using the control in Eq. (4.57), the original vehicle will follow the original path **P**.

The PBR controller provides an interesting solution for skid-steered vehicles with online ICR estimation, relying heavily on the EKF performance. More details can be found in (Pentzer *et al.*, 2014b) and (Pentzer *et al.*, 2014a).

4.9 Experimental Evaluation

The experimental evaluation of the proposed algorithm (HBZ) consists of two parts. First, the algorithm is evaluated on a Robotnik Summit XL robot, up to its practical maximum speed of $2.5\,\mathrm{m\,s^{-1}}$, on three different terrain types. The proposed approach is experimentally compared against the SLPINL and the PBR controller.

The second part of the evaluation is done on a Segway RMP 440 robot, up to the speeds of $6\,\mathrm{m\,s^{-1}}$. Similar as with the Robotnik Summit XL, an experimental comparison between SLPINL, PBR and HBZ is made on three different terrain types.

For both evaluations parts, an offline estimation of the ICR parameters is made for different terrain types, similar as in (Mandow *et al.*, 2007), by using evolutionary algorithms. For the implementation, two software tools were used: CS::APEX, detailed in (Buck *et al.*, 2016b), and EvA2, detailed in (Kronfeld *et al.*, 2010).

All three controllers, HBZ, SLPINL and PBR, are implemented in C++/ROS (detailed in (Quigley *et al.*, 2009)), and integrated into the GeRoNa framework, a generic and modular navigation framework for any wheeled mobile robot, described in Chapter 6.

4.9.1 Robotnik Summit XL

With a practical maximum speed of $\approx 2.5\text{m}\,\text{s}^{-1}$, and the weight of $\approx 50\text{kg}$, the Robotnik Summit XL is a medium-sized skid-steered robot with good performance. The robot used in the experiments presented in this section is shown in Figure 4.7. The computer configuration is the same as described in Section 4.2, while the sensor equipment used for these experiments consists of the internal Hall-effect sensors, the internal gyroscope, and a 3D laser scanner Velodyne Puck VLP-16 together with a 9DOF Razor IMU.

Figure 4.7: *The Robotnik Summit XL robot used in the experiments.*

The experiments with this robot were conducted on the following terrain types:

1. relatively flat, grassy terrain;

2. very smooth vinyl floor;

3. macadam - crushed stones mixed with dust and sand.

For each experiment, PBR and SLPINL were commanded with the same desired speed, in order to achieve similar mean speeds. The proposed approach was commanded with even higher speeds, in order to emphasize its advantages. A video demonstrating the experiments can be found at: https://youtu.be/plaoHHEfM3o.

Table 4.1: *Experiments with the Robotnik Summit XL robot on grass, at the maximum speed at which each algorithm could perform. For the case of HBZ, the maximum speed of the robot is reached.*

Algorithm	Mean Speed [$\mathrm{m\,s^{-1}}$]	Max Speed [$\mathrm{m\,s^{-1}}$]	Mean Error [m]	Max Error [m]
SLPILN	1.55	2.25	0.56	1.83
PBR	1.94	2.29	0.83	3.61
HBZ	**2.15**	**2.53**	**0.07**	**0.22**

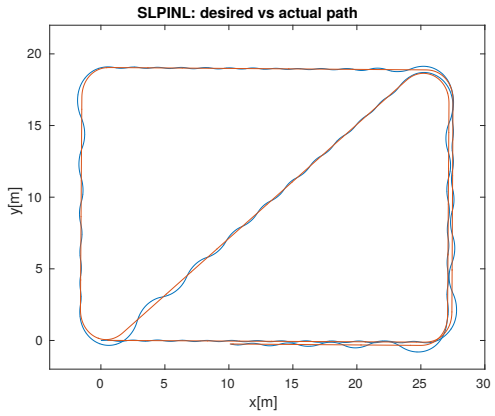
Grass On the grassy terrain, a 159.83m long path was used, having long segments, and abrupt curves. The path was covering a small football field behind our institute building, hence the rectangular-like shape. For the ground truth, a fusion was made using an internal gyroscope and the odometry coming from Hall effect sensors. The localization error was measured while following a planned path at different speeds, by using CS::APEX, proposed in (Buck *et al.*, 2016b). The resulting error at the average speed of $0.89\mathrm{m\,s^{-1}}$ was $(x_{err}, y_{err}, \theta_{err}) = (0.015\mathrm{m}, 0.69\mathrm{m}, 2.9°)$. At the average speed of $2.15\mathrm{m\,s^{-1}}$, the error was $(x_{err}, y_{err}, \theta_{err}) = (0.62\mathrm{m}, -0.65\mathrm{m}, 3.85°)$.

Path following performance of SLPINL, PBR and HBZ at some representative speeds on grass can be seen in Figure 4.8 and Figure 4.9. A direct comparison between the three algorithms at different speeds can be observed in Figure 4.10.

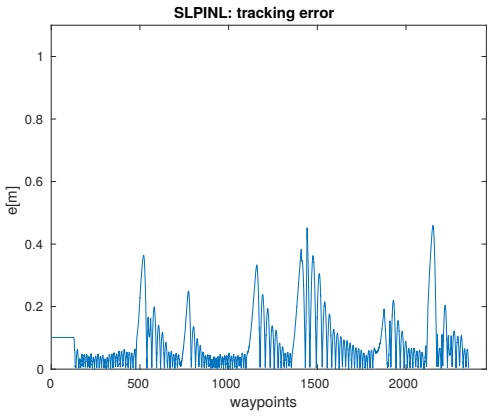
The mean and maximum values of the speed and error, when driving at higher speeds, can be seen in Table 4.1. For SLPINL, when the desired speed is set to $2\mathrm{m\,s^{-1}}$, the maximum measured value is $2.25\mathrm{m\,s^{-1}}$, and the system reaches its critical oscillations, where the error reaches 1.83m. Because of the inherent speed control of the algorithm, the overall mean speed is then $1.55\mathrm{m\,s^{-1}}$. For the case of PBR, when the desired speed is set to $2\mathrm{m\,s^{-1}}$, the mean speed is $1.94\mathrm{m\,s^{-1}}$, the error reaches $3.61\mathrm{m\,s^{-1}}$, and the system is on the border of stability. The mean error in this case is 83cm. When the HBZ controller is commanded with $2.5\mathrm{m\,s^{-1}}$, the mean speed is $2.15\mathrm{m\,s^{-1}}$. The maximum measured error is 22cm in this case, while the mean error is 7cm.

The Robotnik Summit XL robot can theoretically reach a velocity of $3\mathrm{m\,s^{-1}}$, if it would move on a perfectly straight line. Since there always has to be some correction control, i.e. the vehicle needs to slightly turn, then the left or the right wheels need to drive faster than the commanded speed, which is not feasible. This is the reason why the practical maximum speed lies at $\approx 2.5\mathrm{m\,s^{-1}}$. In this way, the left and right wheels can drive at speeds up to $3\mathrm{m\,s^{-1}}$, in order to steer.

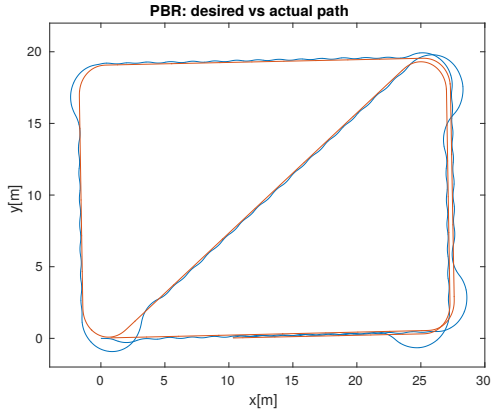
Vinyl On vinyl and macadam, a lemniscate was used as a reference path, overlaid five times, so the robot needed to follow the curve five times in a row in order to finish one run. This way, the repeatability of the algorithms was tested. Furthermore, the curvature of the lemniscate was chosen to be very high, when compared to the dimensions of the



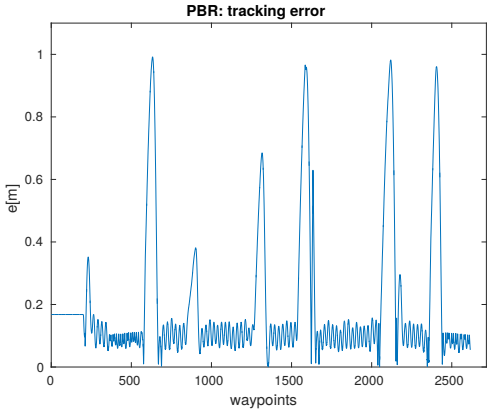
(a) *SLPINL at* $v_{mean} = 1.38\,\mathrm{m\,s^{-1}}$



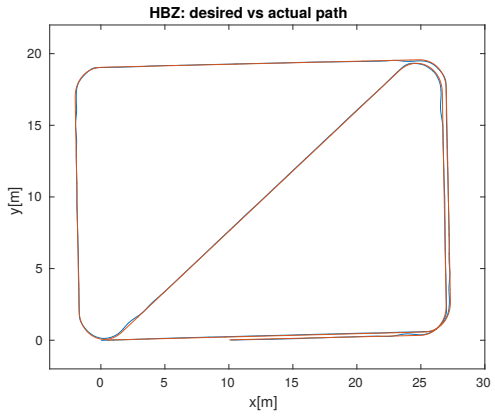
(b) *SLPINL at* $v_{mean} = 1.38\,\mathrm{m\,s^{-1}}$



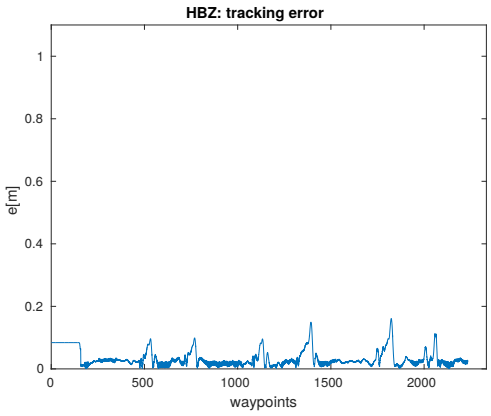
(c) *PBR at* $v_{mean} = 1.49\,\mathrm{m\,s^{-1}}$



(d) *PBR at* $v_{mean} = 1.49\,\mathrm{m\,s^{-1}}$

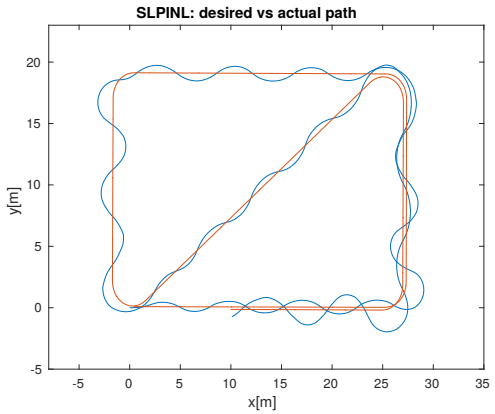


(e) *HBZ at* $v_{mean} = 1.73\,\mathrm{m\,s^{-1}}$

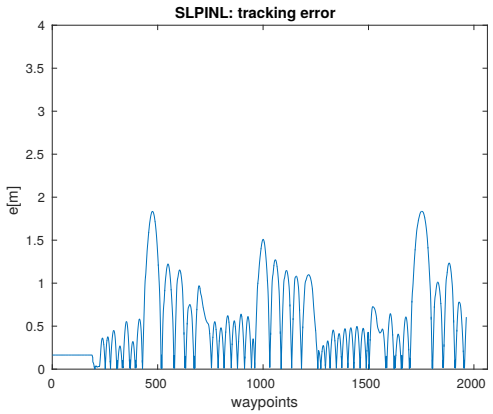


(f) *HBZ at* $v_{mean} = 1.73\,\mathrm{m\,s^{-1}}$

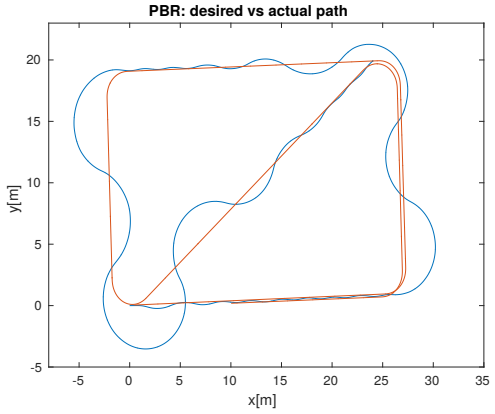
Figure 4.8: *Path following performance with Robotnik Summit XL on grass, following a* 159.83m *long path. The desired speed is the maximum speed at which SLPINL and PBR still perform without major oscillations.*



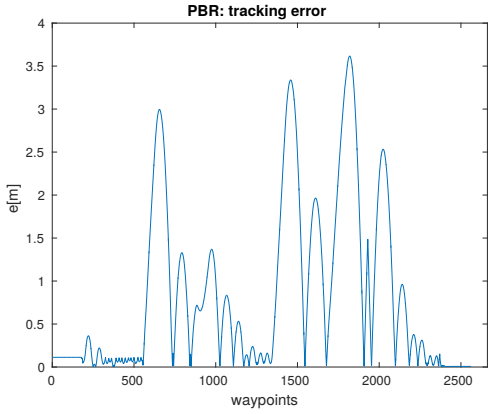
(a) *SLPINL at* $v_{mean} = 1.55\text{m}\,\text{s}^{-1}$



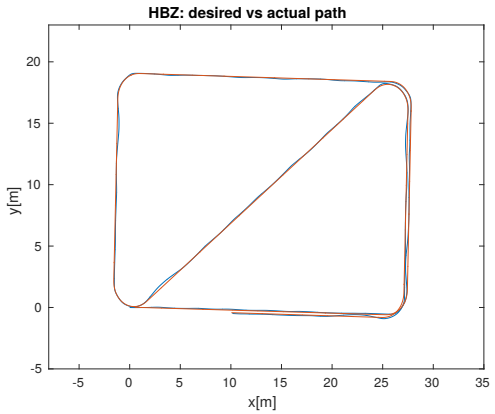
(b) *SLPINL at* $v_{mean} = 1.55\text{m}\,\text{s}^{-1}$



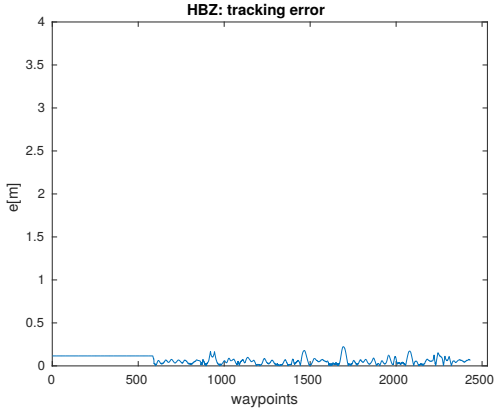
(c) *PBR at* $v_{mean} = 1.94\text{m}\,\text{s}^{-1}$



(d) *PBR at* $v_{mean} = 1.94\text{m}\,\text{s}^{-1}$



(e) *HBZ at* $v_{mean} = 2.15\text{m}\,\text{s}^{-1}$



(f) *HBZ at* $v_{mean} = 2.15\text{m}\,\text{s}^{-1}$

Figure 4.9: *Path following performance with Robotnik Summit XL on grass, following a* 159.83m *long path. The desired speed was chosen as the maximum speed before SLPINL and PBR become unstable. In the case of HBZ, the desired speed is the maximum speed of the robot.*

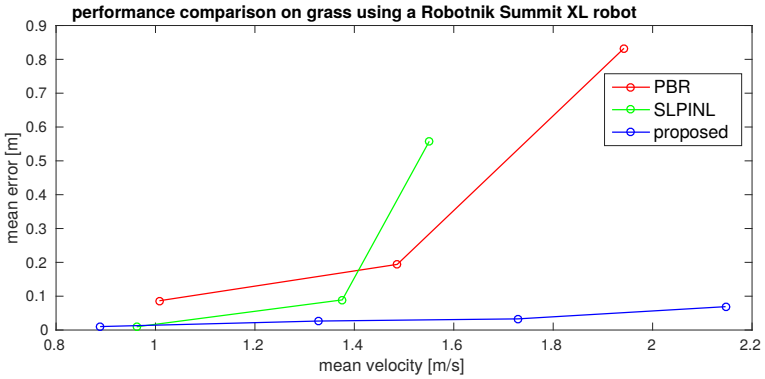


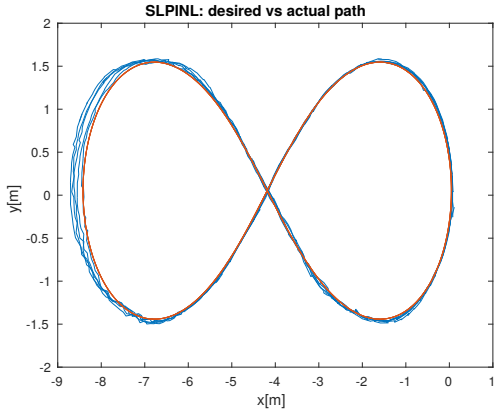
Figure 4.10: *Performance comparison on grass using the Robotnik Summit XL robot.*

robot, in order to make the path following task more challenging.

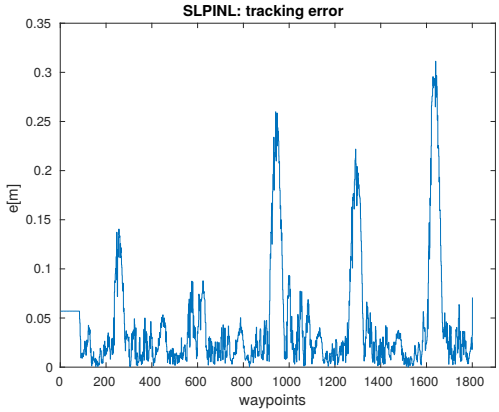
This terrain type is typical for indoor environments, having a very smooth surface, which makes the motion of the vehicle prone to slipping. For the ground truth in this case, three special pillars were used as landmarks for localization. The pillars were easily recognized with a 3D laser scanner mounted on top of the robot, and a Kalman filter was used to estimate their positions, if some of them were not seen in the current time step. Otherwise, a simple triangulation was used to determine the robot's pose.

Some results at representative speeds can be seen in Figure 4.11 and Figure 4.12. The results in Figure 4.11 show the performance of SLPINL and PBR, when commanded with a desired speed of $1\,\mathrm{m\,s^{-1}}$. In both of the cases, the error reaches the value of ≈ 31cm. The HBZ controller is commanded with $2\,\mathrm{m\,s^{-1}}$, having the average speed of $1.34\,\mathrm{m\,s^{-1}}$, and the maximum error stays below 10cm. In Figure 4.12 the results are seen, when SLPINL and PBR are commanded with $1.5\,\mathrm{m\,s^{-1}}$. In both cases, the error reaches values above 1m. The experiments were conducted in a closed room, and hence had to be stopped, in order to avoid colliding with walls. The HBZ controller was commanded with $2.5\,\mathrm{m\,s^{-1}}$, reaching speeds of $2.3\,\mathrm{m\,s^{-1}}$, and having an average speed of $1.64\,\mathrm{m\,s^{-1}}$. The maximum error in this case reaches 46.6cm, and the average error 8.5cm.

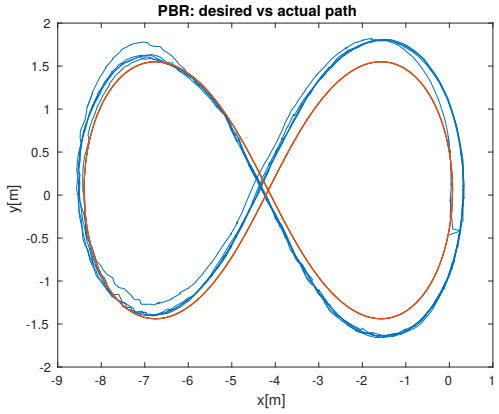
Macadam This terrain type was particularly difficult, since the surface made of crushed stones and dust would move as the robot would drive over it, which would then increase the effect of skidding. The ground truth was computed in the same way as for the vinyl terrain type, and the reference path was the same, as well. Path following results at representative speeds, together with the error profiles, can be observed in Figure 4.13 and Figure 4.14. In Figure 4.13e a localization error as a sudden jump of 88cm can be noticed, while driving with HBZ, but the robot does not lose track. After the localization jump, the maximum path following error remains below 14cm, and the mean error is 5.2cm. For the case of SLPINL and PBR in Figure 4.13, there were no localization jumps, but the path following error reaches 42.7cm with SLPINL, and 45.3cm with PBR. SLPINL and PBR were commanded with $1\,\mathrm{m\,s^{-1}}$, and HBZ with $1.5\,\mathrm{m\,s^{-1}}$.



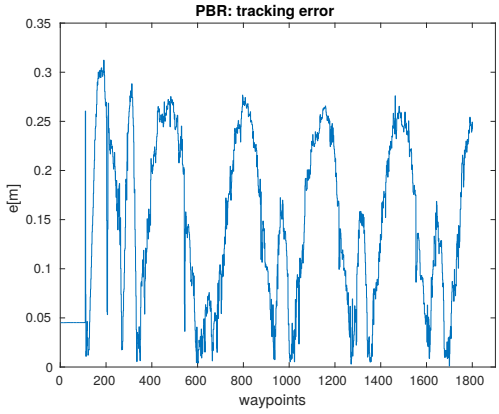
(a) *SLPINL at* $v_{mean} = 0.87\,\mathrm{m\,s^{-1}}$



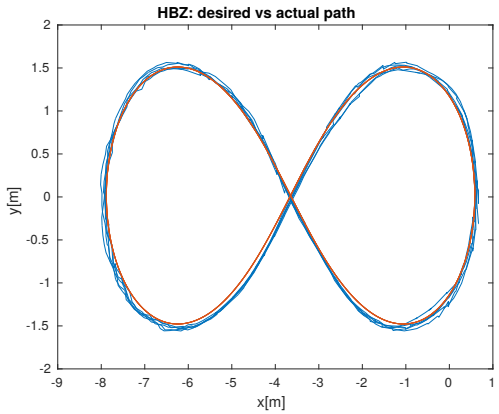
(b) *SLPINL at* $v_{mean} = 0.87\,\mathrm{m\,s^{-1}}$



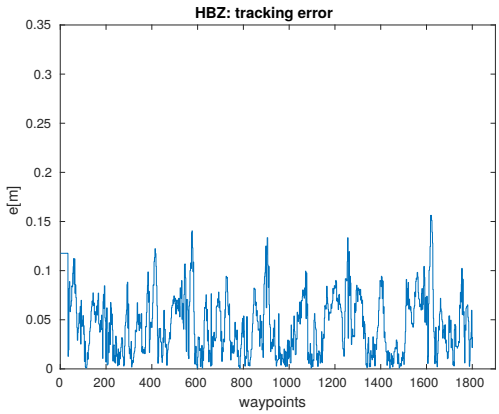
(c) *PBR at* $v_{mean} = 1.01\,\mathrm{m\,s^{-1}}$



(d) *PBR at* $v_{mean} = 1.01\,\mathrm{m\,s^{-1}}$

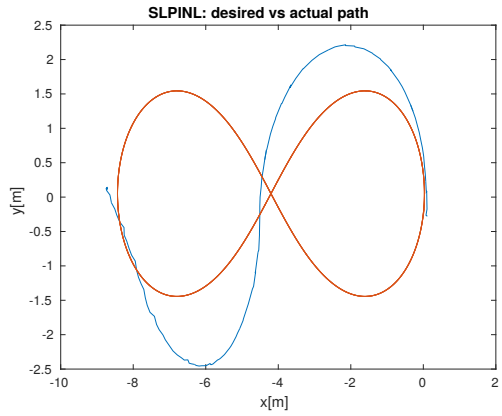


(e) *HBZ at* $v_{mean} = 1.34\,\mathrm{m\,s^{-1}}$

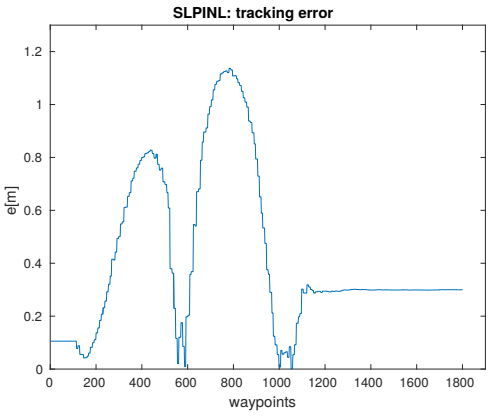


(f) *HBZ at* $v_{mean} = 1.34\,\mathrm{m\,s^{-1}}$

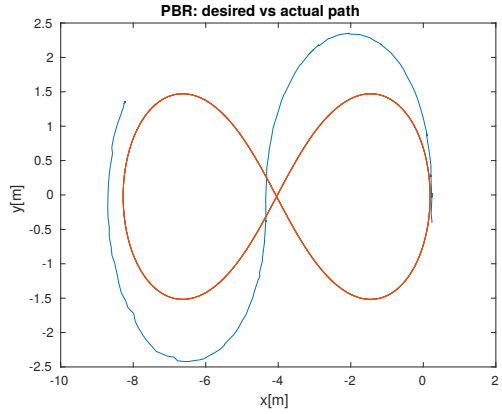
Figure 4.11: *Path following performance with Robotnik Summit XL on vinyl, following a* 110.77m *long path. The desired speed is the maximum speed at which SLPINL and PBR still perform with an acceptable deviation.*



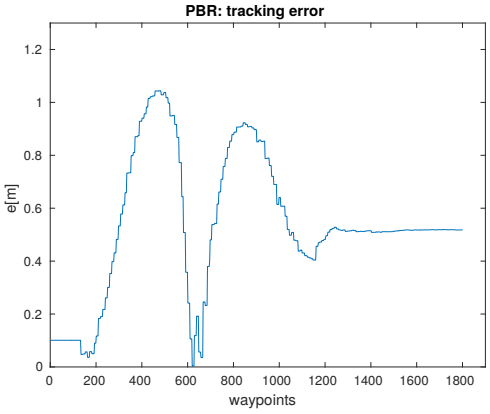
(a) *SLPINL at* $v_{mean} = 1.05\text{m}\,\text{s}^{-1}$



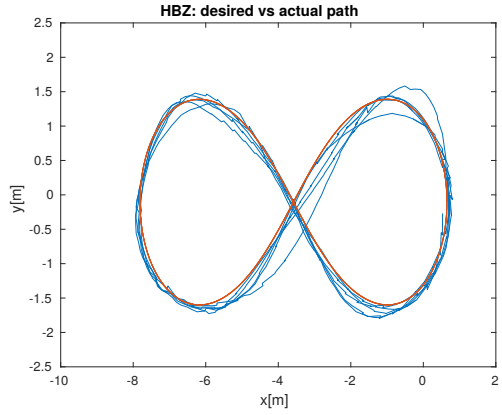
(b) *SLPINL at* $v_{mean} = 1.05\text{m}\,\text{s}^{-1}$



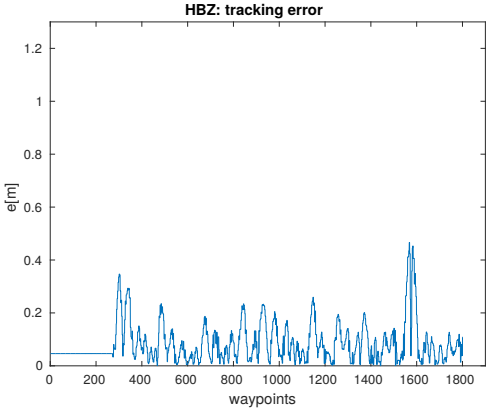
(c) *PBR at* $v_{mean} = 1.46\text{m}\,\text{s}^{-1}$



(d) *PBR at* $v_{mean} = 1.46\text{m}\,\text{s}^{-1}$

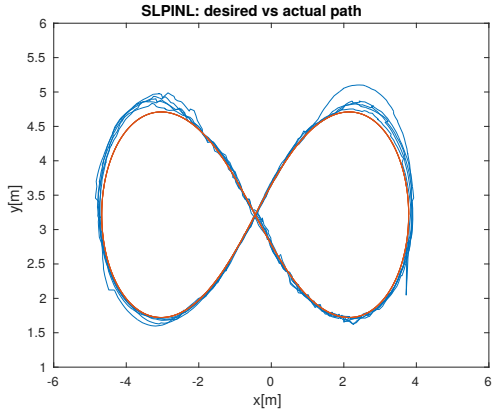


(e) *HBZ at* $v_{mean} = 1.64\text{m}\,\text{s}^{-1}$

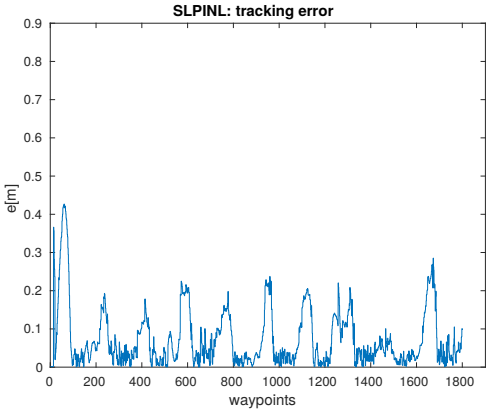


(f) *HBZ at* $v_{mean} = 1.64\text{m}\,\text{s}^{-1}$

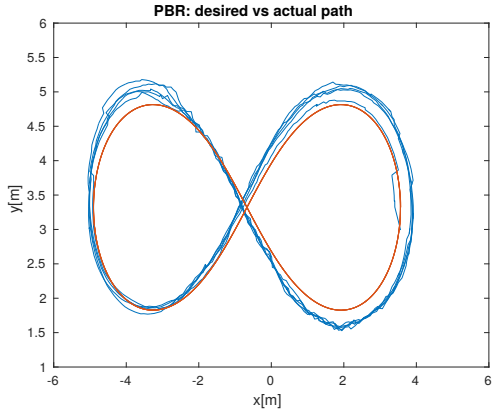
Figure 4.12: *Path following performance with Robotnik Summit XL on vinyl, following a* 110.77m *long path. The desired speed was chosen as the speed at which SLPINL and PBR have such a large deviation, that the limited testing space is violated (the robot would hit the walls, if it wasn't stopped).*



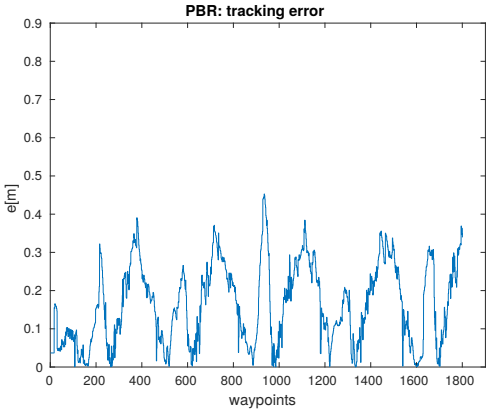
(a) *SLPINL at* $v_{mean} = 0.87\,\mathrm{m\,s^{-1}}$



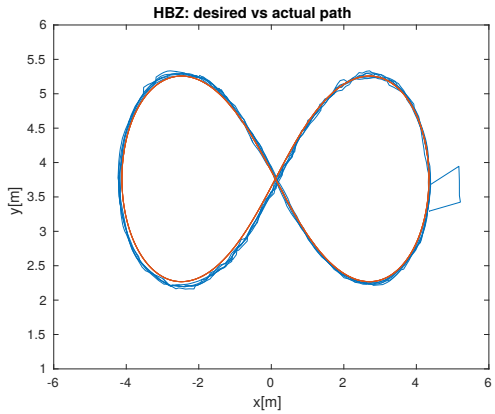
(b) *SLPINL at* $v_{mean} = 0.87\,\mathrm{m\,s^{-1}}$



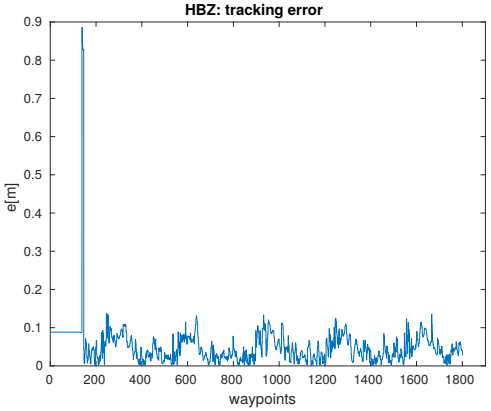
(c) *PBR at* $v_{mean} = 1.0\,\mathrm{m\,s^{-1}}$



(d) *PBR at* $v_{mean} = 1.0\,\mathrm{m\,s^{-1}}$

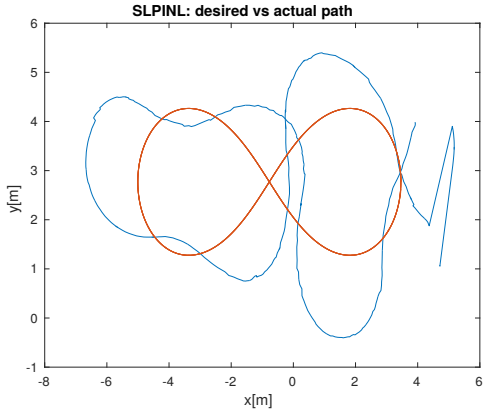


(e) *HBZ at* $v_{mean} = 1.1\,\mathrm{m\,s^{-1}}$

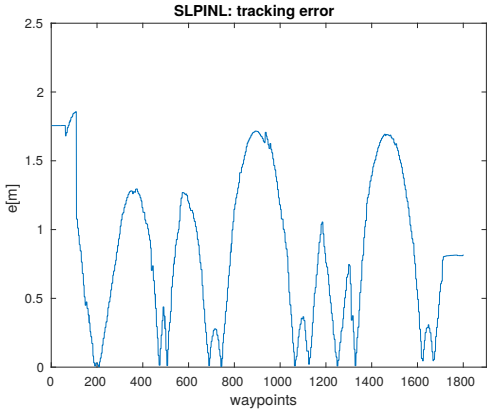


(f) *HBZ at* $v_{mean} = 1.1\,\mathrm{m\,s^{-1}}$

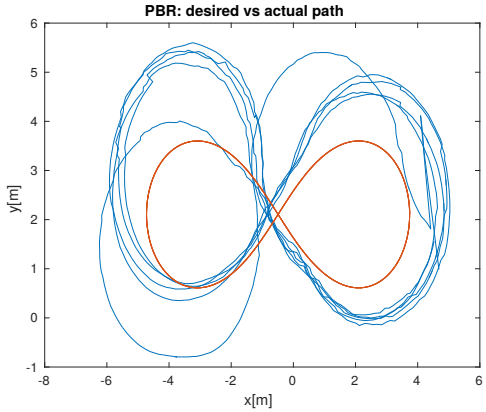
Figure 4.13: *Path following performance with Robotnik Summit XL on macadam, following a* 110.77m *long path. The desired speed is the maximum speed at which SLPINL and PBR still perform with an acceptable deviation.*



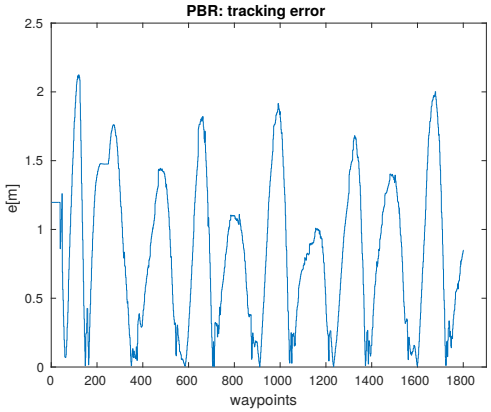
(a) *SLPINL at* $v_{mean} = 0.94\text{m}\,\text{s}^{-1}$



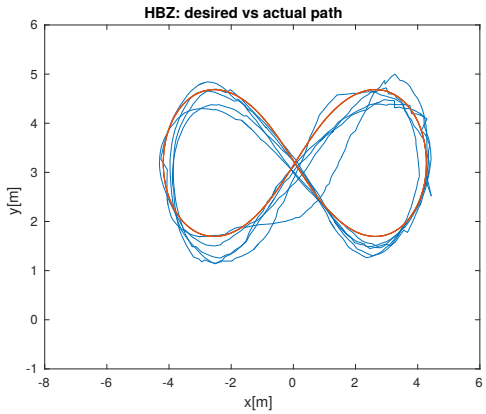
(b) *SLPINL at* $v_{mean} = 0.94\text{m}\,\text{s}^{-1}$



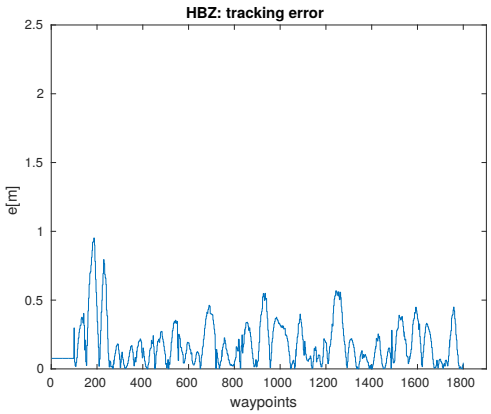
(c) *PBR at* $v_{mean} = 1.48\text{m}\,\text{s}^{-1}$



(d) *PBR at* $v_{mean} = 1.48\text{m}\,\text{s}^{-1}$



(e) *HBZ at* $v_{mean} = 1.68\text{m}\,\text{s}^{-1}$



(f) *HBZ at* $v_{mean} = 1.68\text{m}\,\text{s}^{-1}$

Figure 4.14: *Path following performance at higher speeds with Robotnik Summit XL on macadam, following a* 110.77m *long path. The desired speed was chosen as the maximum speed before SLPINL and PBR become unstable.*

Table 4.2: *Evaluation of the HBZ controller on vinyl with three different ICR parameter sets.*

Parameters	Mean Speed [$\mathrm{m\,s^{-1}}$]	Max Speed [$\mathrm{m\,s^{-1}}$]	Mean Error [cm]	Max Error [cm]
`grass`	1.45	1.96	4.9	44.8
`vinyl`	1.34	1.74	4.5	15.6
`macadam`	1.41	1.92	4.6	34.2

While driving at higher speeds, which can be observed in Figure 4.14, SLPINL loses track when commanded with $1.5\mathrm{m\,s^{-1}}$, where the mean speed is then $0.94\mathrm{m\,s^{-1}}$. The error reaches 1.86m in this case, and only one lap is performed, instead of five. The PBR controller was also commanded with $1.5\mathrm{m\,s^{-1}}$, having the mean speed of $1.48\mathrm{m\,s^{-1}}$. The maximum error is 2.12m, and the mean error 85.6cm. The HBZ controller was commanded with $2.5\mathrm{m\,s^{-1}}$, and the mean speed was $1.68\mathrm{m\,s^{-1}}$. The maximum error is 95.1cm, and the mean error 18.3cm.

Parameters In all of the presented experiments conducted with a Robotnik Summit XL robot, the parameters for the HBZ controller were as follows: $\gamma = 8$, $\zeta = 40$, $\sigma = 1$. The ICR parameters identified for grass were

$$(x_{ICR}, y_{ICRl}, y_{ICRr}, \alpha_l, \alpha_r)_{grass} = (0.28, 0.39, -0.49, 0.9, 0.91). \tag{4.59}$$

The ICR parameters for vinyl were

$$(x_{ICR}, y_{ICRl}, y_{ICRr}, \alpha_l, \alpha_r)_{vinyl} = (0.26, 0.49, -0.35, 0.8, 0.83). \tag{4.60}$$

The ICR parameters for macadam were identified as

$$(x_{ICR}, y_{ICRl}, y_{ICRr}, \alpha_l, \alpha_r)_{macadam} = (0.22, 0.48, -0.47, 0.88, 0.9). \tag{4.61}$$

For every experiment, the appropriate parameter set was used, depending on the terrain. However, it was reasonable to examine the influence of the ICR parameters to the path following performance. In order to do so, an additional experiment on vinyl was made, where the HBZ controller was commanded with $2\mathrm{m\,s^{-1}}$ to follow the five time overlaid lemniscate, like before. This time, three different runs were made, every time with different ICR parameters. The results can be seen in Table 4.2.

By observing Table 4.2 it is obvious that the maximum error is significantly smaller when vinyl parameters are used, but the mean error in all three cases is less than 5cm. This leads to the conclusion that the HBZ controller is robust to changes of the ICR parameters. The reason for this lies in the fact that the ICR parameters, being estimated offline, are a good approximate guess of the kinematics, and implicitly of the dynamics. The fast feedback control then drives the system in order to minimize the error, even

if the ICR parameters are not optimally chosen. Furthermore, the most influential ICR parameter for the feedback control is x_{ICR}, which in this example has similar values for all three runs, and the control parameters γ, ζ, and σ are kept the same.

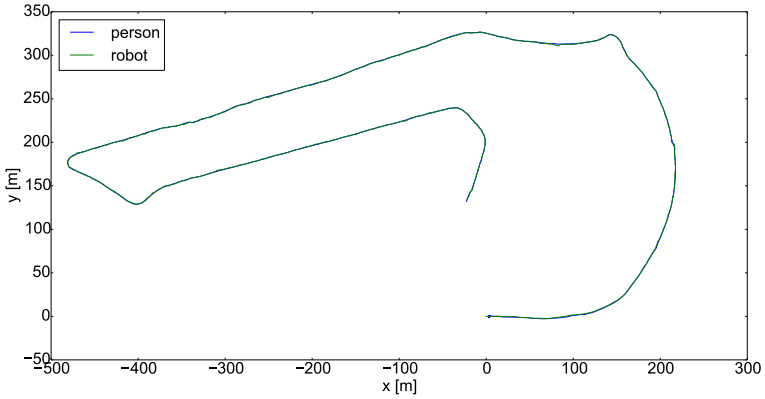


Figure 4.15: *An extension of the HBZ controller tested on a* 1791.6m *long path, over various terrain types, using only one parameter set.*

In (Huskić *et al.*, 2017c), an extension of the HBZ controller was evaluated using a Robotnik Summit XL robot for person following. The feedback control, however, is the same as the one presented in this chapter. Many long-run tests on various different terrain types and under many different conditions, such as on ice and snow, were performed to further evaluate the approach. In all of those experiments, the grass parameters shown in Eq. (4.59) were used, together with the previously mentioned controller parameters: $\gamma = 8$, $\zeta = 40$, $\sigma = 1$. This further emphasizes the robustness of the approach. In Figure 4.15 an example run of 1791.6m can be seen, using the same parameter set on many different terrain types, including asphalt, grass, gravel, dirt, etc. This is described in Chapter 5 in detail.

4.9.2 Segway RMP 440

Segway RMP 440 is a robust and relatively large skid-steered robot, weighing around 150kg and reaching speeds of $\approx 8\,\mathrm{m\,s^{-1}}$. The robot used for the experiments presented in the following text was equipped with an Intel Core i7 610E 2.53GHz processor and 8 GB of RAM. The sensor suite included wheel encoders and an internal fiber optic gyroscope. The robot can be seen in Figure 4.16.

The experiments were conducted on three different terrain types:

1. relatively flat mixture of grass and dirt;
2. very uneven terrain with relatively high grass;

3. asphalt partially covered with crushed stones.

On the first terrain type, a detailed comparison between SLPINL, PBR and HBZ is made at speeds up to $2\,\mathrm{m\,s^{-1}}$. On the second terrain type, the HBZ controller is thoroughly tested at speeds up to $\approx 6\,\mathrm{m\,s^{-1}}$. On the third terrain type, some experiments with the HBZ controller were made at moderate speeds. In all of the experiments conducted with the Segway RMP 440 robot, the following parameter set was used:

$$(x_{ICR}, y_{ICRl}, y_{ICRr}, \alpha_l, \alpha_r)_{RMP440} = (0.6, 0.74, -0.7, 0.96, 0.94). \tag{4.62}$$

The controller parameters γ, ζ, and σ were varying, but the values were similar to the ones used for the Robotnik Summit XL robot. The ground truth for all the experiments with the Segway RMP 440 robot was determined by fusing the odometry measurements with an internal gyroscope. A video demonstrating the conducted experiments can be found at: https://youtu.be/SCv0bu-Zuq8.

Grass/Dirt On this terrain type, a comparison between SLPINL, PBR and HBZ is made on two different reference paths: a three times overlaid oval-like path, and a three times overlaid lemniscate. The terrain was relatively flat, with some parts covered with short grass, and some parts made of dirt mixed with gravel. Some results gained on an oval reference path, including the desired and the actual path, as well as the error profiles, can be observed in Figure 4.17 and Figure 4.18. A direct comparison between the three algorithms at different speeds can be seen in Figure 4.19.

As it can be observed in Figure 4.17, Figure 4.18, and Figure 4.19, the SLPINL controller is very accurate at speeds lower than $1\,\mathrm{m\,s^{-1}}$, while the accuracy decreases, as the speed increases above $1\,\mathrm{m\,s^{-1}}$. At $1.53\,\mathrm{m\,s^{-1}}$, the error reaches the maximum of 1.3m, while the mean error is 38.4cm. The PBR controller is fairly accurate at lower speeds, and the error increases relatively slowly, as the speed increases. However, the error increases faster, when the speed is higher than $1.5\,\mathrm{m\,s^{-1}}$. When driving with a mean speed of $1.49\,\mathrm{m\,s^{-1}}$, the mean error is 17.6cm. At $1.98\,\mathrm{m\,s^{-1}}$, the error reaches its maximum of 1.29m, while the mean error is 39.4cm. The HBZ controller has a mean error of 7.3cm, when driving at $1.6\,\mathrm{m\,s^{-1}}$. When driving with a mean speed of $2\,\mathrm{m\,s^{-1}}$, the mean error is 13.1cm, and the maximum error 50cm.

Further experiments on the same terrain were conducted by using a lemniscate reference path overlaid three times, such that one run includes three laps around the path. Some results are shown in Figure 4.20 and Figure 4.21. A direct comparison of SLPINL, PBR and HBZ on a lemniscate at different speeds can be observed in Figure 4.22.

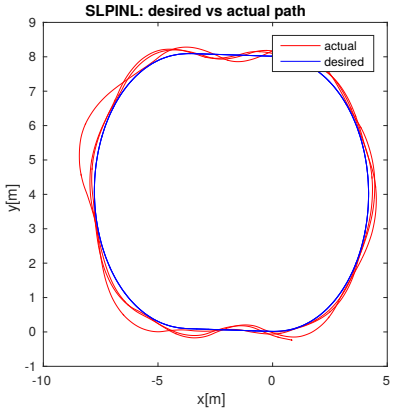
By observing Figure 4.20, Figure 4.21, and Figure 4.22, similar conclusions as in the case of the oval-shaped path can be made. At speeds lower than $1\,\mathrm{m\,s^{-1}}$, the SLPINL controller has a higher accuracy than the PBR controller. As the speed increases, the error quickly increases as well. At $1.32\,\mathrm{m\,s^{-1}}$, the maximum error is 81.7cm, and the mean error 28.4cm. When driving with the mean speed of $1.36\,\mathrm{m\,s^{-1}}$, the maximum error

Figure 4.16: *The Segway RMP 440 robot used in the experiments.*

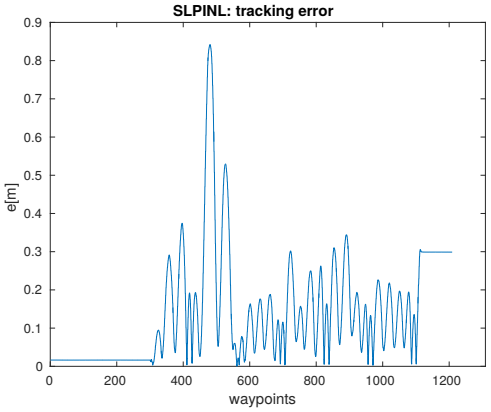
reaches 1.1m, and the mean error is 42.1cm. The reason for such a big error increase at similar mean speeds, is that the desired speeds are different.

In the first case, the commanded speed is $1.5\,\mathrm{m\,s^{-1}}$, and in the second case $1.75\,\mathrm{m\,s^{-1}}$. Because of the inherent speed control of the SLPINL algorithm, the actual speed never reaches the commanded input. Obviously, for this path, the speed of $\approx 1.3\,\mathrm{m\,s^{-1}}$ is the maximum at which this algorithm can perform, since the value is not crossed, even if the commanded speed increases. The PBR controller has a fair accuracy at lower speeds, but when the speed is increased, the error increases slower than in the case of the SLPINL controller. When driving at $1.74\,\mathrm{m\,s^{-1}}$, the error reaches the maximum of 1.3m, while the mean error is 45.9cm. The HBZ algorithm has a very high accuracy through the whole spectrum of speeds. When driving with a mean speed of $1.6\,\mathrm{m\,s^{-1}}$, the mean error is 7.9cm, while at $2\,\mathrm{m\,s^{-1}}$, the mean error increases to 16.3cm.

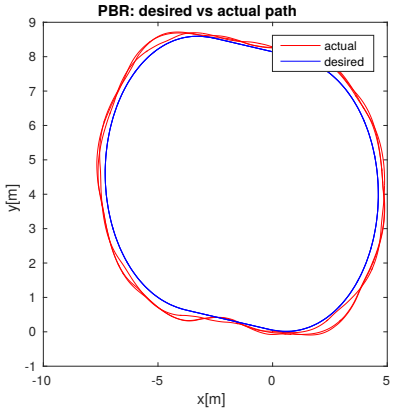
In order to test the robustness to parameter changes, an additional experiment was conducted on this terrain by following the oval path, in a similar way as before. This time, all the parameters were the same as for the Robotnik Summit XL robot on grass, as described in Section 4.9.1. The desired and the actual path, together with the error profile, can be seen in Figure 4.23. When driving with a mean speed of $0.81\,\mathrm{m\,s^{-1}}$, the maximum error is 19.3cm, and the mean error 5.9cm. The results are similar to the ones



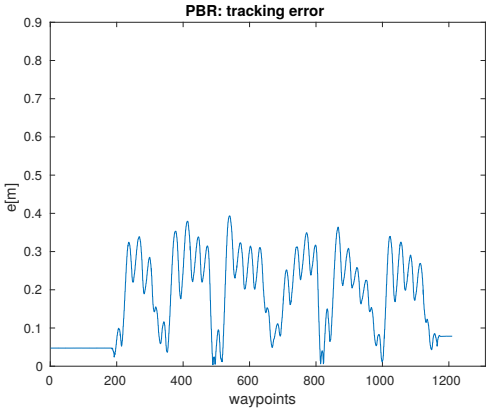
(a) *SLPINL at* $v_{mean} = 1.35\,\mathrm{m\,s^{-1}}$



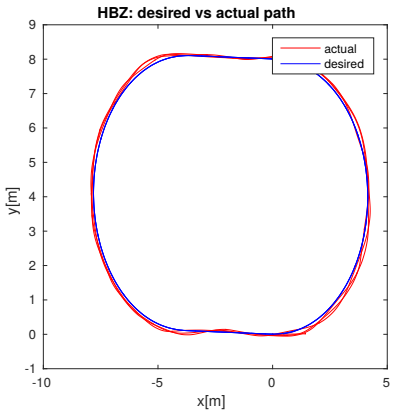
(b) *SLPINL at* $v_{mean} = 1.35\,\mathrm{m\,s^{-1}}$



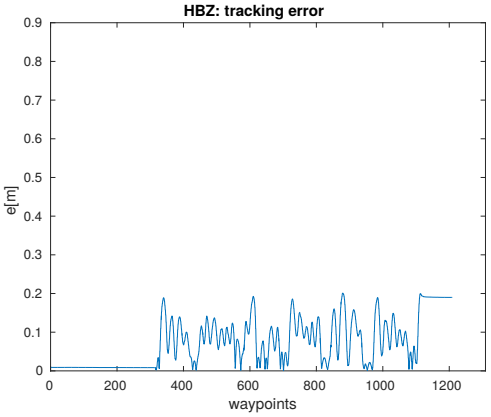
(c) *PBR at* $v_{mean} = 1.49\,\mathrm{m\,s^{-1}}$



(d) *PBR at* $v_{mean} = 1.49\,\mathrm{m\,s^{-1}}$

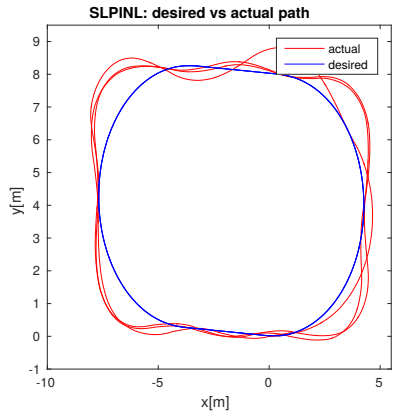


(e) *HBZ at* $v_{mean} = 1.6\,\mathrm{m\,s^{-1}}$

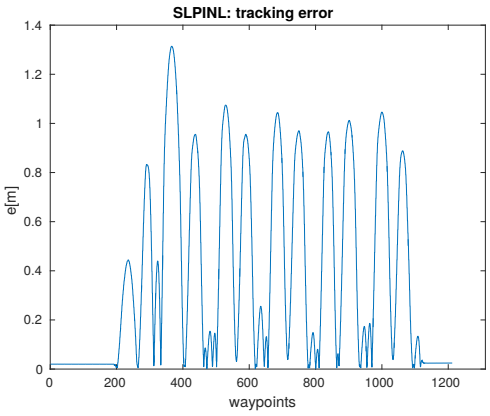


(f) *HBZ at* $v_{mean} = 1.6\,\mathrm{m\,s^{-1}}$

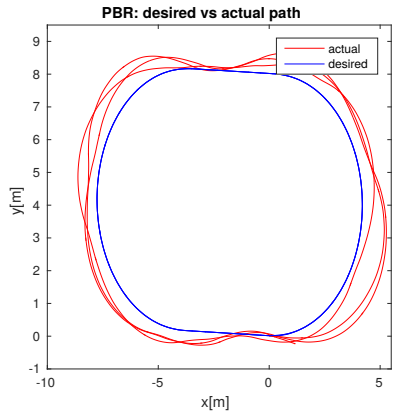
Figure 4.17: *Path following performance with Segway RMP 440 on grass/dirt, following a* 98.65m *long oval path. The desired speed is the maximum speed at which SLPINL and PBR still perform without larger deviation.*



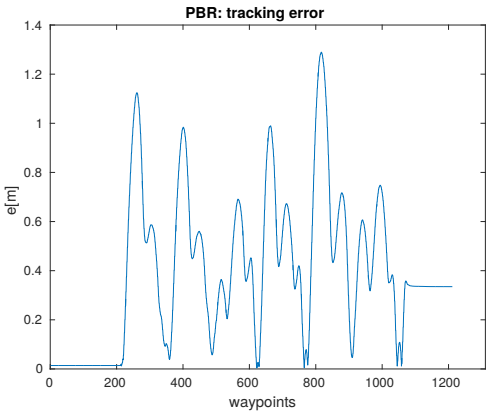
(a) *SLPINL at* $v_{mean} = 1.53\,\mathrm{m\,s^{-1}}$



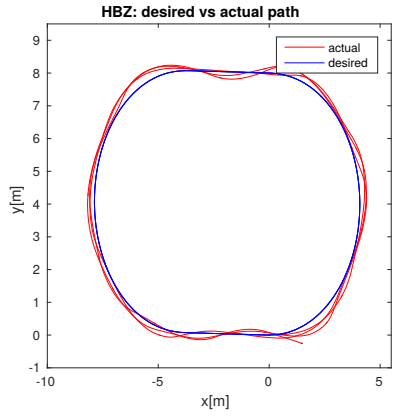
(b) *SLPINL at* $v_{mean} = 1.53\,\mathrm{m\,s^{-1}}$



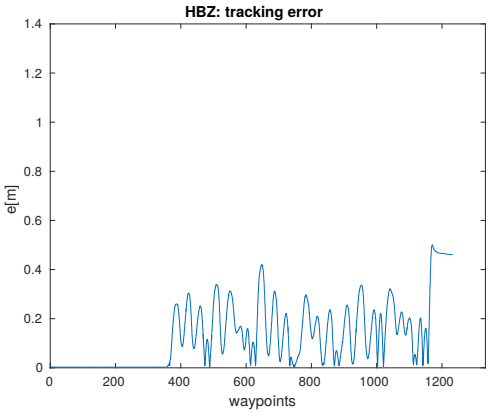
(c) *PBR at* $v_{mean} = 1.98\,\mathrm{m\,s^{-1}}$



(d) *PBR at* $v_{mean} = 1.98\,\mathrm{m\,s^{-1}}$



(e) *HBZ at* $v_{mean} = 2\,\mathrm{m\,s^{-1}}$



(f) *HBZ at* $v_{mean} = 2\,\mathrm{m\,s^{-1}}$

Figure 4.18: *Path following performance with Segway RMP 440 on grass/dirt, following a* 98.65m *long oval path. The desired speed is the speed at which SLPINL and PBR have the maximum deviation allowed by the limited green surface next to the parking lot at LAAS. If a deviation would be larger, the robot would collide with the parked cars, or the neighbouring fence.*

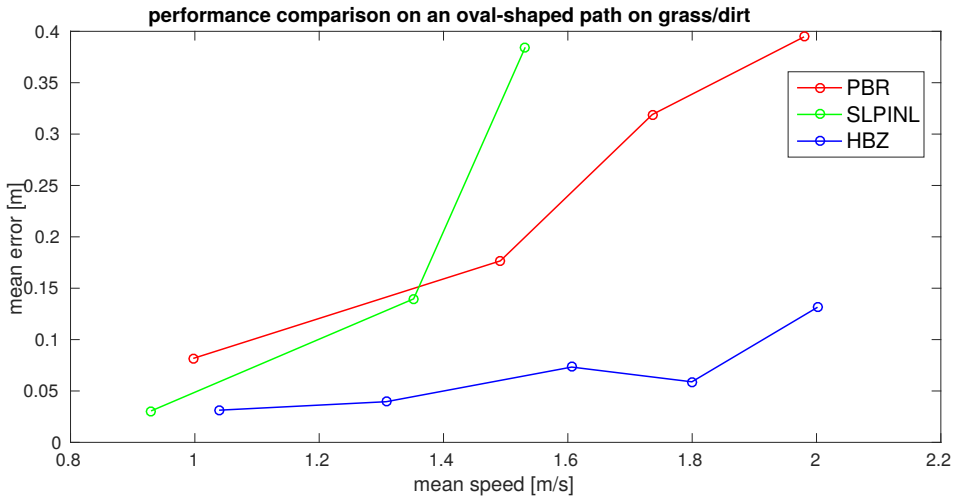


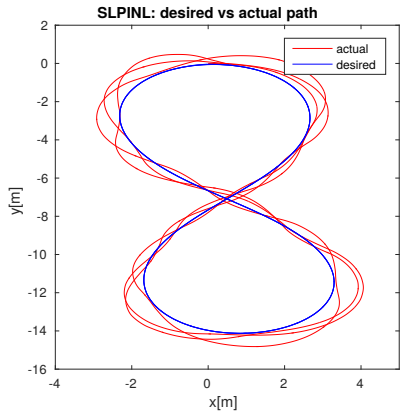
Figure 4.19: *Performance comparison on an oval-shaped path on grass/dirt using the Segway RMP 440 robot.*

gained with HBZ at 1.6m s^{-1}, with the proper ICR parameter set. The fact that the HBZ algorithm was in this case transferred from one vehicle to another very different one, without any parameter tuning, and having a relatively small tracking error, demonstrates the robustness of the approach.

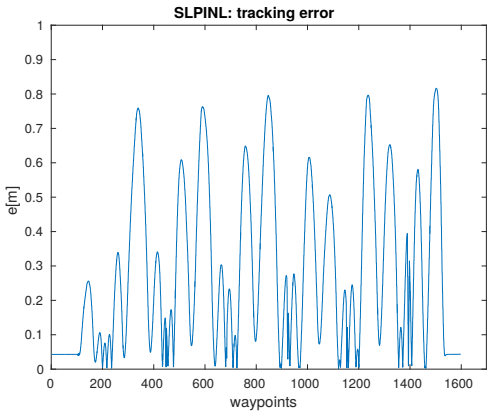
Uneven ground Further experiments were conducted at higher speeds on a very uneven ground with relatively high grass. A lemniscate path was used as a reference again, this time, however, having a larger scale. The path was partially covering a relatively deep ditch in the ground, which made the experiments even more challenging. In these experiments, only the HBZ algorithm is evaluated, since all the previous experiments showed that SLPINL and PBR become rather unstable at $\approx$ 2m s^{-1}.

In Figure 4.24 the desired and the actual paths at some representative speeds are shown, together with the error profiles. In Figure 4.25 the performance of the HBZ algorithm at different speeds can be observed. When driving with a mean speed of 2.25m s^{-1}, the maximum error is 38.96cm, while the mean error is 9.8cm. If the mean speed increases to 3.86m s^{-1}, the maximum error is 1m, and the mean error 25.4cm. The main reason for the increased error at high speeds is the relatively deep ditch, which was located in the lower right part of the lemniscate. This can be observed in Figure 4.24e, since there are some visible oscillations at this spot.

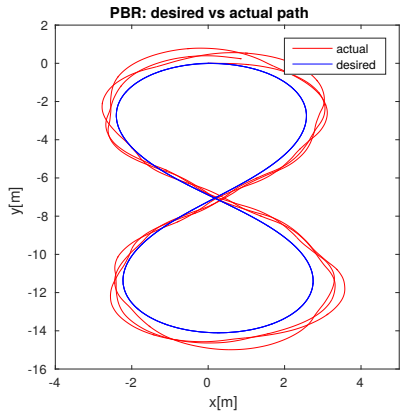
Since the terrain was very difficult to cope with, the speed of 3.86m s^{-1} was the highest mean speed achieved. Driving faster on a reference path of this scale, and on this kind of a terrain, would probably lead to tipping over. In order to drive faster while staying safe as well, the terrain should be relatively flat, and the reference path should have a bigger radius of curvature. However, the experiments described here are more challenging,



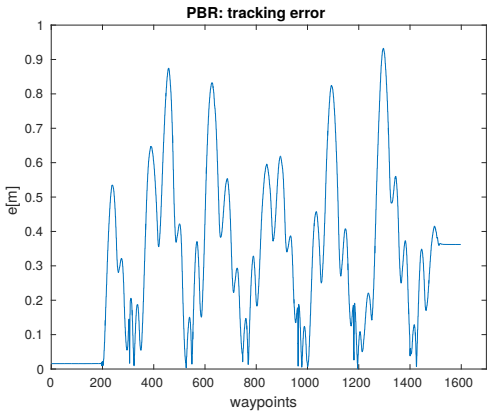
(a) *SLPINL at* $v_{mean} = 1.32\,\mathrm{m\,s^{-1}}$



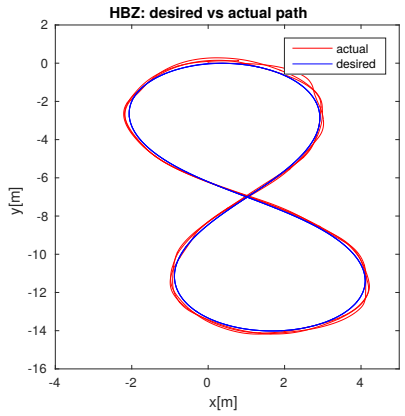
(b) *SLPINL at* $v_{mean} = 1.32\,\mathrm{m\,s^{-1}}$



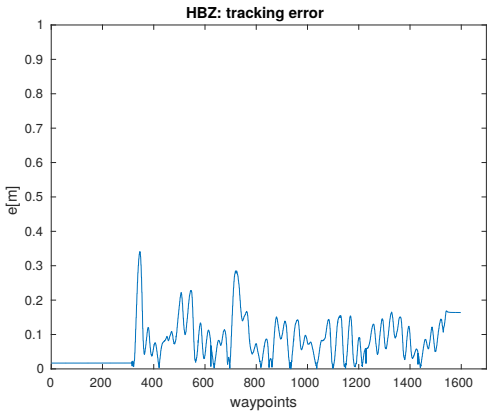
(c) *PBR at* $v_{mean} = 1.49\,\mathrm{m\,s^{-1}}$



(d) *PBR at* $v_{mean} = 1.49\,\mathrm{m\,s^{-1}}$

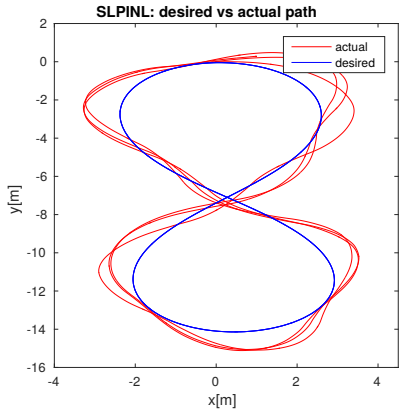


(e) *HBZ at* $v_{mean} = 1.6\,\mathrm{m\,s^{-1}}$

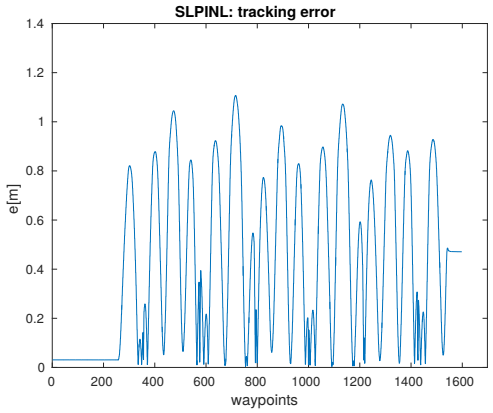


(f) *HBZ at* $v_{mean} = 1.6\,\mathrm{m\,s^{-1}}$

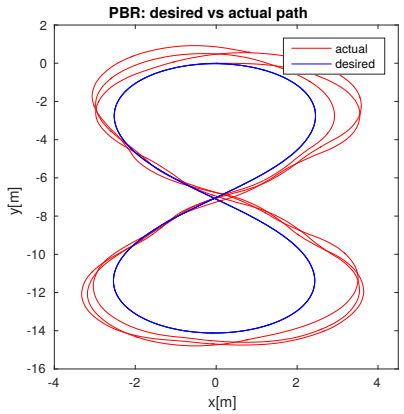
Figure 4.20: *Path following performance with Segway RMP 440 on grass/dirt, following a* 98.65m *long lemniscate. The desired speed is the speed at which SLPINL and PBR still have an acceptable deviation.*



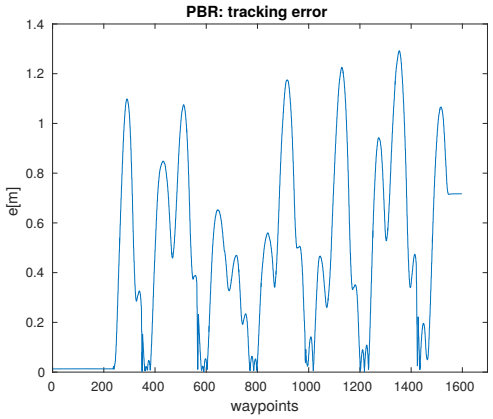
(a) *SLPINL at* $v_{mean} = 1.36\,\mathrm{m\,s^{-1}}$



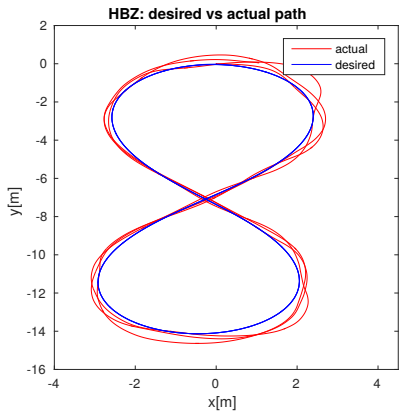
(b) *SLPINL at* $v_{mean} = 1.36\,\mathrm{m\,s^{-1}}$



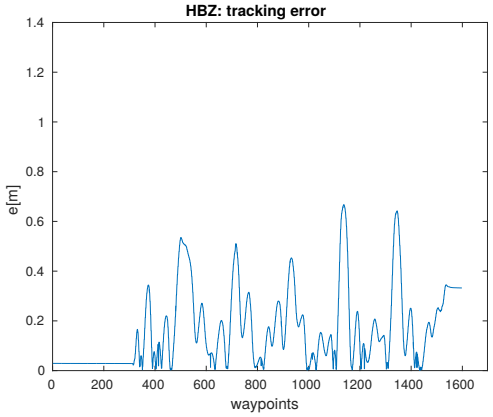
(c) *PBR at* $v_{mean} = 1.74\,\mathrm{m\,s^{-1}}$



(d) *PBR at* $v_{mean} = 1.74\,\mathrm{m\,s^{-1}}$



(e) *HBZ at* $v_{mean} = 1.99\,\mathrm{m\,s^{-1}}$



(f) *HBZ at* $v_{mean} = 1.99\,\mathrm{m\,s^{-1}}$

Figure 4.21: *Path following performance with Segway RMP 440 on grass/dirt, following a* 98.65m *long lemniscate. The desired speed is the speed at which SLPINL and PBR have the maximum deviation allowed by the limited green surface next to the parking lot at LAAS. If a deviation would be larger, the robot would collide with the parked cars, or the neighbouring fence.*

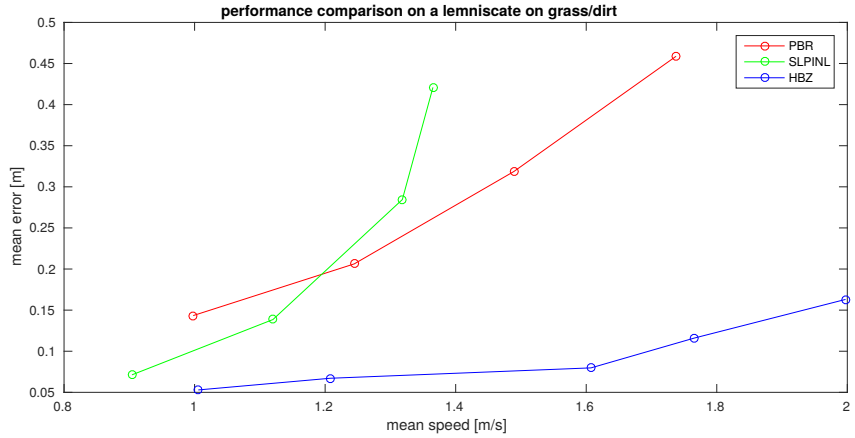
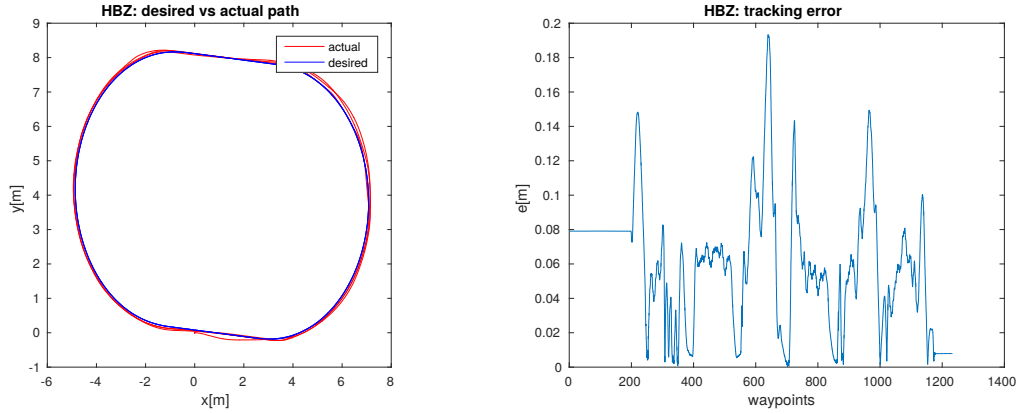


Figure 4.22: *Performance comparison on a lemniscate on grass/dirt using the Segway RMP 440 robot.*



(a) *HBZ at* $v_{mean} = 0.81\,\mathrm{m\,s^{-1}}$ (b) *HBZ at* $v_{mean} = 0.81\,\mathrm{m\,s^{-1}}$

Figure 4.23: *Path following performance with Segway RMP 440 on an oval path on grass/dirt, using the ICR parameters from the Robotnik Summit XL robot.*

since the path is on the border of being feasible, both because of the terrain, and the curvature of the path, relative to the speed of the vehicle.

If the speed control from Eq. (4.33) is properly set, the motion shall stay safe, even if the commanded speed is higher than being feasible. If the commanded speed is $3.5\mathrm{m\,s^{-1}}$ on the reference lemniscate on this uneven terrain, the mean speed is $3.12\mathrm{m\,s^{-1}}$, since the speed control regulates the motors so that the tracking error stays as small as possible. Furthermore, if the curvature of the path increases, the speed decreases. If the commanded speed is $6\mathrm{m\,s^{-1}}$, the maximum speed will reach $6.16\mathrm{m\,s^{-1}}$ indeed, but Eq. (4.33) will reduce the overall mean speed to $3.19\mathrm{m\,s^{-1}}$. This means that the vehicle is allowed to accelerate up to the allowed maximum, if the tracking error and the curvature are small enough, otherwise, the speed is reduced to a safe value. The comparison between these two cases can be seen in Figure 4.26.

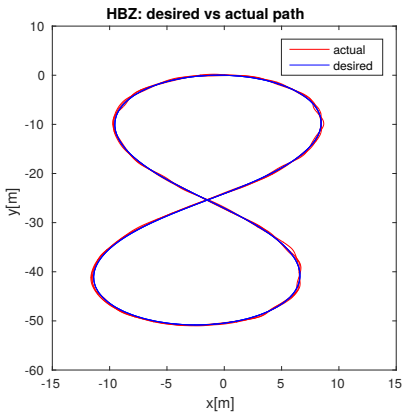
This effect can be best observed in Figure 4.26g and Figure 4.26h. When the desired speed is $3.5\mathrm{m\,s^{-1}}$, the speed profile is smooth. When commanding $6\mathrm{m\,s^{-1}}$, the peaks of $\approx 6\mathrm{m\,s^{-1}}$ show up while driving on the straight line segments of the lemniscate, and the rest of the time the speed remains around $\approx 3\mathrm{m\,s^{-1}}$. When entering a straight line segment and accelerating, the vehicle would sometimes even lose contact between the front wheels and the ground, and drive only on the rear wheels for a short period. The angular velocities are similar in both cases, and the error increases in the case of the high commanded speed, as it can be observed in Figure 4.26d.

Old asphalt Some further experiments at moderate speeds were conducted on old asphalt with smaller cracks and areas covered with crushed stones. Using a skid-steered vehicle on asphalt usually leads to unstable behaviour, since the friction between the ground and the wheels is very high. In order to skid, large forces need to be applied, which then exhibit discontinuous impulses. This can be observed as a swinging behaviour of the vehicle, which could even lead to tipping over, if e.g. the centre of mass is high. This effect is especially dangerous, if the vehicle is turning on the spot.

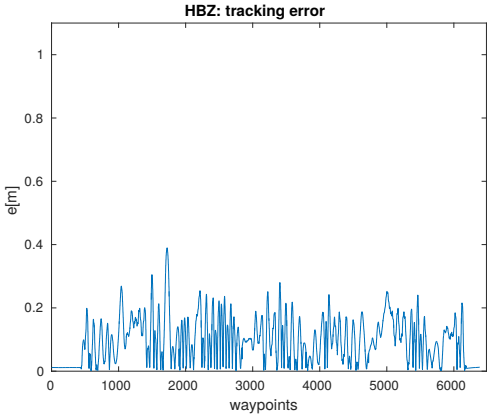
Since the asphalt used for these experiments was covered with crushed stones at some areas, it was possible for the vehicle to exhibit a relatively smooth motion. An oval-shaped path was used as a reference, similar as before. The desired and the actual path, the error profile, together with the angular and longitudinal velocities for one representative experiment can be observed in Figure 4.27. The mean speed in this case is $1.85\mathrm{m\,s^{-1}}$, the maximum error 14cm, and the mean error 6.2cm.

4.10 Conclusions

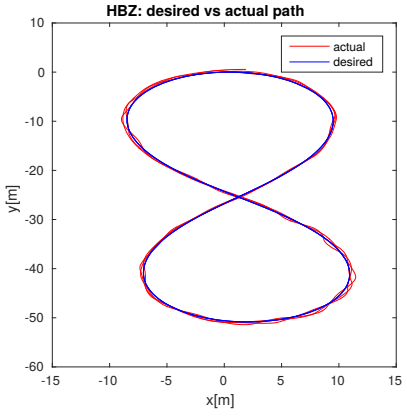
In this chapter, we propose a high speed controller for skid-steered vehicles on rough terrain, and thoroughly evaluate it on two different robots, on different terrain and path types, and compare it with two state-of-the-art algorithms through a whole spectrum of speeds.



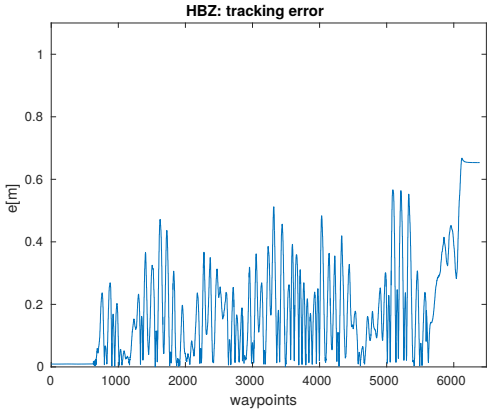
(a) *HBZ path at* $v_{mean} = 2.25\text{m s}^{-1}$



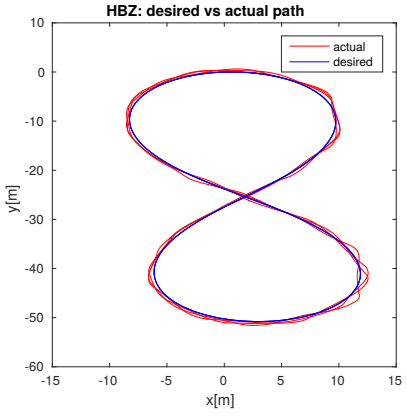
(b) *HBZ error at* $v_{mean} = 2.25\text{m s}^{-1}$



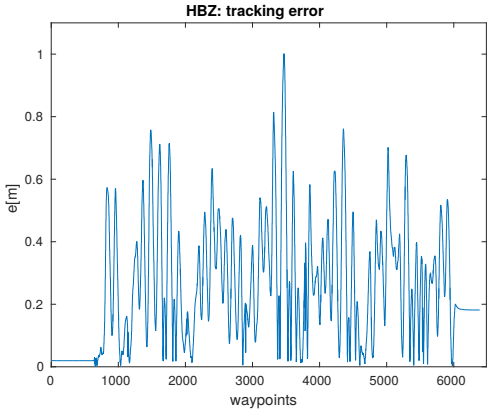
(c) *HBZ path at* $v_{mean} = 3.37\text{m s}^{-1}$



(d) *HBZ error at* $v_{mean} = 3.37\text{m s}^{-1}$



(e) *HBZ path at* $v_{mean} = 3.86\text{m s}^{-1}$



(f) *HBZ path at* $v_{mean} = 3.86\text{m s}^{-1}$

Figure 4.24: *Path following performance at high speeds with Segway RMP 440 on a very uneven terrain with high grass, following a* 400.14m *long lemniscate.*

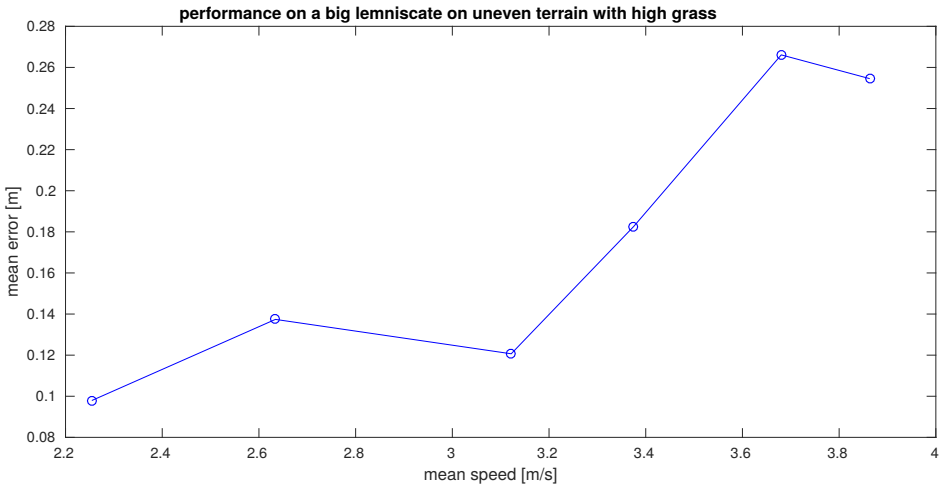


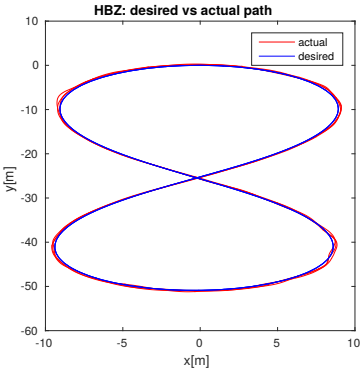
Figure 4.25: *HBZ performance on a very uneven terrain using the Segway RMP 440 robot.*

The proposed algorithm is a combined solution of a nonlinear control law designed for path following, and a control scheme for the longitudinal velocity. The path following control guarantees that the vehicle will converge to the path. The speed control maximizes the speed, while minimizing the tracking error. Both control solutions are based on an experimentally identified kinematic model for skid-steered vehicles.

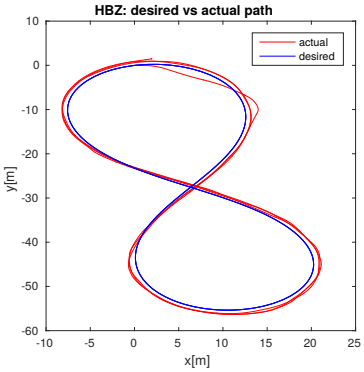
A comprehensive experimental comparison between the proposed solution and two other state-of-the-art algorithms is made, on two different vehicles and on various terrain types, by following several path types. All the terrain scenarios used in the experiments can be seen in Figure 4.28. The experiments include two major difficulties for the path following task: rough terrain and high speeds. The maximum speed reached is $6.16\mathrm{m\,s^{-1}}$.

In the experimental evaluation, it is clear that the proposed approach outperforms the two state-of-the art algorithms on both vehicles, and on each terrain and path type. The difference gets more obvious as the speed increases. The SLPINL algorithm, however, shows very good performance at lower speeds. At higher speeds, the error increases rapidly. The PBR algorithm usually has a small offset even at lower speeds. The reason lies in the fact that this approach relies heavily on the convergence of the Extended Kalman Filter (EKF), which is needed for the online estimation of the ICR parameters. The EKF does not necessarily find the optimal values, which leads to an offset. The PBR controller, however, performs better than SLPINL at higher speeds, since it takes ICR parameters into account, even though it uses a simple control law for unicycles.

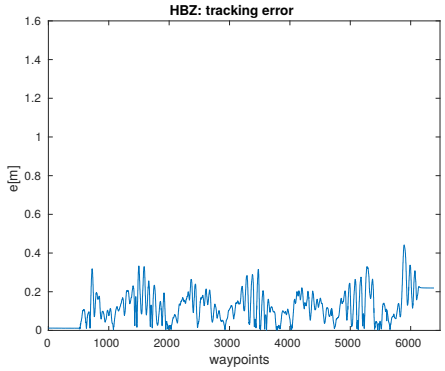
The proposed approach shows very stable performance even at high speeds and rough terrain. In the experiments it is shown that it recovers from localization errors, and that it is robust to parameter changes. The complete solution was cloned from one vehicle to another, very different one, without any parameter tuning, and still gave satisfying



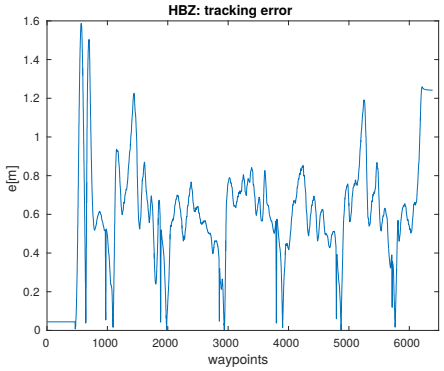
(a) *HBZ path at* $v_{mean} = 3.12\text{m}\,\text{s}^{-1}$



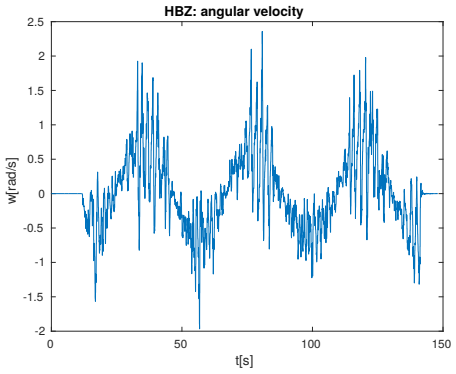
(b) *HBZ path at* $v_{mean} = 3.19\text{m}\,\text{s}^{-1}$



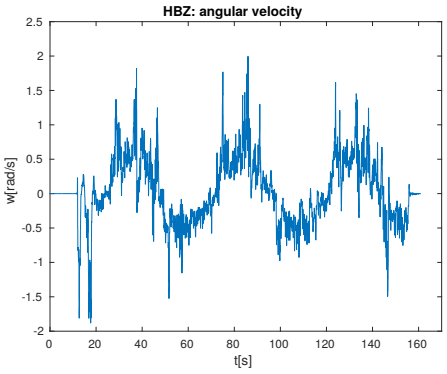
(c) *HBZ error at* $v_{mean} = 3.12\text{m}\,\text{s}^{-1}$



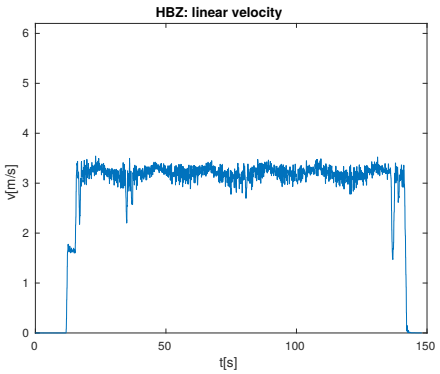
(d) *HBZ error at* $v_{mean} = 3.19\text{m}\,\text{s}^{-1}$



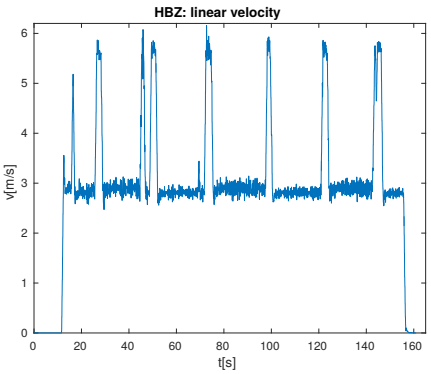
(e) *HBZ rotation at* $v_{mean} = 3.12\text{m}\,\text{s}^{-1}$



(f) *HBZ rotation at* $v_{mean} = 3.19\text{m}\,\text{s}^{-1}$

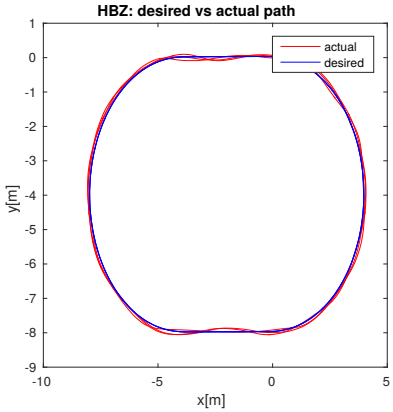


(g) *HBZ speed at* $v_{mean} = 3.12\text{m}\,\text{s}^{-1}$

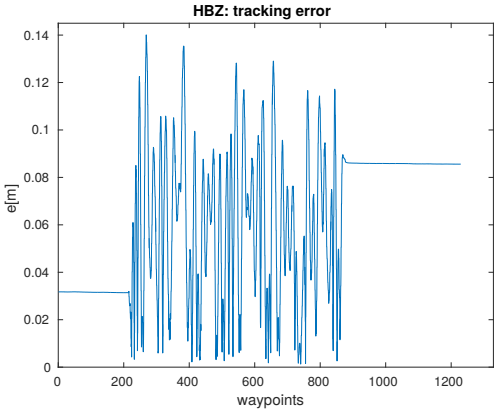


(h) *HBZ speed at* $v_{mean} = 3.19\text{m}\,\text{s}^{-1}$

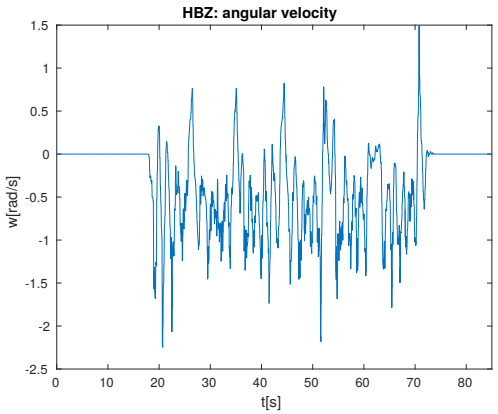
Figure 4.26: *Effects of the speed control when following a path with commanded speeds that are higher than feasible.*



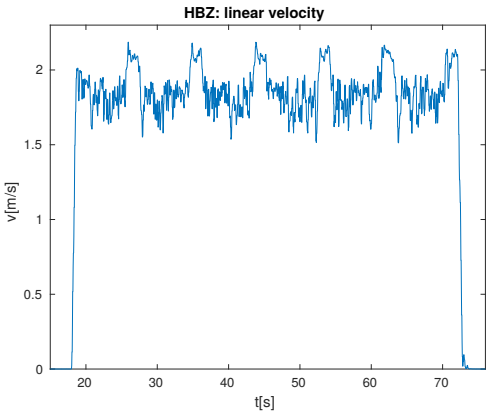
(a) *HBZ path at* $v_{mean} = 1.85\text{m}\,\text{s}^{-1}$



(b) *HBZ error at* $v_{mean} = 1.85\text{m}\,\text{s}^{-1}$



(c) *HBZ rotation at* $v_{mean} = 1.85\text{m}\,\text{s}^{-1}$



(d) *HBZ speed at* $v_{mean} = 1.85\text{m}\,\text{s}^{-1}$

Figure 4.27: *Path following performance with Segway RMP 440 on old asphalt mixed with gravel, following a* 98.65m *long oval path.*

(a) *grass*

(b) *grass/dirt*

(c) *vinyl*

(d) *uneven ground*

(e) *macadam*

(f) *old asphalt*

Figure 4.28: *All the terrain types used for the experiments. On the left hand side the terrain types on which Robotnik Summit XL was tested are shown, and on the right hand side the terrain types on which Segway RMP 440 was tested.*

results. Furthermore, in (Huskić *et al.*, 2017c) the proposed approach is extended for person following and obstacle avoidance, and tested with the same parameter set on various different terrain types, on large-scale paths.

Since the model utilized for the control is a kinematic model which takes the dynamics into account only implicitly, it would be reasonable to extend this work by explicitly modelling the dynamics and the wheel slip.

Chapter 5

Outdoor Person Following at Higher Speeds using a Skid-Steered Mobile Robot

5.1 Introduction

Fast and safe autonomous outdoor navigation of mobile robots is an open and challenging problem, with various possible applications, such as agriculture, search and rescue, exploration or inspection. Furthermore, human-robot cooperation (such as person following) in rough terrain conditions, dynamic environments, and at higher speeds makes the task even more challenging. Another difficulty is the use of skid-steered vehicles, which are relatively cheap, robust and highly maneuverable, but difficult to model and to control at higher speeds, as previously discussed.

Person following at low or moderate speeds is well explored and described in the literature. Cosgun et al. provide an approach for autonomous person following, while avoiding obstacles, for telepresence robots in (Cosgun *et al.*, 2013). In their experiments, they use a differential drive, and the speed is limited to $0.55\,\mathrm{m\,s^{-1}}$. Pucci et al. propose an interesting nonlinear control law in (Pucci *et al.*, 2013), where they follow a person using a unicycle-like robot. In the real experiments, the maximum speed reaches $\approx 1.5\,\mathrm{m\,s^{-1}}$, and there is no obstacle avoidance. Park and Kuipers introduce an approach for autonomous person pacing and following in (Park and Kuipers, 2013), where their differentially driven robot exhibits smooth and collision-free behaviour in dynamic environments, by driving at speeds up to $\approx 1.0\,\mathrm{m\,s^{-1}}$. Person following with a MetraLabs Scitos G5 robot at speeds up to $0.3\,\mathrm{m\,s^{-1}}$ is proposed by Liu et al. in (Liu *et al.*, 2014) and (Liu *et al.*, 2015). Leigh et al. propose a method for person following using 2D laser scanners in (Leigh *et al.*, 2015), using a skid-steered Clearpath Robotics Husky robot. They use a combination of two PID controllers to reactively follow the person, without avoiding obstacles on the way, and move at speeds up to $1.0\,\mathrm{m\,s^{-1}}$. Garzon Oviedo et al. introduce a person tracking and following system in (Garzon Oviedo *et al.*, 2015), where they use a Robotnik Summit XL robot and move at speeds up to $1.46\,\mathrm{m\,s^{-1}}$. The maximum speed of all these approaches is lower than the average speed in the approach

Figure 5.1: *A typical outdoor jogger following scenario using our Robotnik Summit XL mobile robot.*

we are proposing.

An interesting concept of a person following robot has been recently announced by the Italian company Piaggio (see (PiaggioFastForward, 2017)). This cargo-carrying robot is indeed intended for outdoor purposes, but limited to urban scenarios.

Lenain et al. propose an efficient control strategy for target tracking in (Lenain *et al.*, 2014), where they use a car-like robot to follow another robot driving at around $2.5\,\mathrm{m\,s^{-1}}$. These speeds are comparable to ours, but we use a skid-steered robot, which is more challenging to control, due to its terrain-dependent kinematics. Furthermore, our approach actively avoids obstacles, which seems not to be the case in (Lenain *et al.*, 2014).

Our contributions can be listed as follows:

- We propose a new path planning algorithm, which considers the vehicle's kinematics and dimensions, and provides smooth and obstacle-free paths;
- We extend our path following control law described in Chapter 4 for dynamic object (person) following;
- We experimentally evaluate the performance of our obstacle avoidance by comparing it with two classical obstacle avoidance approaches on a real outdoor robot;
- We experimentally evaluate the complete navigation system for outdoor person following in various rough terrain scenarios at higher speeds.

Our approach is also highly modular, which means that each component can be easily exchanged or extended. Instead of following a person, with minor modifications we

can follow any dynamic object, i.e. another robot, which may be very interesting for multi-robot systems.

5.2 Framework

Our navigation system consists of a global path planner, based on the A* path planning algorithm ((Hart *et al.*, 1968)), a local path planner, and a high speed controller. The global path planner provides a general direction from the robot to the person, while the local path planner provides optimized and collision-free paths in the local area around the robot. The path following controller then assures the convergence of the robot to the best local path. The person's pose is determined by using a Kalman-filter-like tracking of LIDAR data coming from a 3D laser scanner Velodyne Puck VLP-16. Our obstacle detection is based on the algorithm proposed by Buck et al. in (Buck *et al.*, 2016a). Localization is done by using a fusion of Hall effect sensors and an internal gyroscope. A block diagram of our navigation system can be seen in Figure 5.2.

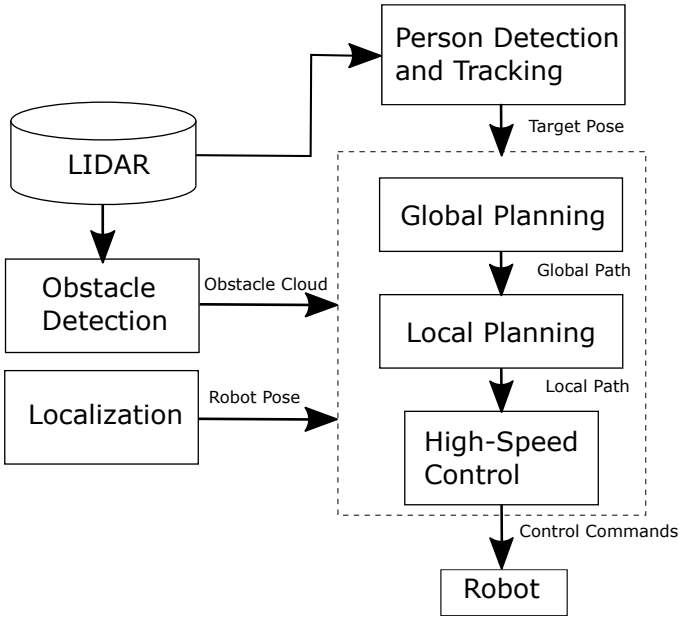


Figure 5.2: *A block diagram of our navigation system.*

5.3 Local Path Planner

While global path planning in unknown environments might take a lot of time, and driving reactively incurs the danger of getting into local minima, we propose a solution for

both problems. We use an inexpensive A* global planner for a fast global search, and a novel local planner for a refined search in a reduced search space.

Given a global path, we restrict the search space for the local planner to a customizable area around the global path, excluding all the obstacles on the way. The general idea of the local planner can be seen in Fig.5.3.

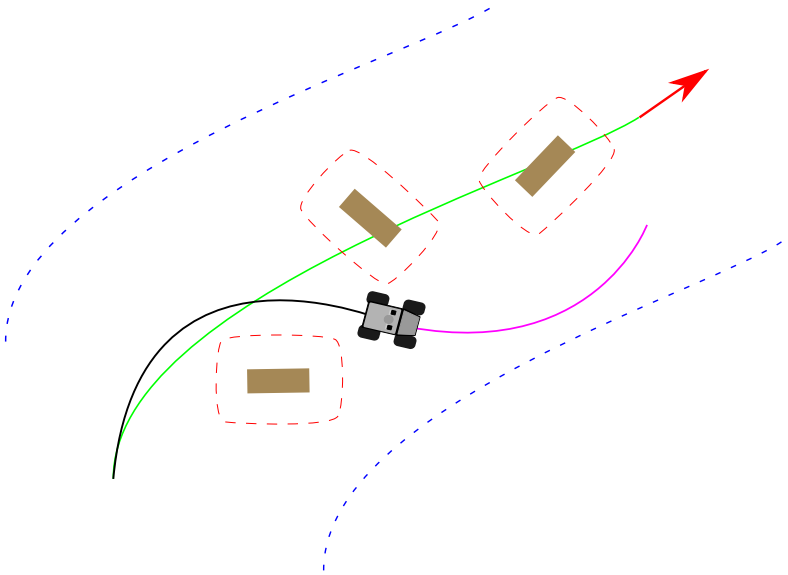


Figure 5.3: *The idea of the proposed local planner. The green full line represents the global path, the red arrow is the target pose, the black line is the driven path, and the blue line is the current local path. Yellow boxes represent obstacles, the red dashed lines the restricted areas around each obstacle, and the blue dashed line the border of the search space around the global path.*

5.3.1 Initial Search

The initial search is based on the Hybrid A* approach (described in (Dolgov *et al.*, 2008)). For the Hybrid A* approach, a node in the tree is expanded by simulating the kinematics of a car-like vehicle, applied for a small period of time, corresponding to the resolution of the grid. We also use a car-like kinematics, even though we use a skid-steered robot in our experiments. Paths generated in this way are drivable by almost every ground mobile robot (including skid-steered robots), regardless of its kinematics. This serves the modularity of our approach, since for using our navigation system on a kinematic configuration other than skid steering, one would just have to change the high speed controller (as depicted in Figure 5.2).

While Hybrid A* assumes a grid-like structure of the search space, we propose a different kind of discretization. Instead of constructing a rectangular grid in the entire search space, we construct a circle around each node, with the radius being the distance to the

neighbouring node of the same level. In this way, there is no unnecessary discretization of the entire search space. The principle can be seen in Figure 5.4.

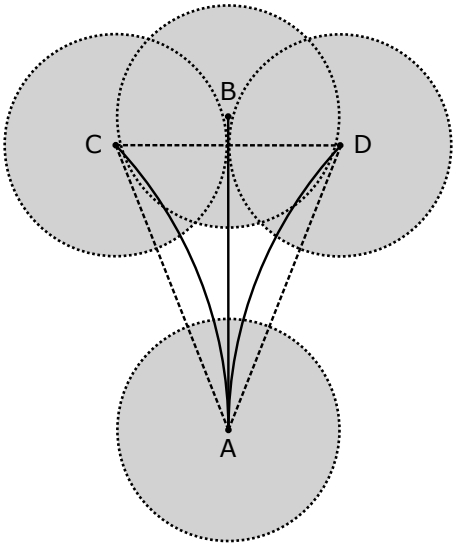


Figure 5.4: *The proposed circular discretization. Node A is the parent node for nodes B, C, and D, which are gained by using car-like kinematics. Around each node, a circle is constructed, where the radius is the distance to the neighbouring node of the same level.*

By defining the radii of the circles to equal the distance between the nodes of the same level, it is guaranteed that the discretization is dependent on the kinematics. An example of a search tree demonstrating the node expansion can be seen in Figure 5.5.

5.3.2 Scoring

For the search described in the previous section, it is necessary to define a scoring system for nodes, i.e. the costs of each node. Each node is associated with the nearest point on the global path, as depicted in Figure 5.6.

The heuristic for each node is computed as the distance in path coordinates from the nearest point on the global path to the goal of the global path. Additional costs are computed by using five different criteria:

- distance to the global path,
- distance difference between the current node and its parent,
- curvature in the current node,
- curvature difference between the current node and its parent,
- pose of the nearest obstacle.

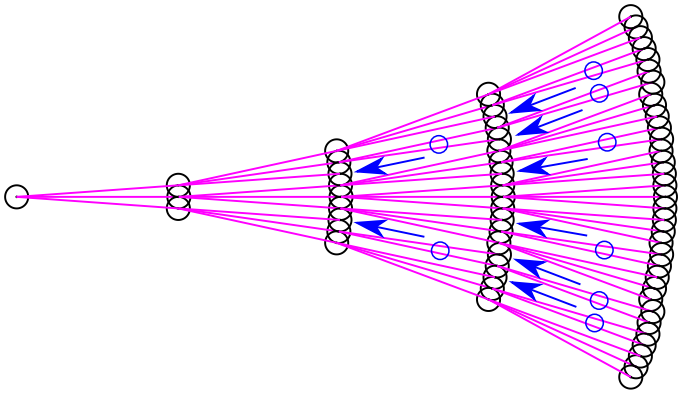


Figure 5.5: *The initial search tree. Black circles represent nodes, and magenta lines connections between them. Blue arrows indicate that some nodes cannot be expanded, and blue circles show that some nodes are only partially expanded. However, the search space is fully covered.*

The nodes far away from the global path get high costs, as well as the nodes which increase the distance to the global path, when compared to their parent node. Similarly, the nodes with higher curvature get higher costs, as well as the nodes whose curvature increases, in comparison to their parent. All of these criteria are computed as linear functions, except for the position of the nearest obstacle.

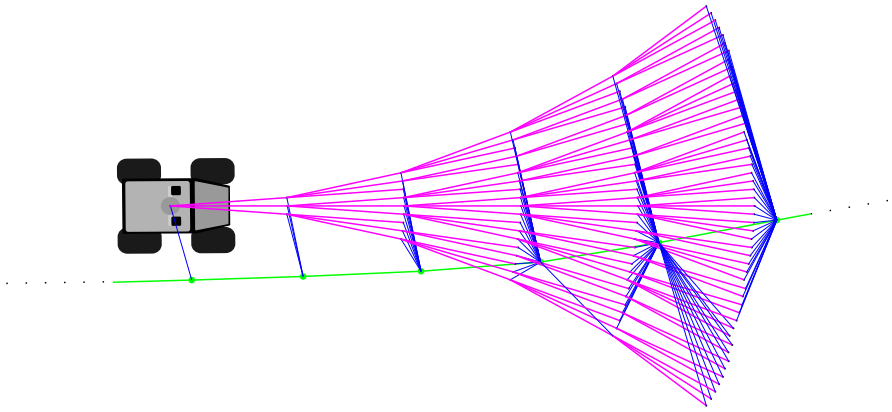


Figure 5.6: *The computation of costs. Each node is associated with the nearest point on the global path.*

The distance to the nearest obstacle is computed taking into account the dimensions of the robot, plus an additional static safety area around the robot, and a dynamic front safety area, scaled with the current speed of the robot. Cost computation using obstacle distance is done as follows. If the angle between the current node and the nearest obstacle is denoted with α, where $\alpha \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, and the angle between the current node and the

nearest point on the global path with β, then the following cost function can be defined

$$\zeta = \cos\alpha \left(1 - \left|\cos\frac{\beta}{2}\right|\right)\left(e^{\frac{k}{d_{min}}} - 1\right), \tag{5.1}$$

with k being a positive parameter, and d_{min} the distance to the nearest obstacle. This provides a realistic model of obstacle influence, allowing safe paths in dynamic and cluttered environments at higher speeds. The angles α and β are defined in Figure 5.7. The green line represents the global path, and the blue dots are its points. Two grey objects represent obstacles, and the two red dots are the nearest obstacle points. The dotted line lying on the robot's y-axis, together with the orange arrows, represents the area in which α is defined. This means that the obstacle point A will be taken into account, while B will not, since α would be outside of range in this case.

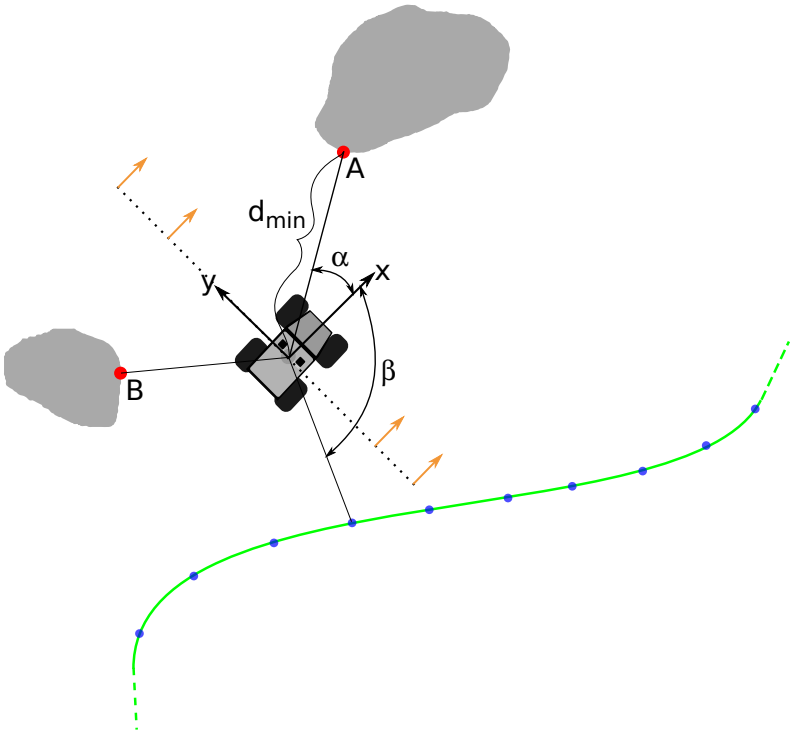


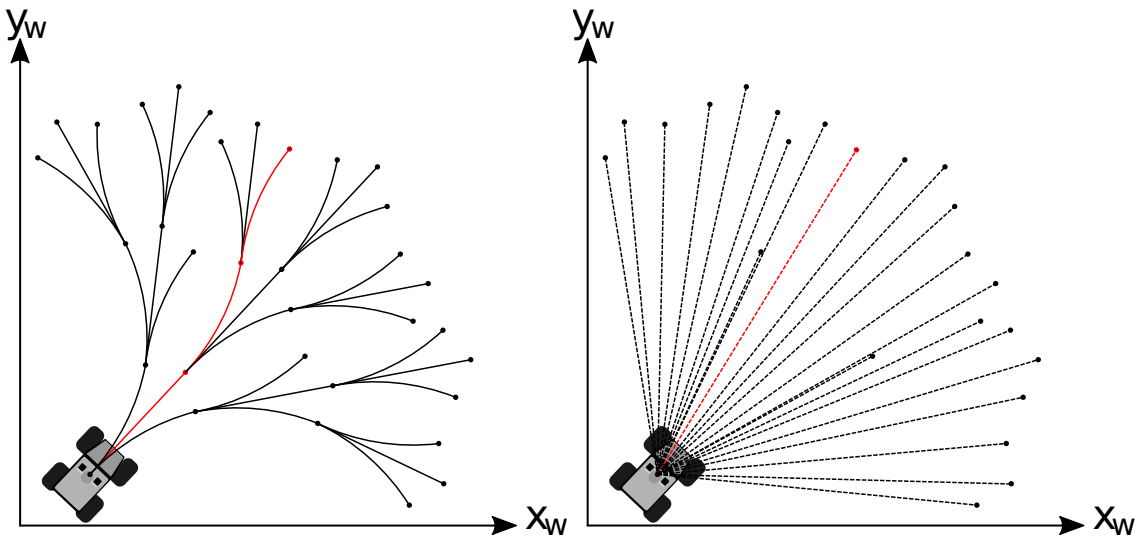
Figure 5.7: *The definition of* α, β *and* d_{min}.

Costs computed in this way provide a realistic estimation of obstacle influence, being rigorous concerning the distance to obstacles, and simultaneously allowing configurations close to the obstacles, if the robot is passing by them. This allows safe paths in dynamic environments, and through narrow passages, even at higher speeds.

5.3.3 Restructuring

After the best local path is found, it may often contain segments that have curvatures of a different sign (as it can be seen on the red path in Figure 5.8a). Different curvatures on a short local path provide either jerky behaviour while following the path at higher speeds, or the speed is reduced in the curves, which then reduces the overall speed.

In order to overcome this problem, we propose to restructure the graph, after the initial search is finished. If we only save the leaf nodes, and neglect the rest of the graph (except for the nodes of the best path), then we have the structure shown in Figure 5.8. The inspiration for this idea comes from the Theta* algorithm presented in (Nash *et al.*, 2007). However, unlike Theta*, we don't only search for the shortest path in continuous space, but rather use this as a mid-step. The next step is to construct new paths by only using



(a) *Initial search* (b) *Considering only the leaf nodes*

Figure 5.8: *After the initial search is finished, only the leaf nodes are considered.*

the saved final leaves, and by applying car-like kinematics again. The newly constructed paths are scored again, and the ones that are too close to an obstacle are discarded. Then, the best of the valid paths is chosen. If it happens that no new, restructured path is found, we use the best non-restructured path. The construction of a new, restructured path can be seen in Figure 5.9, constructed for the same node as the red path in Figure 5.8a. By restructuring the initial search graph in this way, each local path has constant curvature, which allows smooth motion at higher speeds.

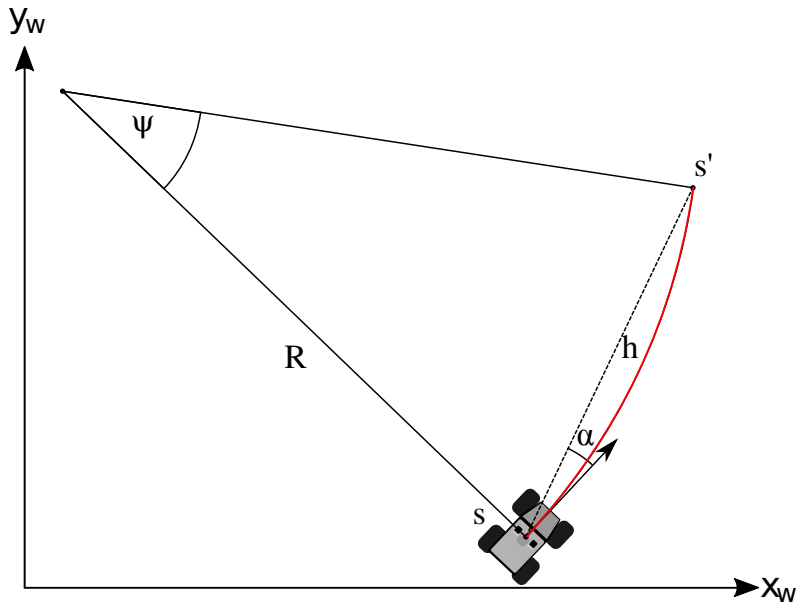


Figure 5.9: *Construction of restructured paths shown on one example node. Here, R is the radius of the curvature, ψ is the angle increment for one time step, s is the current, and s′ the next state, while h represents the linear distance between these two states, and α the angle between the robot's orientation and this linear distance. All of these variables can be computed by using car-like kinematics.*

5.4 Path Following Control

After the local planner has generated an optimized and smooth local path, the robot needs to follow this geometric reference. We use the path following control law defined in Eq. (4.15) for the lateral control, and three different layers of longitudinal control. The first layer is the high speed control defined in Eq. (4.33).

In order to cope with unpredictable dynamic obstacles at higher speeds, we extend this speed control by using a modified version of (3.10). The output speed of the expression (4.33) is restrained by using the following expression:

$$c_o = e^{-\frac{k_o}{d_{min}} cos\alpha}, \tag{5.2}$$

where k_o is a positive constant, and d_{min} and α are the previously defined distance to the nearest obstacle, and the angle between the robot and the obstacle, respectively. The constraint c_o is multiplied with the output speed from (4.33), and will exponentially increase, as the obstacle gets closer, and decrease, as it moves away. At the same time, this effect gets weighted more, if the obstacle is positioned in front of the robot, and less, if the obstacle is on the side of the robot.

5.4.1 Person Following Control

In order to achieve smooth dynamic target (person) following, we need to define a desired curvilinear distance between the robot and the person, which we denote with $l(t)$. Furthermore, we denote the person's position in path coordinates as s_p, and the current robot's position as s. Now, if the current curvilinear distance is $\Delta s = s_p - s$, then we can define the person following error as $e_p = \Delta s - l(t)$. By using the control law (4.15), this error's dynamics can be written as:

$$\dot{e}_p = v_p - (v_x \cos\theta_e + x_{ICR}\omega \sin\theta_e + k_1 x_e) - \dot{l}(t), \tag{5.3}$$

where v_p is the speed of the person. In order for this error to converge to zero, we define the following control expression:

$$v_x = \frac{1}{\cos\theta_e}\left(v_p - x_{ICR}\omega \sin\theta_e - k_1 x_e - \dot{l}(t) + k_e e_p\right), \tag{5.4}$$

where k_e is a positive constant, and by properly initializing the robot, it has to be ensured that $\theta_e \neq (2k+1)\frac{\pi}{2}$, where k is a positive integer. By defining v_x in this way, the person following error's dynamics reduces to $\dot{e}_p = -k_e e_p$, which is an asymptotically stable linear system. For this idea we were inspired by (Lenain *et al.*, 2014).

Now, we state that $V_m = v_x$, and input this control expression to the control scheme (4.33), which is then restrained by (5.2). In this way, we guarantee that the robot smoothly keeps its desired distance to the person, by adjusting its speed according to the person's position and pace. Furthermore, expression (4.33) assures accurate following in curved areas, while the expression (5.2) acts as a safety measure for unexpected dynamic obstacles. Using this speed control scheme, the final speed magnitude will never be greater than V_m, which is bounded, thus the stability is preserved.

Now two classical obstacle avoidance algorithms are introduced, which are used as comparison algorithms against the proposed algorithm consisting of a local planner described in Section 5.3 and a path and person following controller described in Section 5.4.

5.5 The Potential Field Method

The first classical algorithm for comparison is a very well known method, originally proposed in (Koren and Borenstein, 1991) for robot manipulator path planning, and further developed and used in many contexts, such as the mobile robot obstacle avoidance described in (Siegwart *et al.*, 2004). It can also be used as a path following algorithm with inherent obstacle avoidance, if the goal position is always set to coincide with the next point of the reference path.

Let the robot's environment be represented by potentials, in such a way that the po-

tential increases, as the robot approaches the obstacles. By deriving the potentials, the forces acting upon the robot can be defined. So, if an attractive potential of the goal position is defined as a parabolic function, the attractive force can be defined as

$$F_{att}(q) = -k_{att}(q - q_{goal}), \tag{5.5}$$

where k_{att} is a positive parameter, q is the robot's position $q = [x\ y]^T$, and q_{goal} the goal position. Similarly, repulsive forces that drive the robot away from the obstacles can be expressed as

$$F_{rep}(q) = \begin{cases} k_{rep}\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)\frac{q - q_{obst}}{\rho^3(q)}, & \rho(q) \leqslant \rho_0 \\ 0, & \rho(q) \geqslant \rho_0 \end{cases}, \tag{5.6}$$

where k_{rep} is a positive parameter, $\rho(q)$ is the minimal distance from the robot to the obstacle, ρ_0 is a positive parameter which determines the distance of influence of an obstacle, and q_{obst} is the position of the nearest obstacle.

The resulting force that acts upon the robot can be expressed as $F(q) = F_{att}(q) + F_{rep}(q)$, such that the angle of the resulting force may be used as the control steering for the robot.

The Potential Field Method provides a robust obstacle avoidance approach, which may be used for path planning and path following, having its weaknesses with local minima and concave obstacles.

5.6 The Dynamic Window Approach

The second classical algorithm used for comparison is the Dynamic Window Approach (DWA), a very well known obstacle avoidance method proposed in (Fox *et al.*, 1997) for synchro-drives, which considers only short time intervals, or windows, when computing the next steering command, in order to reduce the computation effort and increase the speed. Inside this window, the trajectories are approximated by circular curvatures, and the search space is reduced to the admissible velocities, taking dynamic constraints into consideration. The search among the admissible velocities inside one window maximizes a properly chosen objective function.

Circular trajectories By defining the robot configuration with a triplet (x, y, θ), and observing the motion in a time interval $[t_0, t_n]$, where the linear and angular accelerations are kept constant in a time interval $[t_i, t_{i+1}](i = 1...n)$, the motion of a synchro-drive can be approximated with

$$x(t_n) = x(t_0) + \sum_{i=0}^{n-1} F_x^i(t_{i+1}), \tag{5.7}$$

where

$$F_x^i(t) = \begin{cases} \frac{v_i}{\omega_i}\big[\sin\theta(t_i) - \sin(\theta(t_i) + \omega_i(t-t_i))\big], & \omega \neq 0 \\ t v_i \cos\theta(t_i), & \omega_i = 0 \end{cases}, \tag{5.8}$$

and

$$y(t_n) = y(t_0) + \sum_{i=0}^{n-1} F_y^i(t_{i+1}), \tag{5.9}$$

with

$$F_y^i(t) = \begin{cases} \frac{v_i}{\omega_i}\big[\cos\theta(t_i) - \cos(\theta(t_i) + \omega_i(t-t_i))\big], & \omega \neq 0 \\ t v_i \sin\theta(t_i), & \omega_i = 0 \end{cases}. \tag{5.10}$$

Here, $v_i \in [v(t_i), v(t_{i+1})]$ and $\omega_i \in [\omega(t_i), \omega(t_{i+1})]$.

Admissible velocities If the term $dist(v,\omega)$ represents the distance to the closest obstacle on the corresponding curvature, and $\dot{v}_b$ and $\dot{\omega}_b$ represent the accelerations for breakage, then the set of admissible velocities can be defined as

$$\begin{aligned} V_a = \big\{(v,\omega) | v \leqslant \sqrt{2 dist(v,\omega)\dot{v}_b} \\ \wedge\, \omega \leqslant \sqrt{2 dist(v,\omega)\dot{\omega}_b}\big\}, \end{aligned} \tag{5.11}$$

representing the set of velocities which allow the robot to stop on time, without colliding with obstacles.

Dynamic window In order to reduce the search space and take the dynamics into account, a dynamic window is used, containing only the velocities that can be reached within the next time interval. If t is the time interval, and (v_a, ω_a) is the actual velocity, the dynamic window V_d can be defined as

$$\begin{aligned} V_d = \big\{(v,\omega) | v \in \big[v_a - \dot{v}t, v_a + \dot{v}t\big] \\ \wedge\, \omega \in \big[\omega_a - \dot{\omega}t, \omega_a + \dot{\omega}t\big]\big\}, \end{aligned} \tag{5.12}$$

having its center at the actual velocity and reaching its borders by applying the accelerations during the time interval t.

Search space If the space of possible velocities is defined as V_s, then the resulting search space can be defined as

$$V_r = V_s \cap V_a \cap V_d. \tag{5.13}$$

Objective function After defining the search space, the next objective function is computed over V_r

$$\begin{aligned} G(v,\omega) = &\sigma(\alpha \cdot angle(v,\omega) + \beta \cdot dist(v,\omega) \\ &+ \gamma \cdot velocity(v,\omega)). \end{aligned} \tag{5.14}$$

Here, the target heading $angle(v,\omega)$ measures the angle difference between the predicted robot position and the target direction, the term $dist(v,\omega)$ represents the distance to the closest obstacle that intersects with the curvature, and $velocity(v,\omega)$ is a projection on the translational velocity v. Furthermore, σ, α, β and γ are positive constants.

The Dynamic Window Approach is a fast obstacle avoidance method which takes the dynamics of the vehicle into account, and therefore provides smooth and collision-free trajectories even at higher speeds.

5.7 Experimental Evaluation

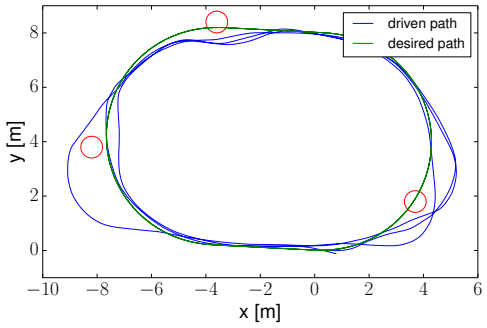
In order to experimentally evaluate the performance of our proposed approach, we have separately tested obstacle avoidance and person following. In all the experiments, a Robotnik Summit XL robot was used, and the maximum speed was around $2.5\mathrm{m\,s^{-1}}$, which is the practical maximum speed of the robot.

5.7.1 Obstacle Avoidance

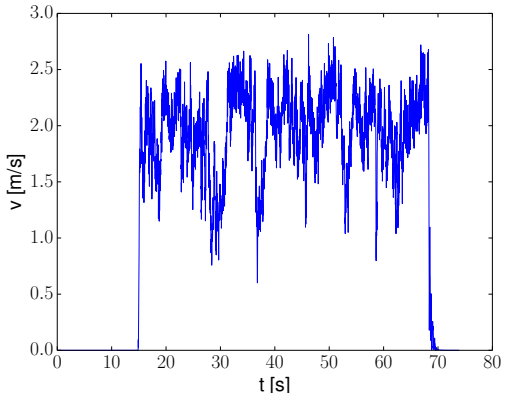
We have compared our approach with the Potential Field Method (PFM), and the Dynamic Window Approach (DWA). These two algorithms are widely used, and the DWA is considered to be a state-of-the-art open-source obstacle avoidance algorithm, available at (Marder-Eppstein, 2017a). The high speed obstacle avoidance algorithm proposed in (Shiller *et al.*, 2013) assumes known environments and the robot's average speed in real experiments is $1.2\mathrm{m\,s^{-1}}$. Our approach deals with unknown environments, and with much higher speeds.

For our experiments, an oval-shaped static path was given on a clear meadow, with three cardboard boxes as obstacles distributed along its circumference. The path was overlaid three times, in order to test the repeatability of the approaches. The total length of the path was 99.15m. All the three approaches had the task of approximately following the static path, while avoiding obstacles on the way.

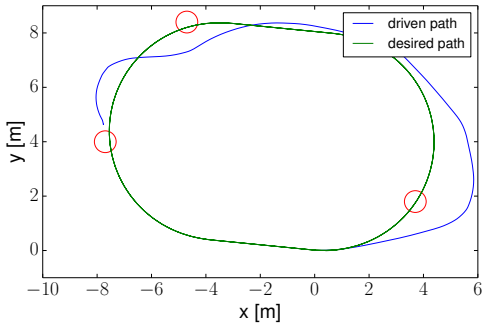
The results can be seen in Figure 5.10, where the planted obstacles are represented by red circles. Both our approach and the DWA were commanded with the maximum speed. Our approach achieves all the three overlaid ovals, while successfully avoiding all the obstacles. The DWA successfully avoids two obstacles, while the third one stops the robot. The PFM was commanded with $1\mathrm{m\,s^{-1}}$, since it was impossible to follow the path at higher speeds. As seen in Figure 5.10, after successfully avoiding all three obstacles



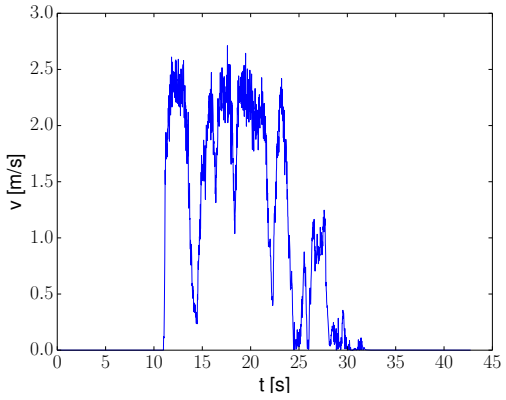
(a) *The proposed approach: successful obstacle avoidance*



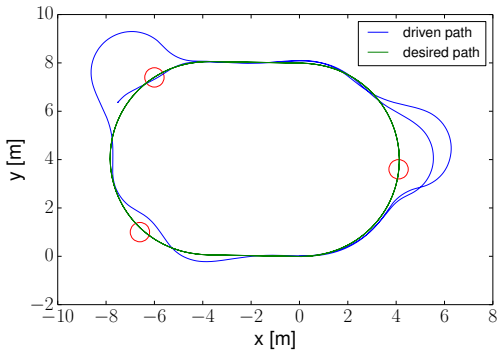
(b) *The proposed approach:* $v_{avg} = 1.95\,\mathrm{m\,s^{-1}}$



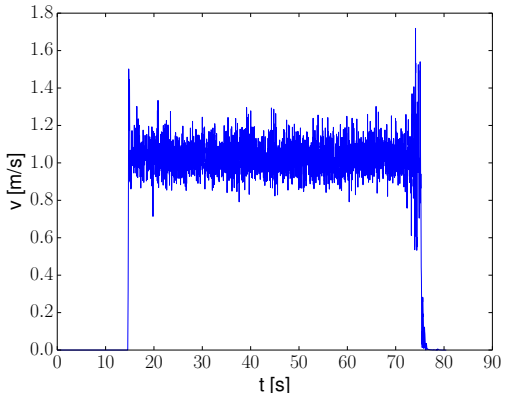
(c) *DWA: stopping in front of the third obstacle*



(d) *DWA:* $v_{avg} = 1.52\,\mathrm{m\,s^{-1}}$



(e) *PFM: running over an obstacle*



(f) *PFM:* $v_{avg} = 1.03\,\mathrm{m\,s^{-1}}$

Figure 5.10: *Obstacle avoidance with three obstacles (red circles) on a three times overlaid,* 99.15m *long, static oval path.*

on the first oval, the second time only the first obstacle is avoided, while the next one is run over.

5.7.2 Person Following

For the person following evaluation, many fully autonomous drives have been made, in various outdoor scenarios, including narrow, cluttered and highly dynamic environments with pedestrians, joggers, and cars, on uneven and challenging terrain. A typical situation can be seen in Figure 5.12. Videos demonstrating the performance of our approach can be found at: https://youtu.be/4O7twdWFm4s.

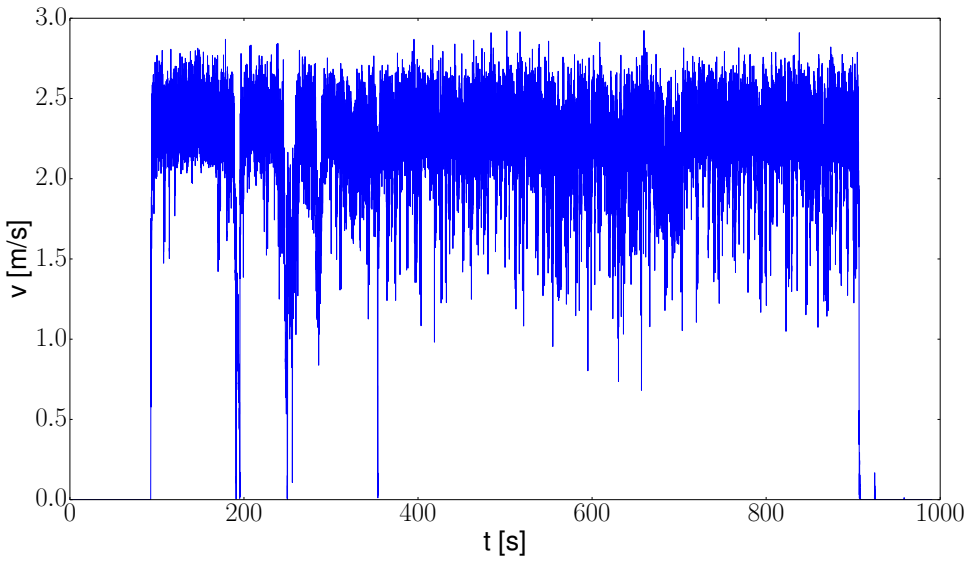
As an example run, in Figure 5.11a a 3D model of our institute and the surroundings is shown, together with the path where our robot autonomously followed a person. The speed profile for this run can be seen in 5.11b.

5.8 Conclusions

We propose a fully autonomous navigation system for outdoor person following at higher speeds using a skid-steered vehicle. Our obstacle avoidance outperforms the classical, well known obstacle avoidance algorithms, as it is shown in the experiments. Our person following with inherently embedded obstacle avoidance runs at the speeds higher than the ones achieved by the state-of-the-art algorithms available in the literature. Our approach shows robust behaviour in arbitrary outdoor environments, autonomously avoiding any danger on the way.

The applications of the proposed approach could be numerous, varying from search and rescue, agricultural, mining, or delivery tasks. The extension of this work would include person identification, since the followed person is perceived as a dynamic object. Another potential extension would be explicit modelling of moving obstacles and predicting their motion.

(a) *The driven path:* 1791.6m



(b) *Speed profile:* $v_{avg} = 2.19\,\mathrm{m\,s^{-1}}$

Figure 5.11: *A fully autonomous person following run at our institute. During this run, the robot encountered various static and dynamic obstacles, all of which were successfully avoided.*

Figure 5.12: *A typical situation when performing autonomous jogger following. The jogger (on the right, white sweater) is being followed, and the pedestrians on the left are avoided.*

Chapter 6

GeRoNa: Generic Robot Navigation

6.1 Introduction

Navigation for mobile robots includes all actions that drive the robot from its current pose to a desired destination. This implies that the current position already needs to be known, i.e. the robot needs to be able to localize itself, and the desired destination needs to be defined. By having all this initial information, the robot should be able to plan its collision-free motion from the current to the desired pose.

A well known open-source framework for solving this problem is the ROS Navigation Stack, presented in (Marder-Eppstein, 2017b). It is a 2D navigation package, which takes the information from the odometry and the sensors as input, then computes the velocity command and sends it to the mobile base. The framework is written in C++/ROS (Robot Operating System). Details on ROS can be found in (Quigley *et al.*, 2009). Although the ROS Navigation Stack is a very useful framework, and already implemented on several different robot platforms, it still has some major limitations, as stated in (Marder-Eppstein, 2017b):

- The framework is meant for only differential drives and holonomic wheeled robots.
- It requires a planar laser mounted on the robot, needed for map building and localization.
- It performs best on robots that are nearly square or circular, and it may have difficulties with large rectangular robots in narrow spaces.

These limitations prevent the usage on arbitrary robots, and in outdoor environments, where planar lasers are not enough for mapping and localization. Furthermore, the framework's simple control does not account for different kinematic and dynamic effects, which become crucial in high speed and outdoor applications. Driven by these reasons, we have developed another framework, which we named GeRoNa (Generic Robot Navigation), and which can be used on an arbitrary wheeled mobile robot, regardless of its kinematics, sensors, or dimensions, both indoors and outdoors. This framework is written in ROS with C++, as well, and is already open-source available. It is also highly modular, so that each component can be easily exchanged, or extended.

GeRoNa assumes that the robot has at least odometry data, which is the minimum required input, while the output is the computed command velocity. Optionally, if the robot has some exteroceptive sensors, their streams are unified as a point cloud, which can be used for obstacle detection. Furthermore, arbitrary SLAM (Simultaneous Localization and Mapping) algorithms can be used. Otherwise, simple odometry is sufficient. The goal position can be either chosen by the user, or autonomously, e.g. by decision making in an exploration scenario. After the goal pose is set, the robot plans its collision-free path using A*-based path planning, and then follows the path by using one of the state-of-the-art control algorithms from the provided collection. This collection consists of twelve different control algorithms, and can be easily extended. Furthermore, the robot can avoid both static and dynamic obstacles, even at speeds up to $2.5\,\mathrm{m\,s^{-1}}$, and follow dynamic objects, as described in Chapter 5. The framework was successfully tested on eight different real-world robotic platforms, both indoors and outdoors, in cluttered and dynamic environments, flat and rough terrains, moving from low speeds up to $6\,\mathrm{m\,s^{-1}}$, and includes all the work previously described in this thesis, as well as some other state-of-the-art algorithms.

A video demonstrating GeRoNa can be found at: https://youtu.be/Ppdi7dQ7Vzw.

6.2 Framework

The GeRoNa architecture is shown in Figure 6.1. The implementation is split into three modules: *high level interface*, *path planning* and *path following*. Communication between modules and with client applications is implemented using the ActionLib library, which is provided as a ROS package. In contrast to standard, topic-based communication, this allows the system to constantly give feedback about the current state of execution. This means that high level code has fine-grained control over the navigation system.

The architecture is also kept modular concerning dependencies on available data. This means that both the planning and following modules can use available mapping and localization data, but they also work if no such information is available. The same is true for obstacle detection. The above mentioned point cloud interface is kept as generic as possible, such that all kinds of sensor setups can be used. Algorithms for obstacle detection can range from a standard planar LIDAR setup, where every measurement results in an obstacle point, to more elaborate algorithms, such as (Buck *et al.*, 2016a), where 3D sensor data is processed and filtered.

6.2.1 High Level Interface

The interface to high level applications is responsible for controlling the path following process and is called *path_control*. It is implemented as a facade for the navigation system and therefore represents the primary interface for client applications.

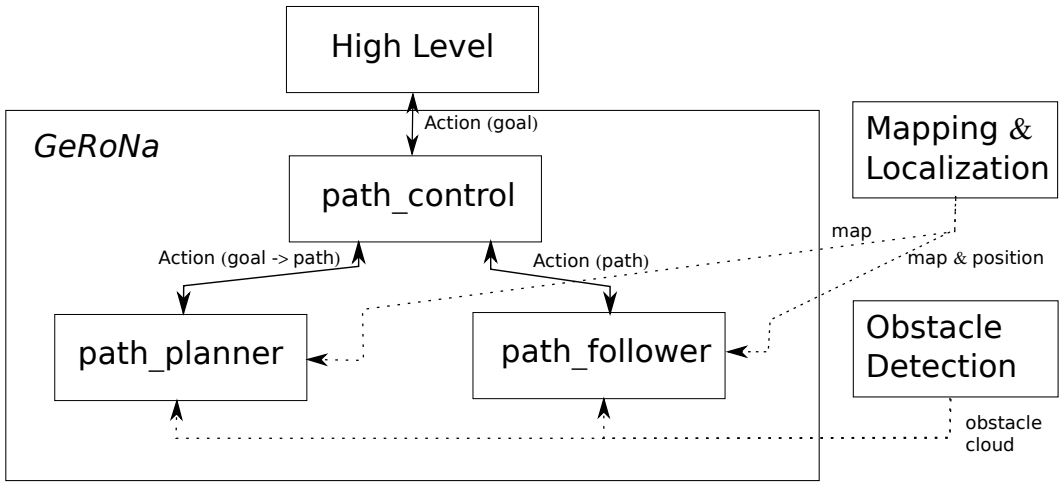


Figure 6.1: *System architecture of the GeRoNa framework. Three components are included: A high level interface (path_control), several path planning algorithms (path_planner), and a path following sub-system (path_follower.) Mapping, localization and obstacle detection algorithms are supported, but not included.*

A goal specification for the navigation is given as an action to this module. The *path_control* module then instructs the *path_planner* module to generate a path, which is then relayed to the *path_following* node. Due to the feedback mechanism of ActionLib, *path_control* can constantly supervise the path following behaviour and is able to stop the robot when an interruption occurs. The module is also able to issue requests to replan the global path multiple times, without relying on the high level application. Due to the modular approach, it is also possible for users to communicate with *path_planner* and *path_follower* directly.

6.2.2 Path Planning

The *path_planning* module contains several implementations of path planning algorithms. The goal specification can request a specific planner, otherwise a default planner will automatically be used to find a path to the given goal. In Section 6.3 we present three path planning algorithms that are available by default. It is also easy to add new path planners.

6.2.3 Path Following

The *path_follower* module is responsible for following a given path as accurately as possible. It is further subdivided into a *local planner* system and a set of *controllers*. The individual components are described in more detail below, where the high speed local

planner is shown in Section 6.3. We especially focus on the available implementations of different controllers in Section 6.4.

6.3 Planning Algorithms

Path planning in GeRoNa can be divided into *global planning* and *local planning*. Both of these planning profiles are described in this section.

6.3.1 Global Planning

Global planning plans a path between the robot and the goal pose in a given map. The map is optional and needs to be provided by some external node. As the robot perceives obstacles, they are integrated into the map before the search. If no map is available, only the obstacles are used for planning.

There are three planning algorithms available:

- Kinematic Path Planner,
- Static Planner,
- Course Planner.

Kinematic Path Planner This planning algorithm is based on the A* approach described in (Hart *et al.*, 1968). Here, a node in the search tree is expanded by simulating the kinematics of an Ackermann vehicle for a small period of time, taking into account the resolution of the grid. Paths generated using the Ackermann kinematics are smooth and drivable by almost every wheeled mobile robot.

Static Planner This algorithm specifies a simple sequence of straight lines and curves, starting at the goal pose specified in the request. The sign of the straight line determines the direction, where a negative distance means that the robot should drive in reverse. An example path planned by this algorithm can be seen in Figure 6.2. The robot is seen as a gray object, and the path is represented as follows: a green curve segment with a central angle π and a radius of 4m, a straight green line of length 4m, another green curve segment with a central angle $-\pi$ and a radius of 4m, and a red straight line of length -4m. The red line needs to be driven in reverse.

Course Planner The course planner uses directed path segments, and plans a topological path on these segments. An example can be observed in Figure 6.3. The blue arrows represent the directed path segments, the purple circles and the dark green arcs smooth crossings, and the light green line represents the planned path from the goal pose,

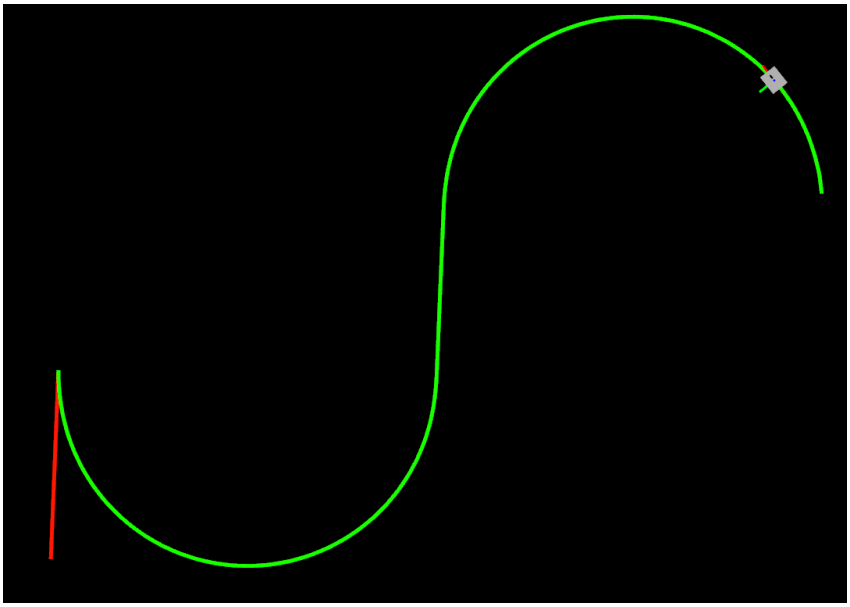


Figure 6.2: *An example path planned by the Static Planner, visualized in ROS/RViz.*

denoted by the red arrow, up to the robot, represented by the grey object. This search is very fast, since already defined directed segments are used, and the only grid search is the one used for finding the path between the goal pose and the closest directed path segment, as seen in the lower right corner.

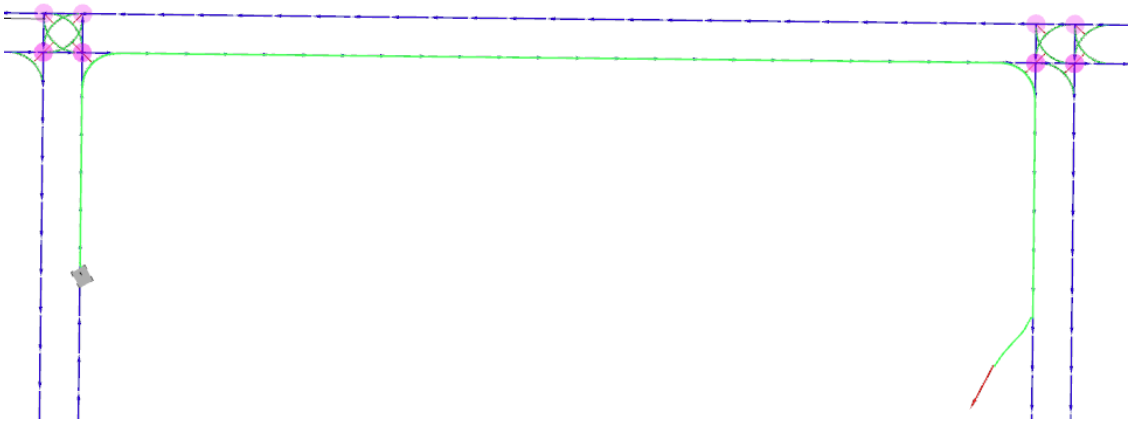


Figure 6.3: *An example path planned by the Course Planner in the long corridors at our institute.*

6.3.2 Local Planning

Details on the local planning are described in Chapter 5.

6.4 Control Algorithms

All the control algorithms described in the previous chapters are integrated into GeRoNa. In this section, additional controllers available in GeRoNa are described. The controllers in GeRoNa are classified as the algorithms that send control commands to the robot and can be used for path following. Most of the controllers are pure path following algorithms, while some of them are obstacle avoidance, or target tracking algorithms, capable of following a geometric reference path.

6.4.1 The Input-Scaling Controller

This controller is the path following algorithm presented in (De Luca *et al.*, 1998). To simplify its naming, we will refer to it as the Input-scaling controller, named after the control derivation procedure. This controller can be used for unicycles, Ackermann steering, and bi-steerable vehicles. Kinematic models of these vehicles are assumed to be subject to the nonholonomic constraint, i.e. there is only pure rolling in the longitudinal direction, and no lateral nor longitudinal slipping.

Unicycle The basis for modelling of such behaviour is the kinematic model of a single wheel. A single wheel exhibits pure rolling in its longitudinal direction, and can rotate around its vertical axis. In Figure 6.4 this behaviour is visualized, where v stands for the linear velocity, θ is the orientation, and $\dot{x}$ and $\dot{y}$ are the longitudinal and lateral velocity components in the global frame, respectively. This kind of a kinematic model of a single wheel in global coordinates is usually referred to as the unicycle model. The unicycle model can be applied to differential drives, since in the global sense, the vehicle can be seen as a single wheel. Namely, differential drives steer in a way that their wheels turn with different speeds, but globally seen, the whole vehicle rotates around its vertical axis. The same applies for synchro-drives. This kind of modelling might even be used to approximate skid-steered vehicles, where the steering principle is equivalent to the differential one, even though not identical. This has been discussed in Chapter 4. The kinematic model of a unicycle can now be written as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{6.1}$$

where v and ω represent the linear and rotational velocity, respectively. Following e.g. (Samson, 1992), this model can be expressed in path coordinates as

$$\begin{bmatrix} \dot{s} \\ \dot{d} \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} \frac{\cos\theta_e}{1-dc(s)} & 0 \\ \sin\theta & 0 \\ -\frac{c(s)\cos\theta_e}{1-dc(s)} & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{6.2}$$

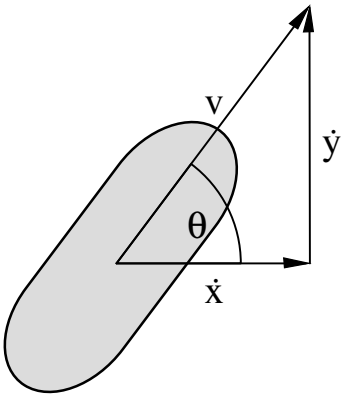


Figure 6.4: *Kinematic representation of a unicycle.*

where s is an arbitrary path parameter, d is the orthogonal distance from the robot to the path, $c(s)$ is the path curvature, and $\theta_e = \theta - \theta_p$ is the orientation error, where θ is the robot's global orientation, and $\theta_p = c(s)\dot{s}$ is the angle between the current tangent of the path and the global x axis.

Following the work presented in (De Luca *et al.*, 1998), the model in Eq. (6.2) can be transformed to the so called chained form, a canonical form for kinematic models of nonholonomic robots suitable for systematic development of control strategies. For a two-input system, which covers the most of the kinematic models of wheeled mobile robots, the chained form with generalized coordinates $x_1, ..., x_n$ and inputs u_1, u_2 can be expressed as

$$\begin{aligned} \dot{x}_1 &= u_1, \\ \dot{x}_2 &= u_2, \\ \dot{x}_3 &= x_2 u_1, \\ &\vdots \\ \dot{x}_n &= x_{n-1} u_1. \end{aligned} \tag{6.3}$$

In order to transform the model in Eq. (6.2) to the chained form, the following change of coordinates should be used

$$\begin{aligned} x_1 &= s, \\ x_2 &= (1 - dc(s)) \tan \theta_e, \\ x_3 &= d, \end{aligned} \tag{6.4}$$

together with the input transformation

$$\begin{aligned} v &= \frac{1-dc(s)}{\cos\theta_e}u_1, \\ \omega &= u_2\frac{\cos^2\theta_e}{1-dc(s)} + u_1\left[c(s)\left(1+\sin^2\theta_e\right) + \right. \\ &\left. d\frac{\partial c(s)}{\partial s}\frac{\sin\theta_e\cos\theta_e}{1-dc(s)}\right]. \end{aligned} \tag{6.5}$$

As stated in (De Luca *et al.*, 1998), when u_1 is a piecewise-continuous, bounded, and strictly positive or negative function, a controllable system transformation can be derived

$$\begin{aligned} \dot{\chi}_1 &= u_1, \\ \dot{\chi}_2 &= \chi_3 u_1, \\ \dot{\chi}_3 &= \chi_4 u_1, \\ &\vdots \\ \dot{\chi}_{n-1} &= \chi_n u_1, \\ \dot{\chi}_n &= u_2, \end{aligned} \tag{6.6}$$

where

$$\boldsymbol{\chi} = (\chi_1, \chi_2, ..., \chi_{n-1}, \chi_n) = (x_1, x_n, ..., x_3, x_2)\,. \tag{6.7}$$

In the case of a unicycle, the transformed controllable system can be expressed as

$$\begin{aligned} \dot{\chi}_1 &= u_1, \\ \dot{\chi}_2 &= \chi_3 u_1, \\ \dot{\chi}_3 &= u_2. \end{aligned} \tag{6.8}$$

Following the procedure originally proposed in (Samson, 1992), and further developed in (De Luca *et al.*, 1998), the first input for the unicycle model is chosen as $u_1 = v$, where v is the linear velocity, and the second one is derived as $u_2 = -k_1\chi_2 u_1 - k_2|u_1|\chi_3$, where k_1 and k_2 are positive constants. By using the transformation (6.7), the inputs for the unicycle model can be written as

$$\begin{aligned} u_1 &= v, \\ u_2 &= -k_1 u_1 x_3 - k_2 u_1 x_2. \end{aligned} \tag{6.9}$$

The procedure of finding these control inputs is called the *input-scaling* procedure, since the second control input is scaled by the first one, otherwise there would be a singularity at $u_1 = 0$. Furthermore, this also implies that the second control input goes to zero when the first one is zero. Further details can be found in (De Luca *et al.*, 1998).

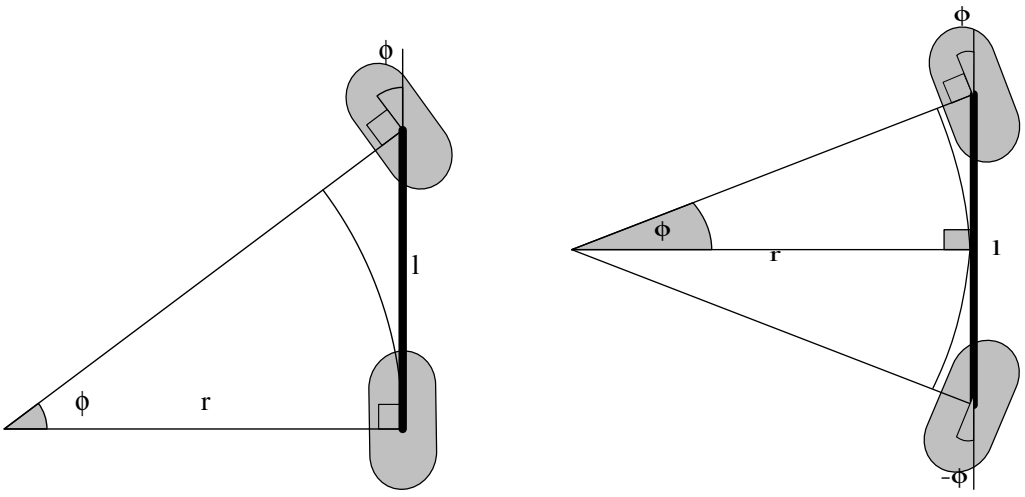
Ackermann and double steering For the kinematic modelling of Ackermann and double steering (bi-steerable) vehicles, it is reasonable to use the so called bicycle model to describe the motion. With this simplified model it is assumed that the Ackermann vehicle has perfectly aligned front wheels, where both of them have the same steering angle. For the case of bi-steerable vehicles, this assumption applies to the rear wheels as well. For both of these kinematic configurations, the distance between the two wheels on the same axis is neglected. In Figure 6.5a and Figure 6.5b the bicycle model of Ackermann and bi-steerable vehicles is shown, from which the basic relations between the length of the vehicle l, its steering angle ϕ, and the radius of curvature r can be concluded. For Ackermann vehicles this relation can be expressed as

$$\tan\phi = \frac{l}{r}. \tag{6.10}$$

In the case of bi-steerable vehicles, this relation becomes

$$\tan\phi = \frac{l}{2r}. \tag{6.11}$$

It is easy to see that with this simple model bi-steerable vehicles can make exactly two times sharper turns than Ackermann vehicles. Using these two relations and by observ-



(a) *Ackermann* (b) *bi-steerable*

Figure 6.5: *Kinematic representations of Ackermann and bi-steerable vehicles.*

ing Figure 6.5, the kinematic model of an Ackermann vehicle with rear-wheel driving

can be expressed as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ \tan\phi/l & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{6.12}$$

where v and ω represent linear and steering velocity, respectively. For an ideal bi-steerable vehicle, the expression $\tan\phi/l$ becomes $2\tan\phi/l$. In order to be able to follow a geometric path, it is reasonable to represent this kinematic model in path coordinates as

$$\begin{bmatrix} \dot{s} \\ \dot{d} \\ \dot{\theta}_e \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \frac{\cos\theta_e}{1-dc(s)} & 0 \\ \sin\theta_p & 0 \\ \left(\frac{\tan\phi}{l} - \frac{c(s)\cos\theta_e}{1-dc(s)}\right) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{6.13}$$

for the Ackermann steering, and with $2\tan\phi/l$, instead of $\tan\phi/l$, for ideal bi-steerable vehicles. Now, similar as with the unicycle, this model can be expressed in the chained form as

$$\begin{aligned} x_1 &= s, \\ x_2 &= -\frac{\partial c(s)}{\partial s} d\tan\theta_e - c(s)\big(1-dc(s)\big)\frac{1+\sin^2\theta_e}{\cos^2\theta_e} + \\ &\quad \frac{\big(1-dc(s)\big)^2\tan\phi}{l\cos^3\theta_e}, \\ x_3 &= (1-dc(s))\tan\theta_e, \\ x_4 &= d, \end{aligned} \tag{6.14}$$

with the output transformation defined as

$$\begin{aligned} v &= \frac{1-dc(s)}{\cos\theta_e} u_1, \\ \omega &= \alpha_2(u_2 - \alpha_1 u_1), \end{aligned} \tag{6.15}$$

where

$$\begin{aligned} \alpha_1 &= \frac{\partial x_2}{\partial s} + \frac{\partial x_2}{\partial d}(1-dc(s))\tan\theta_e + \\ &\quad \frac{\partial x_2}{\partial \theta_e}\left(\frac{\tan\phi(1-dc(s))}{l\cos\theta_e} - c(s)\right), \\ \alpha_2 &= \frac{l\cos^3\theta_e\cos^2\phi}{(1-dc(s))^2}. \end{aligned} \tag{6.16}$$

Now, by following the above mentioned input-scaling procedure, the outputs for an Ackermann vehicle are

$$\begin{aligned} u_1 &= v, \\ u_2 &= -k_1|u_1|x_4 - k_2 u_1 x_3 - k_3|u_1|x_2, \end{aligned} \tag{6.17}$$

where k_1, k_2, and k_3 are positive constants. For an ideal bi-steerable drive, in equations (6.14) and (6.16) the term $\tan\phi/l$ should be replaced by $2\tan\phi/l$, and the rest of the procedure remains the same.

The Input-scaling controller is an accurate path following algorithm for unicycles, Ackermann and bi-steerable vehicles. When used for the latter two, access to the steering angle measurements is necessary.

6.4.2 The Pure Pursuit Controller

This is a well known algorithm presented in (Coulter, 1992). The idea is to always look ahead of the robot at a certain point on the reference path, and approach this point by moving along a locally constructed arc. This local goal point can be considered as a virtual vehicle, and its distance from the robot is a parameter named *lookahead distance*. The principle of the algorithm can be seen in Figure 6.6, where the radius of the current arc is denoted as r, and the angle between the vehicle's orientation and the line connecting the vehicle and the lookahead point as α. Similar as in (Snider *et al.*,

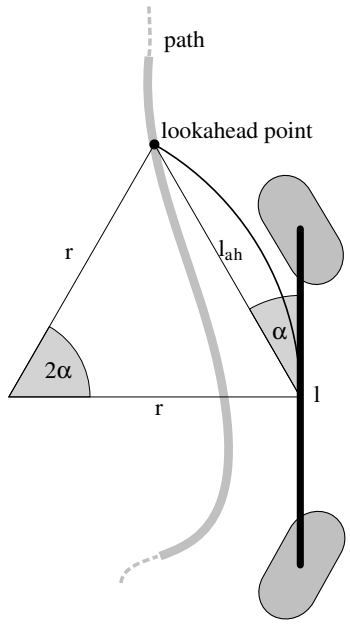


Figure 6.6: *Geometry of the Pure Pursuit algorithm.*

2009), by applying the law of sines, the following relations can be derived:

$$\begin{aligned} \frac{r}{\sin(\frac{\pi}{2}-\alpha)} &= \frac{l_{ah}}{\sin(2\alpha)}, \\ \frac{r}{\cos\alpha} &= \frac{l_{ah}}{2\sin\alpha\cos\alpha}, \\ r &= \frac{l_{ah}}{2\sin\alpha}, \end{aligned} \tag{6.18}$$

where l_{ah} is the lookahead distance. Now, the control command for Ackermann vehicles can be expressed as

$$\phi = \tan^{-1}\left(\frac{2l\sin\alpha}{l_{ah}}\right), \tag{6.19}$$

while the same control for ideal bi-steerable vehicle would be computed as

$$\phi = \tan^{-1}\left(\frac{l\sin\alpha}{l_{ah}}\right). \tag{6.20}$$

In practice, the lookahead distance is usually scaled by the speed, since the minimum radius of curvature increases at higher speeds, in order to avoid slippage or tipping over. A common solution is to scale the lookahead distance proportionally, such that $l_{ah} = kv$, where k is a positive constant.

The Pure Pursuit controller provides a smooth solution for Ackermann and bi-steerable vehicles, since it is locally always moving along arcs.

6.4.3 The Stanley Controller

This algorithm was used by the Stanford University on an autonomous car named *Stanley*, which led them to win the DARPA Grand Challenge. The algorithm is described in detail in (Thrun *et al.*, 2006) and (Hoffmann *et al.*, 2007). The main idea can be derived by observing Figure 6.7. The lateral error e between the vehicle and the closest point on the path P needs to be minimized, for which a nonlinear feedback function can be used, providing exponential convergence, as shown in (Hoffmann *et al.*, 2007). The steering control law is given as

$$\phi = \theta_e + \tan^{-1}\left(\frac{ke}{v}\right), \tag{6.21}$$

where θ_e represents the difference between the orientation of the vehicle, and the angle of the path tangent in the point P, while k is a positive parameter. The first part of Eq. (6.21) sets the control steering angle to compensate for the orientation error, while the second part steers the robot towards the path, as the lateral error increases.

The Stanley controller provides a robust solution for Ackermann vehicles, even if using only the kinematic equations of the controller. For the dynamical part of the con-

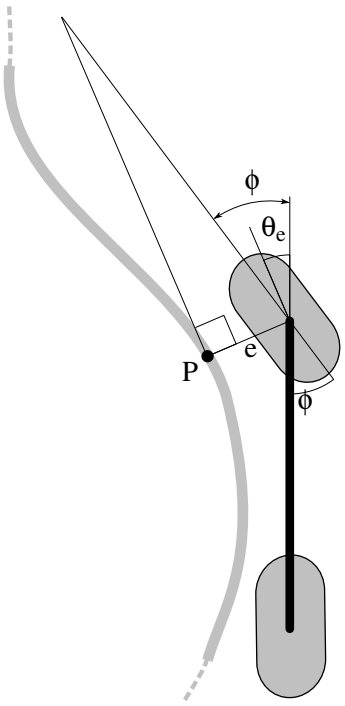


Figure 6.7: *Geometry of the Stanley algorithm.*

troller, see (Hoffmann *et al.*, 2007).

6.4.4 The Ackermann-PID Controller

This algorithm is a simple PID controller for Ackermann robots. The main idea is to fit a line to the path and then drive the robot towards that line. For this we select the next way point wp_n on the path in front of the robot and construct a line through wp_n and wp_{n-1}, as shown in Figure 6.8.

Then the front and rear axle positions are predicted into the future. For both predictions the signed distance to the line is calculated, resulting in two error terms e_f and e_r.

Then the overall error for the PID controller to minimize is then simply given by

$$e = k_1 \cdot e_f + k_2 \cdot e_r + \theta_t - \theta, \tag{6.22}$$

where θ_t is the angle of the line and θ is the orientation of the robot, while k_1 and k_2 are positive parameters. This simple law thus drives the angle difference between the line and the robot, as well as the distances to the line to zero.

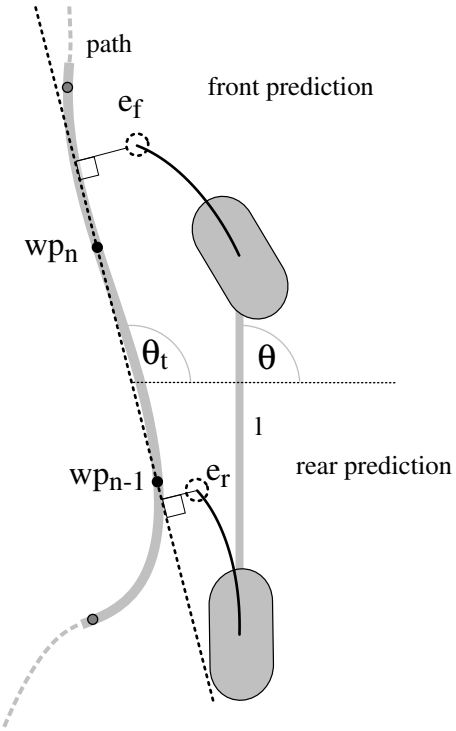


Figure 6.8: *Geometry of the Ackermann-PID algorithm.*

6.4.5 The OFC Controller

The Object Following Controller (OFC), as proposed by the authors in Olmedo *et al.* (2014) and Leigh *et al.* (2015), is a reactive controller for dynamic object following based on PID control. The OFC controller uses two vectors to compute the position and the orientation error of the tracked dynamic object with respect to a fixed goal position, defined in robot's coordinates. The OFC controller will steer the robot in such a way, that the dynamic object's position and the goal position coincide. As it can be seen in Figure 6.9, the first vector is defined from the robot's center O to the goal position G, and is a constant, while the second vector is defined from the robot's center O to the position of the tracked object M. By using a PID controller, the command angular velocity is computed using the angle θ_r of the difference vector $\mathbf{r}$ as an error input, while the magnitude of the difference vector $\mathbf{r}$ is used as an input for a second PID controller, in order to compute the linear velocity.

The OFC controller offers a simple, reactive approach for following of dynamic objects. It is also possible to follow any path, instead of a dynamic object, just by letting the goal position coincide with the next point of the reference path in front of the robot.

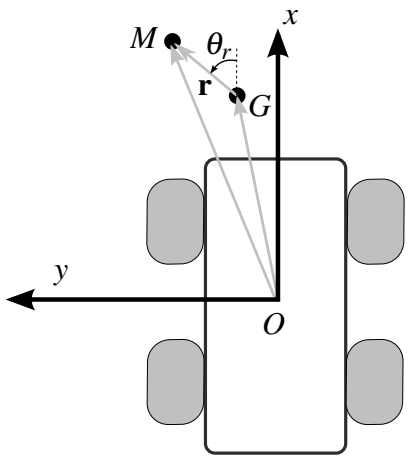


Figure 6.9: *The idea of the OFC algorithm.*

6.5 Conclusions

We present a modular framework GeRoNa for navigation of any wheeled mobile robot. GeRoNa is implemented in ROS/C++ and the code is available open-source at this link: https://github.com/cogsys-tuebingen/gerona. In the online documentation, the usage of individual software components is explained. In this thesis, the theoretical background of all components is given, as well as experimental evaluation and comparison between the algorithms of a same type.

GeRoNa offers more than the ROS Navigation Stack, presented in Marder-Eppstein (2017b), since it can be used to control any wheeled mobile robot, regardless of its dimensions, kinematics, or sensors. It provides not only several state-of-the-art algorithms, but also authors' work which outperforms state-of-the-art. The framework can be used indoors, as well as outdoors, regardless of the terrain conditions, even in highly dynamic environments at higher speeds.

Chapter 7

Conclusions

This thesis presents a study on control of wheeled mobile robots, especially at higher speeds, and in outdoor conditions. The study comprises of examining path following in general, as well as high speed control, planning, obstacle avoidance and person following using skid-steered vehicles at higher speeds on challenging terrain.

As a result of examining path following of wheeled mobile robots in general, a simple, yet robust heuristic method is developed. The proposed method deals with longitudinal control of mobile robots, independent of the kinematic configuration. Such a control scheme provides safer navigation in cluttered and dynamic areas, taking into account the neighbouring obstacles, the distance to the goal position, curvature of the geometric reference path and the current angular velocity. This solution is then coupled with the lateral control proposed in (Mojaev and Zell, 2004), which was further developed in (Li, 2009). The integral solution is implemented on three different kinematic configurations: omnidirectional, Ackermann and bi-steering. The implementation on the omnidirectional robot is evaluated in the dynamic and challenging robotic competition SICK Robot Day 2014. The competition simulated a warehouse scenario, in which four robots were supposed to collect and deliver special objects at the same time. The goal was to collect as many objects as possible in a specified time window, while safely navigating among other robots. Using the presented approach, our team Attempto Tübingen scored 2nd, among 14 international teams. Further evaluation of the approach includes experiments in laboratory conditions on the three mentioned kinematic configurations, and comparing the approach with the one presented in (Mojaev and Zell, 2004). The results show smoother and more stable motion, when using the proposed approach. Furthermore, in the case of the proposed approach, the experiments are more repeatable and reliable, having much lower standard deviation values. The reason lies in the fact that the proposed longitudinal control stabilizes the vehicle in sharp curves, as well as in the vicinity of obstacles, while driving with a constant speed increases the risk of largely deviating from the path, or hitting an obstacle. The presented experiments consist of a U-shaped geometric reference path, and a planted obstacle in the area of highest curvature. In five consecutive runs, the proposed approach did always follow the path without touching the obstacle, while the compared approach did deviate from path and hit the obstacle in every run. This approach is published in (Huskić *et al.*, 2016) and (Buck *et al.*, 2015).

As a result of studying the motion at higher speeds on different terrain types, a non-linear robust control law for skid-steered vehicles is developed, which allows high speed control on uneven terrain. Skid-steered vehicles *per se* present a challenging problem in terms of modelling and control, which becomes even more challenging in high speed and off-road conditions. The proposed control algorithm consists of two parts: a lateral, and a longitudinal one. As a first step, a new representation of skid-steered vehicles in path coordinates is derived. This new kinematic model is used to develop the lateral control using the Lyapunov theory. The lateral control guarantees the convergence of the vehicle to the path, and the stability is proven. The experimentally identified model is furthermore used to design longitudinal control, where reachable curvatures are taken into account, as well as actuator saturation, together with the asymmetry of skid steering, and the previously constructed Lyapunov function. The integral solution provides a robust and accurate control law for skid-steered vehicles, even at high speeds and rough terrain. The approach is evaluated on several different terrain types, using a Robotnik Summit XL robot at speeds up to $\approx 2.5\,\mathrm{m\,s^{-1}}$, and a Segway RMP 440 robot at speeds up to $\approx 6\,\mathrm{m\,s^{-1}}$. In contrary to many related manuscripts, where a new control law is presented without comparing it with other approaches, we make a thorough experimental comparison between the proposed approach, and two other state-of-the-art approaches. The first state-of-the-art approach, named *SLPINL* here, is proposed in (Soetanto *et al.*, 2003), and extended in (Indiveri *et al.*, 2007), while the second one, here named *PBR*, is proposed in (Pentzer *et al.*, 2014b), which is mainly based on (Pentzer *et al.*, 2014a) and (de Wit *et al.*, 1993). In the experiments, while driving with both Robotnik Summit XL and Segway RMP 440 on different paths, different speeds, and different terrain types, the proposed approach outperforms the two state-of-the-art approaches by far.

The reason lies in the completeness of the control law. The model in path coordinates is based on the kinematic model presented in (Mandow *et al.*, 2007), where the parameters are experimentally evaluated, capturing dynamics implicitly in the kinematic model. The lateral control takes into account the most important parameter from the kinematic model, the longitudinal offset, which is experimentally found and bounded. There is no need in artificially setting this parameter to a fixed value, like in e.g. (Caracciolo *et al.*, 1999). By using the Lyapunov theory, the model is then driven towards the path. Finally, the longitudinal control maximizes the speed, while minimizing the error, and taking the entire kinematic model into account.

The experiments conducted with Segway RMP 440 at high speeds on uneven terrain covered with high grass, with a lemniscate as a geometric reference path, show an interesting property of the proposed longitudinal control. Even if commanded with a desired speed of $6\,\mathrm{m\,s^{-1}}$, the average speed is similar to the one when commanded $3.5\,\mathrm{m\,s^{-1}}$. This is a result of the longitudinal control not letting the vehicle drive beyond its capabilities. The desired speed is only reached on the straight segments of the path, while in the curves, the speed is reduced in order to keep the track of the path. If the radius of curvature of the path was bigger, the longitudinal control would allow the robot to drive faster in the curves, and meet the desired speed value. The results on high speed control

on rough terrain with skid steered vehicles are published in (Huskić *et al.*, 2017d) and (Huskić *et al.*, 2017b).

An alternative to the proposed approach would be an algorithm based on the kinematic model proposed in (Wong, 2001). In this case, kinematic model is not expressed in coordinates of the Instantaneous Centres of Rotation, but rather in wheel slip ratios. This type of a model is adaptive and requires an accurate estimation of the individual wheel speeds. However, such estimations usually require additional sensors like Inertial Measurement Units (IMU), and highly depend on the calibration and measurement accuracy. The proposed approach does not require any additional sensors, it can be used by any vehicle with a minimal localization system, such as odometry. This allows an easy transfer between vehicles, which was confirmed by testing the approach on two vehicles of very different dimensions and dynamic properties, namely the Robotnik Summit XL, and the Segway RMP 440 robot.

After the kinematic modelling and control design of skid-steered vehicles, a control design that couples the kinematic with the dynamic model is introduced. It is derived using a generalized dynamic model, extending the approach presented in (Huskić *et al.*, 2017d) and (Huskić *et al.*, 2017b). The resulting control law takes both the kinematic and the dynamic law into account, and the stability of the entire system is proven. In this way, it is possible to take dynamic effects into account, such as e.g. friction, or inertia.

As a result of studying planning, obstacle avoidance, and person following at higher speeds in outdoor conditions, a new approach is developed, which enables a skid-steered vehicle to follow a human jogger in all-terrain conditions. The approach consists of two parts: path planning and control.

The path planning part is a two-layer system, where the top layer we refer to as the global planner, and the bottom layer as the local planner. The global planner is basically an A* planner, as described in (Hart *et al.*, 1968). This planner finds a kinematically feasible path from the robot to the current position of the person that needs to be followed. This gives a general direction to the local planner, which then searches for a locally optimal path in the vicinity of the global path, taking kinematics, dimensions of the robot, as well as obstacles into account. The search uses a new circular search space discretization, where around each node there is a constructed circle with a radius equal to the distance between the two neighbouring nodes of the same level. Local paths that go outside of the pre-specified vicinity of the global path, or go near to an obstacle, are cancelled. Otherwise, the costs are computed taking the distance to the global path, curvature, and distance to the nearest obstacle into consideration. The costs depending on the distance to the obstacles are computed in a way that they allow paths passing close to, but not through an obstacle. This allows higher speeds even in cluttered and narrow areas.

The control part is based on the control proposed in (Huskić *et al.*, 2017d) and (Huskić *et al.*, 2017b), where the longitudinal control is extended by terms that take into account the distance to the nearest obstacle, as well as distance to the tracked person. There is a desired distance between the robot and the person, which is easily defined as a

parameter, and the longitudinal control then adapts the speed of the robot, so that the desired distance is kept. If the person slows down, the robot will slow down as well. If the person accelerates, the robot keeps up.

The integral approach is evaluated in two different ways: as an obstacle avoidance method, and as a navigation system for person following. In the first case, instead of following a running person, the robot was set to follow a static path with three planted obstacles on unknown positions. The approach is then tested against two classical obstacle avoidance methods, the Dynamic Window Approach, proposed in (Fox *et al.*, 1997), and the Potential Fields Method, described in (Siegwart *et al.*, 2004). The comparison was performed by having the desired speed set to $2.5\text{m}\,\text{s}^{-1}$, and the goal was to safely follow the path three times in a row. Only the proposed approach could finish the task without collisions or stopping, keeping the average speed close to the desired one. This was an expected result, since the proposed approach is a complex navigation system consisting of explicit modelling and control design. The two classical approaches provide good results at lower and average speeds, but at slightly higher speeds, the Potential Fields Method fails. The Dynamic Window Approach outperforms it by far, since it models the dynamics of the vehicle, and predicts its trajectories in the future. However, by lacking explicit modelling and advanced control design, the Dynamic Window Approach cannot deal with high speeds we set as desired.

As a navigation system for person following, the proposed approach is tested by following a jogger with a Robotnik Summit XL robot on various terrain types, in many different situations. The terrain types include asphalt, grass, gravel, macadam, snow and ice. During the experiments, many static and dynamic obstacles were on the way, such as other joggers, children, cars or dogs. In all the experiments, the robot was able to safely and smoothly avoid the obstacles and follow the tracked person. This system can also be employed to follow any dynamic object, such as another mobile robot, which would be interesting for multi-robot systems. The proposed approach is published in (Huskić *et al.*, 2017c). Further contributions in the field of person following, developed as a part of this thesis, are published in (Liu *et al.*, 2014) and (Liu *et al.*, 2015).

An alternative to the proposed approach would be a system with integrated trajectory generation and tracking, producing optimal trajectories, and using e.g. Model Predictive Control (MPC). Advantages of path following over trajectory tracking are already discussed in Chapter 2. Furthermore, techniques like MPC can be computationally expensive, while a fast person following and obstacle avoidance system needs a fast and lightweight system. The proposed approach is designed to be as simple as possible, in order to be as fast as possible, and allow robust high speed obstacle avoidance in real-life conditions. Furthermore, the proposed approach is designed to be modular, where the path planning is separated from the path following. In this way, the individual components can easily be exchanged and extended. This means that by using another path following control law, the complete approach can be used on e.g. an Ackermann-steering vehicle, while the rest of the system remains the same.

As a further result of studying path following control in general, twelve different con-

trol algorithms, together with the described local planning approach, have been implemented and integrated into one framework. The framework is written in C++/ROS and can be used by any wheeled mobile robot, regardless of the kinematic configuration. When controlling a mobile robot with the proposed framework, a generic control approach can be used, such as the one previously described and published in (Huskić *et al.*, 2016), or a specialized approach for the specific kinematic type. The framework also offers different path planning modes, and all the contributions previously mentioned. This means that the framework offers high speed control, as well as obstacle avoidance, in all-terrain conditions, both indoors and outdoors. The framework is an alternative to the well known ROS Navigation Stack, described in (Marder-Eppstein, 2017b), offers solutions to its drawbacks, and more features. The main drawbacks of the ROS Navigation Stack are that it is not supporting different kinematic types, it is only supporting planar laser scanners as sensors, and it supports only quadratic or circular robots. All of these drawbacks are eliminated in the proposed framework, offering more features, such as model-based controllers for each kinematic type. The simple control in ROS Navigation Stack does not consider the specific kinematic limitations, nor does it consider the effects important for high speed applications on uneven terrain. The proposed framework is published in (Huskić *et al.*, 2017a), focusing on the control aspects.

Since all contributions of the thesis are integrated into the proposed framework, and released as open-source software, the complete work is transparent and accessible to the research community. This allows the reproducibility of the results of both the contributed, as well as the implemented state-of-the-art algorithms.

The presented results answer some, yet open other questions, and can serve as a base for further research and rapid development of autonomous robotic systems.

7.1 Future Work

It would be reasonable to further investigate the dynamic control design presented in Chapter 4. Dynamic modelling of skid-steered vehicles can indeed be complex and tedious, yet at speeds even higher than tested in the presented experiments, it is probably necessary. The most important component of such a model would be the description of the interactions between the wheels and the ground, and the friction. The friction still represents a challenge when it comes to modelling, even though it is one of the basic physical quantities. Some basic work on modelling of skid-steered vehicles is presented in (Wong, 2001), and it can be taken as a starting point. Furthermore, modelling the effects acting upon the wheels can be done in a similar way as it is described in (Pacejka, 2005).

Depending on the dimensions of the vehicle, when driving on a very uneven terrain at even higher speeds, as in the case of the Segway RMP 440 robot presented in Chapter 4, a tilt control component would be necessary in order to keep the vehicle from tipping over.

Another component that could be investigated more is the kinematic model. While estimating the ICR parameters offline, and using constant values, offers a very robust approach, it would still be of great interest to investigate online estimation of ICR parameters. An online estimation of ICR parameters is already done in (Pentzer *et al.*, 2014a) by using an EKF, but this approach proved to be inadequate at higher speeds. It would be more reasonable to find an equivalence between the ICR model, and a model based on the wheel slip.

The approach described in Chapter 5 could be used at even higher speeds, and in that case, a special modelling of dynamic obstacles would be beneficiary. Modelling and predicting the behaviour of all the dynamic objects would lead to a safer navigation.

Bibliography

Aguiar, A. P. and Hespanha, J. P. (2007). Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty. *IEEE Transactions on Automatic Control*, **52**(8), 1362–1379.

Aguiar, A. P., Atassi, A. N., and Pascoal, A. M. (2000). Regulation of a nonholonomic dynamic wheeled mobile robot with parametric modeling uncertainty using lyapunov functions. In *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, volume 3, pages 2995–3000. IEEE.

Aguiar, A. P., Dačić, D. B., Hespanha, J. P., and Kokotović, P. (2004). Path-following or reference-tracking? *rn*, **1**, 1.

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

Brockett, R. W. *et al.* (1983). Asymptotic stability and feedback stabilization. *Differential geometric control theory*, **27**(1), 181–191.

Buck, S., Hanten, R., Huskić, G., Rauscher, G., Kloss, A., Leininger, J., Ruff, E., Widmaier, F., and Zell, A. (2015). Conclusions from an object-delivery robotic competition: Sick robot day 2014. In *Advanced Robotics (ICAR), 2015 International Conference on*, pages 137–143. IEEE.

Buck, S., Hanten, R., Bohlmann, K., and Zell, A. (2016a). Generic 3d obstacle detection for agvs using time-of-flight cameras. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 4119–4124. IEEE.

Buck, S., Hanten, R., Pech, C. R., and Zell, A. (2016b). Synchronous dataflow and visual programming for prototyping robotic algorithms. In *Intelligent Autonomous Systems (IAS-14)*. Springer.

Campion, G., Bastin, G., and D'Andrea-Novel, B. (1996). Structural properties and classification of kinematic and dynamic models of wheeled mobile robots. *IEEE transactions on robotics and automation*, **12**(1), 47.

Caracciolo, L., De Luca, A., and Iannitti, S. (1999). Trajectory tracking control of a four-wheel differentially driven mobile robot. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 4, pages 2632–2638. IEEE.

Cosgun, A., Florencio, D. A., and Christensen, H. I. (2013). Autonomous person following for telepresence robots. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4335–4342. IEEE.

Coulter, R. C. (1992). Implementation of the pure pursuit path tracking algorithm. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA ROBOTICS INST.

De Luca, A., Oriolo, G., and Samson, C. (1998). Feedback control of a nonholonomic car-like robot. In *Robot motion planning and control*, pages 171–253. Springer.

de Wit, C. C., Khennouf, H., Samson, C., and Sordalen, O. (1993.). Nonlinear control design for mobile robots. In *Recent Developments in Mobile Robots*. World Scientific.

Derametz, M., Lenain, R., Thuilot, B., and Rousseau, V. (2017). Adaptive trajectory control of off-road mobile robots: A multi-model observer approach. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 4407–4413. IEEE.

Dolgov, D., Thrun, S., Montemerlo, M., and Diebel, J. (2008). Practical search techniques in path planning for autonomous driving. *Ann Arbor*, **1001**, 48105.

Egerstedt, M., Hu, X., and Stotsky, A. (2001). Control of mobile platforms using a virtual vehicle approach. *Automatic Control, IEEE Transactions on*, **46**(11), 1777–1782.

Elbanhawi, M., Simic, M., and Jazar, R. (2016). Receding horizon lateral vehicle control for pure pursuit path tracking. *Journal of Vibration and Control*, page 1077546316646906.

Fierro, R. and Lewis, F. L. (1995). Control of a nonholonomic mobile robot: backstepping kinematics into dynamics. In *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*, volume 4, pages 3805–3810. IEEE.

Fox, D., Burgard, W., and Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, **4**(1), 23–33.

Garzon Oviedo, M. A., Barrientos, A., Del Cerro, J., Alacid, A., Fotiadis, E., Rodríguez-Canosa, G. R., and Wang, B.-C. (2015). Tracking and following pedestrian trajectories, an approach for autonomous surveillance of critical infrastructures. *Industrial Robot: An International Journal*, **42**(5), 429–440.

Ghommam, J., Mehrjerdi, H., Saad, M., and Mnif, F. (2010). Formation path following control of unicycle-type mobile robots. *Robotics and Autonomous Systems*, **58**(5), 727–736.

Godhavn, J.-M. and Egeland, O. (1997). A lyapunov approach to exponential stabilization of nonholonomic systems in power form. *IEEE Transactions on Automatic Control*, **42**(7), 1028–1032.

Hart, P. E., Nilsson, N. J., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, **4**(2), 100–107.

Hespanha, J. P. and Morse, A. S. (1999). Stabilization of nonholonomic integrators via logic-based switching. *Automatica*, **35**(3), 385–393.

Hoffmann, G. M., Tomlin, C. J., Montemerlo, M., and Thrun, S. (2007). Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing. In *American Control Conference, 2007. ACC'07*, pages 2296–2301. IEEE.

Huskić, G., Buck, S., and Zell, A. (2016). A simple and effcient path following algorithm for wheeled mobile robots. In *Intelligent Autonomous Systems (IAS-14)*. Springer.

Huskić, G., Buck, S., and Zell, A. (2017a). Gerona: Generic robot navigation; a modular framework for robot navigation and control. *Journal of Intelligent & Robotic Systems*. (submitted).

Huskić, G., Buck, S., Herrb, M., Lacroix, S., and Zell, A. (2017b). High-speed path following control of skid-steered vehicles. *International Journal of Robotics Research*. (submitted).

Huskić, G., Buck, S., Ibragüen González, L. A., and Zell, A. (2017c). Outdoor person following at higher speeds using a skid-steered mobile robot. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE.

Huskić, G., Buck, S., and Zell, A. (2017d). Path following control of skid-steered wheeled mobile robots at higher speeds on different terrain types. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3734–3739. IEEE.

Iagnemma, K. and Dubowsky, S. (2004). *Mobile robots in rough terrain: Estimation, motion planning, and control with application to planetary rovers*, volume 12. Springer Science & Business Media.

Indiveri, G., Nüchter, A., and Lingemann, K. (2007). High speed differential drive mobile robot path following control with bounded wheel speed commands. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 2202–2207. IEEE.

Jiang, Z.-P. and Nijmeijer, H. (1999). A recursive technique for tracking control of nonholonomic systems in chained form. *IEEE Transactions on Automatic control*, **44**(2), 265–279.

Kaminer, I., Pascoal, A., Hallberg, E., and Silvestre, C. (1998). Trajectory tracking for autonomous vehicles: An integrated approach to guidance and control. *Journal of Guidance, Control, and Dynamics*, **21**(1), 29–38.

Kanayama, Y., Kimura, Y., Miyazaki, F., and Noguchi, T. (1990). A stable tracking control method for an autonomous mobile robot. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 384–389. IEEE.

Kanjanawanishkul, K. and Zell, A. (2009). Path following for an omnidirectional mobile robot based on model predictive control. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 3341–3346. IEEE.

Khalil, H. K. and Grizzle, J. (1996). *Nonlinear systems*, volume 3. Prentice hall New Jersey.

Koren, Y. and Borenstein, J. (1991). Potential field methods and their inherent limitations for mobile robot navigation. In *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pages 1398–1404. IEEE.

Kozłowski, K. and Pazderski, D. (2004). Modeling and control of a 4-wheel skid-steering mobile robot. *Int. J. Appl. Math. Comput. Sci*, **14**(4), 477–496.

Kozłowski, K. and Pazderski, D. (2006). Practical stabilization of a skid-steering mobile robot-a kinematic-based approach. In *Proc. IEEE 3rd Int. Conf. on Mechatronics*, pages 519–524.

Kronfeld, M., Planatscher, H., and Zell, A. (2010). The eva2 optimization framework. In *International Conference on Learning and Intelligent Optimization*, pages 247–250. Springer.

Lapierre, L., Soetanto, D., and Pascoal, A. (2006). Nonsingular path following control of a unicycle in the presence of parametric modelling uncertainties. *International Journal of Robust and Nonlinear Control*, **16**(10), 485–503.

Leigh, A., Pineau, J., Olmedo, N., and Zhang, H. (2015). Person tracking and following with 2d laser scanners. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 726–733. IEEE.

Lenain, R., Thuilot, B., Cariou, C., and Martinet, P. (2006). High accuracy path tracking for vehicles in presence of sliding: Application to farm vehicle automatic guidance for agricultural tasks. *Autonomous robots*, **21**(1), 79–97.

Lenain, R., Lucet, E., Grand, C., Thuilot, B., and Amar, F. B. (2010). Accurate and stable mobile robot path tracking: An integrated solution for off-road and high speed context. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 196–201. IEEE.

Lenain, R., Thuilot, B., Hach, O., and Martinet, P. (2011). High-speed mobile robot control in off-road conditions: a multi-model based adaptive approach. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 6143–6149. IEEE.

Lenain, R., Thuilot, B., Guillet, A., and Benet, B. (2014). Accurate target tracking control for a mobile robot: A robust adaptive approach for off-road motion. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2652–2657. IEEE.

Li, X. (2009). *Dribbling control of an omnidirectional soccer robot*. Ph.D. thesis, Tübingen, Univ., Diss., 2009.

Li, X., Wang, M., and Zell, A. (2007). Dribbling control of omnidirectional soccer robots. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 2623–2628. IEEE.

Liu, R., Huskić, G., and Zell, A. (2014). Dynamic objects tracking with a mobile robot using passive uhf rfid tags. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 4247–4252. IEEE.

Liu, R., Huskić, G., and Zell, A. (2015). On tracking dynamic objects with long range passive uhf rfid using a mobile robot. *International Journal of Distributed Sensor Networks*.

Lucet, E., Grand, C., Sallé, D., and Bidaud, P. (2008). Dynamic sliding mode control of a four-wheel skid-steering vehicle in presence of sliding. *Proc. RoManSy, Tokyo, Japan*.

Lucet, E., Grand, C., Sallé, D., and Bidaud, P. (2009). Dynamic velocity and yaw-rate control of the 6wd skid-steering mobile robot roburoc6 using sliding mode technique. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, Missouri, USA*.

Mandow, A., Martinez, J. L., Morales, J., Blanco, J. L., Garcia-Cerezo, A., and Gonzalez, J. (2007). Experimental kinematics for wheeled skid-steer mobile robots. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 1222–1227. IEEE.

Marder-Eppstein, E. (2017a). Dynamic window approach available open-source in ros. Accessed: 2017-02-21.

Marder-Eppstein, E. (2017b). Ros navigation stack - a 2d navigation package that controls a mobile base. Accessed: 2017-06-09.

Martínez, J. L., Mandow, A., Morales, J., Pedraza, S., and García-Cerezo, A. (2005). Approximating kinematics for tracked mobile robots. *The International Journal of Robotics Research*, **24**(10), 867–878.

Martins, F. N., Sarcinelli-Filho, M., and Carelli, R. (2017). A velocity-based dynamic model and its properties for differential drive mobile robots. *Journal of Intelligent & Robotic Systems*, **85**(2), 277–292.

Maček, K., Petrović, I., and Siegwart, R. (2005). A control method for stable and smooth path following of mobile robots. In *Proceedings of the European Conference on Mobile Robots*.

Micaelli, A. and Samson, C. (1993). Trajectory tracking for unicycle-type and two-steering-wheels mobile robots.

Miller, L. M. and Murphey, T. D. (2012). Simultaneous optimal parameter and mode transition time estimation. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 719–724. IEEE.

Mojaev, A. and Zell, A. (2004). Tracking control and adaptive local navigation for nonholonomic mobile robot. In *Intelligent Autonomous Systems (IAS-8)*, pages 521–528, Amsterdam, Netherlands. IOS Press.

Morin, P. and Samson, C. (2008). Motion control of wheeled mobile robots. In *Springer Handbook of Robotics*, pages 799–826. Springer.

Morro, A., Sgorbissa, A., and Zaccaria, R. (2011). Path following for unicycle robots with an arbitrary path curvature. *IEEE Transactions on Robotics*, **27**(5), 1016–1023.

Nash, A., Daniel, K., Koenig, S., and Felner, A. (2007). Theta^*: Any-angle path planning on grids. In *AAAI*, pages 1177–1183.

Nizard, A., Thuilot, B., Lenain, R., and Mezouar, Y. (2016). Nonlinear path tracking controller for bi-steerable vehicles in cluttered environments. *IFAC-PapersOnLine*, **49**(15), 19–24.

Oftadeh, R., Ghabcheloo, R., and Mattila, J. (2014). Time optimal path following with bounded velocities and accelerations for mobile robots with independently steerable wheels. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 2925–2931. IEEE.

Olmedo, N. A., Zhang, H., and Lipsett, M. (2014). Mobile robot system architecture for people tracking and following applications. In *Robotics and Biomimetics (ROBIO), 2014 IEEE International Conference on*, pages 825–830. IEEE.

Ostafew, C. J., Schoellig, A. P., and Barfoot, T. D. (2016). Robust constrained learning-based nmpc enabling reliable mobile robot path tracking. *The International Journal of Robotics Research*, **35**(13), 1547–1563.

Pacejka, H. (2005). *Tire and vehicle dynamics*. Elsevier.

Park, J. J. and Kuipers, B. (2013). Autonomous person pacing and following with model predictive equilibrium point control. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1060–1067. IEEE.

Pentzer, J., Brennan, S., and Reichard, K. (2014a). Model-based prediction of skid-steer robot kinematics using online estimation of track instantaneous centers of rotation. *Journal of Field Robotics*, **31**(3), 455–476.

Pentzer, J., Brennan, S., and Reichard, K. (2014b). The use of unicycle robot control strategies for skid-steer robots through the icr kinematic mapping. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 3201–3206. IEEE.

PiaggioFastForward (2017). Gita - A person following cargo robot from the Italian motorcycle manufacturer Piaggio. Accessed: 2017-02-21.

Płaskonka, J. (2015). Different kinematic path following controllers for a wheeled mobile robot of (2, 0) type. *Journal of Intelligent & Robotic Systems*, **77**(3-4), 481.

Pucci, D., Marchetti, L., and Morin, P. (2013). Nonlinear control of unicycle-like robots for person following. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 3406–3411. IEEE.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5.

Rajagopalan, V., Kelly, A., *et al.* (2016). Slip-aware model predictive optimal control for path following. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4585–4590. IEEE.

Samson, C. (1992). Path following and time-varying feedback stabilization of a wheeled mobile robot. In *Int. Conf. ICARCV'92*.

Samson, C. (1995). Control of chained systems application to path following and time-varying point-stabilization of mobile robots. *Automatic Control, IEEE Transactions on*, **40**(1), 64–77.

Shiller, Z., Sharma, S., Stern, I., and Stern, A. (2013). Online obstacle avoidance at high speeds. *The International Journal of Robotics Research*, **32**(9-10), 1030–1047.

Shimoda, S., Kuroda, Y., and Iagnemma, K. (2007). High-speed navigation of unmanned ground vehicles on uneven terrain using potential fields. *Robotica*, **25**(4), 409–424.

Siegwart, R., Nourbakhsh, I. R., and Scaramuzza, D. (2004). Autonomous mobile robots. *Massachusetts Institute of Technology*.

Slotine, J.-J. E., Li, W., *et al.* (1991). *Applied nonlinear control*, volume 199. Prentice hall Englewood Cliffs, NJ.

Snider, J. M. *et al.* (2009). Automatic steering methods for autonomous automobile path tracking. *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RITR-09-08*.

Soetanto, D., Lapierre, L., and Pascoal, A. (2003). Adaptive, non-singular path-following control of dynamic wheeled robots. In *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, volume 2, pages 1765–1770. IEEE.

Spenko, M., Iagnemma, K., and Dubowsky, S. (2004). High speed hazard avoidance for mobile robots in rough terrain. In *Proceedings of the SPIE Conference on Unmanned Ground Vehicles*, pages 439–450.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., *et al.* (2006). Stanley: The robot that won the DARPA grand challenge. *Journal of field Robotics*, **23**(9), 661–692.

Toponogov, V. A. (2006). *Differential geometry of curves and surfaces*. Springer.

Walsh, G., Tilbury, D., Sastry, S., Murray, R., and Laumond, J.-P. (1994). Stabilization of trajectories for systems with nonholonomic constraints. *IEEE Transactions on Automatic Control*, **39**(1), 216–222.

Williams, G., Drews, P., Goldfain, B., Rehg, J. M., and Theodorou, E. A. (2016). Aggressive driving with model predictive path integral control. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1433–1440. IEEE.

Williams, G., Wagener, N., Goldfain, B., Drews, P., Rehg, J. M., Boots, B., and Theodorou, E. A. (2017). Information theoretic MPC for model-based reinforcement learning. In *International Conference on Robotics and Automation (ICRA)*.

Wong, J. Y. (2001). *Theory of ground vehicles*. John Wiley & Sons.

Xin, M. and Minor, M. (2012). Backstepping vehicle steering controller using integral and robust control based on dynamic state estimation. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 3132–3137. IEEE.

Yi, J., Song, D., Zhang, J., and Goodwin, Z. (2007). Adaptive trajectory tracking control of skid-steered mobile robots. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 2605–2610. IEEE.

Yi, J., Wang, H., Zhang, J., Song, D., Jayasuriya, S., and Liu, J. (2009). Kinematic modeling and analysis of skid-steered mobile robots with applications to low-cost inertial-measurement-unit-based motion estimation. *Robotics, IEEE Transactions on*, **25**(5), 1087–1097.

Yu, W., Chuy Jr, O. Y., Collins Jr, E. G., and Hollis, P. (2010). Analysis and experimental verification for dynamic modeling of a skid-steered wheeled vehicle. *IEEE transactions on robotics*, **26**(2), 340–353.