ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

RSL Robotic Systems Lab
Prof. Dr. Marco Hutter

Master Thesis

Load Sharing - Obstacle Avoidance and Admittance Control on a Mobile Manipulator

Spring Term 2018

Supervised by:
Professor Jonathan Kelly
Professor Marco Hutter

Author:
Tobias Ulrich

Declaration of Originality

I hereby declare that the written work I have submitted entitled

Your Project Title

is original work which I alone have authored and which is written in my own words.[1]

Author(s)

Tobias	Ulrich

Student supervisor(s)

Jonathan	Kelly

Supervising lecturer

Marco	Hutter

With the signature I declare that I have been informed regarding normal academic citation rules and that I have read and understood the information on 'Citation etiquette' (`https://www.ethz.ch/content/dam/ethz/main/education/rechtliches-abschluesse/leistungskontrollen/plagiarism-citationetiquette.pdf`). The citation conventions usual to the discipline in question here have been respected.

The above written work may be tested electronically for plagiarism.

Place and date	Signature

[1] Co-authored work: The signatures of all authors are required. Each signature attests to the originality of the entire piece of written work in its final form.

Intellectual Property Agreement

The student acted under the supervision of Prof. Hutter and contributed to research of his group. Research results of students outside the scope of an employment contract with ETH Zurich belong to the students themselves. The results of the student within the present thesis shall be exploited by ETH Zurich, possibly together with results of other contributors in the same field. To facilitate and to enable a common exploitation of all combined research results, the student hereby assigns his rights to the research results to ETH Zurich. In exchange, the student shall be treated like an employee of ETH Zurich with respect to any income generated due to the research results.
This agreement regulates the rights to the created research results.

1. Intellectual Property Rights

1. The student assigns his/her rights to the research results, including inventions and works protected by copyright, but not including his moral rights ("Urheberpersönlichkeitsrechte"), to ETH Zurich. Herewith, he cedes, in particular, all rights for commercial exploitations of research results to ETH Zurich. He is doing this voluntarily and with full awareness, in order to facilitate the commercial exploitation of the created Research Results. The student's moral rights ("Urheberpersönlichkeitsrechte") shall not be affected by this assignment.
2. In exchange, the student will be compensated by ETH Zurich in the case of income through the commercial exploitation of research results. Compensation will be made as if the student was an employee of ETH Zurich and according to the guidelines "Richtlinien für die wirtschaftliche Verwertung von Forschungsergebnissen der ETH Zürich".
3. The student agrees to keep all research results confidential. This obligation to confidentiality shall persist until he or she is informed by ETH Zurich that the intellectual property rights to the research results have been protected through patent applications or other adequate measures or that no protection is sought, but not longer than 12 months after the collaborator has signed this agreement.
4. If a patent application is filed for an invention based on the research results, the student will duly provide all necessary signatures. He/she also agrees to be available whenever his aid is necessary in the course of the patent application process, e.g. to respond to questions of patent examiners or the like.

2. Settlement of Disagreements

Should disagreements arise out between the parties, the parties will make an effort to settle them between them in good faith. In case of failure of these agreements, Swiss Law shall be applied and the Courts of Zurich shall have exclusive jurisdiction.

Place and date	Signature

Contents

Preface

Bla bla ...

Abstract

We present an implementation of a load sharing algorithm between a human and a robot, designed to work in an indoor cluttered environment. Human robot cooperation is applied, using information exchange through forces and torque applied to the jointly handled object. The work is set within the master-slave paradigm and combines an admittance controller with obstacle avoidance to ensure collision free trajectories at all times. We derive the implementation from existing literature and validate the working algorithm on a mobile manipulator, consisting of a Clearpath Ridgeback platform, a Universal Robot 10 and a Robotiq three finger gripper.

Symbols

Symbols

ϕ, θ, ψ	roll, pitch and yaw angle
b	gyroscope bias
Ω_m	3-axis gyroscope measurement

Indices

x	x axis
y	y axis

Acronyms and Abbreviations

ETH	Eidgenössische Technische Hochschule
EKF	Extended Kalman Filter
IMU	Inertial Measurement Unit
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter

Chapter 1

Introduction

As with many fields in robotics, Human Robot Interaction (HRI) has seen a lot of development in the last twenty years. Research has come from tele-operated assistive robots to dynamically and independently collaborating robots. This advancement is expressed by the newly joined terms and taxonomies in literature to differentiate between types of motor interactions [1] and general interactions [2], expressing different levels of autonomy for the robot.
HRI is defined as the integration of human actions into a robot's decision making process and Figure 1.1 outlines a recent taxonomy [2], which subdivides HRI in three categories, based on the level of interdependence between the human and robot partners.

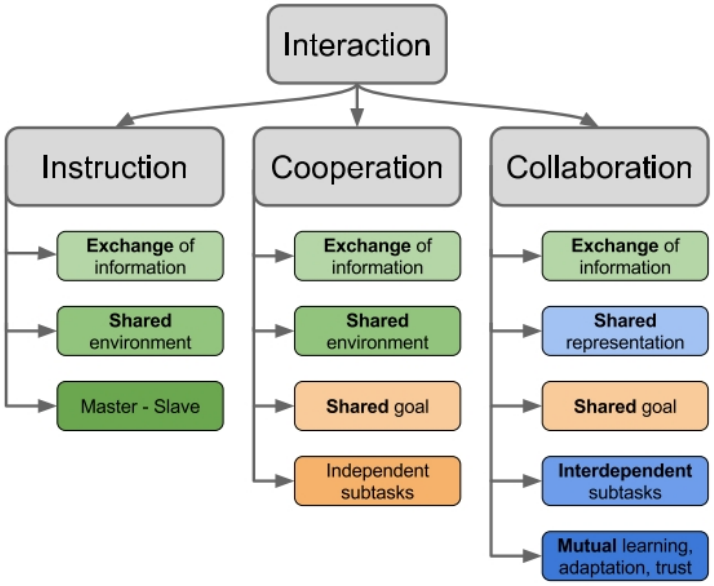


Figure 1.1: Taxonomy of human robot interactions (HRI) [2]. Three categories are defined, namely instruction, cooperation and collaboration, based on ascending level of interdependence between the human and the robot partners.

Instruction On the lowest level of interdependence, instruction groups types of interactions, which are a sequence of actions governed by the human. Although there is exchange of information between the two partners and they are sharing an environment, the robot is subordinated to the human in a

master-slave paradigm and does not independently try to reach a goal, but merely assists the human in reaching his. A common example is learning from demonstration [3].

Cooperation This concept is expanded in cooperation tasks to a shared goal, where the goal is divisible in subtasks, on which the partners can work independently. Here, roles are fixed but do not necessarily have to be in a master-slave scenario.

Collaboration Finally, collaboration groups tasks that require a shared representation of the environment and have subtasks, where the partners are interdependent on each other, i.e., the partners actively engage in the activities of the other. Here, an equal role distribution or role switching between the two is crucial for efficiency and to reduce the cognitive workload for the human.

This project is set within the field of *cooperation*. The goal is to achieve a load sharing algorithm between a robot and a human partner in a cluttered environment. The cooperation comes with several advantages, for both partners. For one, a human physically cooperating with a robotic system can greatly enhance its capabilities and act as a guide or a second control mechanism. In exchange, the robot can take physical load off the human and also reduce the cognitive load, if it is pro-actively working towards the shared goal. Finally, the actuation redundancy of any two partners carrying an object, allows for greater flexibility and improved navigation through narrow passages over a single agent carrying the same object [4].
As with any joint carrying task in either human human interaction (HHI) or HRI, the richest sensory exchange channel between the two partners is the haptic feedback through physical coupling of the jointly carried object [5]. Therefore, it is crucial to include the force and torque input from the human into the control loop of the robot. Similarly, a robot which is actively contributing to the shared goal by its obstacle avoidance, although being bound in a fixed subordinate role to the human, will greatly enhance the system.
In the following chapters, we will first analyse other work that has been done in the field of load sharing, then derive the necessary theory for our algorithm before we elaborate on our implementation and finally validate the algorithm in real world scenarios.

Chapter 2

Related Work

There are a lot of fields of human-robot motor interaction, which can come into play when talking about load sharing, because the task consists of numerous subtasks, such as haptic interaction, obstacle avoidance, path planning, effort and load sharing strategies, role distribution, signaling and intent prediction. Accordingly, there is a vast amount of literature available, each tackling individual subtasks.
In [8] an asymmetric master-slave cooperation in a load sharing scenario with effort sharing policies is achieved. Using the actuation redundancy, they vary the degree of robot assistance in the direction of the trajectory and show an interdependency of the load sharing strategy and the resulting task performance. Figure 2.1.
In [**?**], an extensive study of human-humanoid carrying and a complete control framework is presented.
Research with role switching has been conducted amongst others in [6], where a homotopy switching model between the human and the robot partner allows for continuous role switching between leader and follower.
Recent research focusing on the admittance control can be found in [7], where two drones cooperate in a fixed master slave scenario and make use exclusively of the haptic interaction information channel.

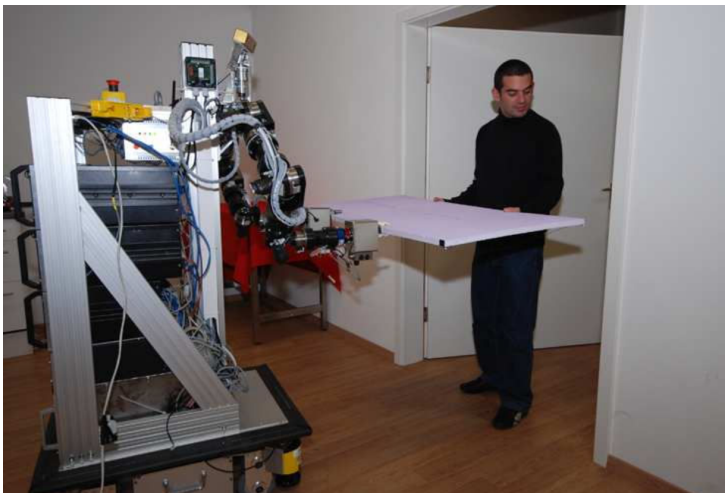


Figure 2.1: Load sharing example [8].

Chapter 3

The Thing Mobile Manipulator

We conduct our research on a mobile manipulator, lovingly called the *Thing* and depicted in Figure 3.1. It is composed of three main components and various sensing technologies, on which we elaborate in detail in this chapter. The three components, from bottom to top, are a Ridgeback omnidirectional robot platform [9], a UR10 robotic arm [10] and a Robotiq three finger gripper [11].

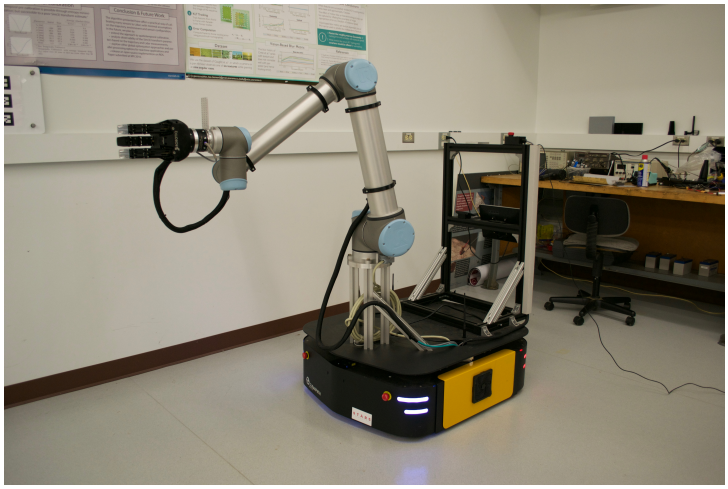


Figure 3.1: The Thing mobile manipulator, which consists of a Ridgeback omnidirectional platform, a UR10 and a Robotic gripper.

These hardware packages are essentially independent and are interfaced in a central computing unit, located in the ridgeback. The interface is achieved using a Robot Operating System (ROS) server, which also acts as the user interface. It allows for information sharing between the system and access for the user on high and low system levels.
The rich and easily expandable sensing equipments include wheel odometry, an Inertial Measurement Unit (IMU) and a Hokuyo laser range finder (LIDAR) [12] in the Ridgeback and a FT 300 force torque sensor [13] in between the gripper and the UR10.
The omnidirectional capability together with the modularity of the system and the vast amount of sensor arrays make the Thing mobile manipulator a popular and widely used system for autonomous robotics research.

The whole manipulator is an out of the box system assembled by Clearpath, which collaborates with Universal Robots and Robotiq and mounts the parts on the platform in house.

3.1 Clearpath Ridgeback

The Ridgeback, depicted in Figure 3.2 is an omnidirectional robot platform designed by Clearpath for indoor movement and payload carrying tasks, such as autonomous warehousing for example. It is a fully integrated system with sensors, actuation and control and features a native ROS interface. The wheel odometry provides accurate localization and can be fused with the front facing LIDAR and IMU for obstacle avoidance and SLAM implementations, which are implemented for example in the ROS navigation stack. Optionally, a second rear facing LIDAR can be mounted for full 360 ° coverage. Specifications are summarized in Table 3.1.

Figure 3.2: Clearpath Ridgeback

Table 3.1: Clearpath Ridgeback Specifications

Length	960 mm
Width	793 mm
Height	296 mm
Weight	135 kg
Maximum payload	100 kg
Maximum velocity	1.1 $^{m}/_{s}$
Average power consumption	800 W
Drivers and APIs	ROS
Sensors	Wheel odometry
	Inertial measurement unit (IMU)
	Laser range finder LIDAR

The broad range of sensors, it's flexibility and low drift in odometry makes the Ridgeback a suitable and popular platform for research in controlled indoor environments.

Because the Ridgeback is omnidirectional, it has three degrees of freedom (DOF) and can move instantaneously along the two linear horizontal axes x and y, as well as rotate around the vertical axis z, which is expressed by the yaw angle θ.
Additionally, the Ridgeback houses the on-board computer that runs the low-level drivers of all the elements of the manipulator. On top thereof, there is a high-level driver that ensures accord and offers a ROS interface for the user to connect to.

3.2 Universal Robot 10

The UR10 is a collaborative industrial robot arm by Universal Robots. It has six rotary joints, hence six DOF and can support payloads up to 10 kg. Together with it's little brother the UR5, it is widely regarded as the standard manipulator within robotics research. Hence, extensive platform and software integration resources are available and ROS is supported out of the box.
As we see in Table 3.2, the repeatability of 0.1 mm allows for repeatedly achieving a pose with high accuracy, which is important for industrial and research applications alike.

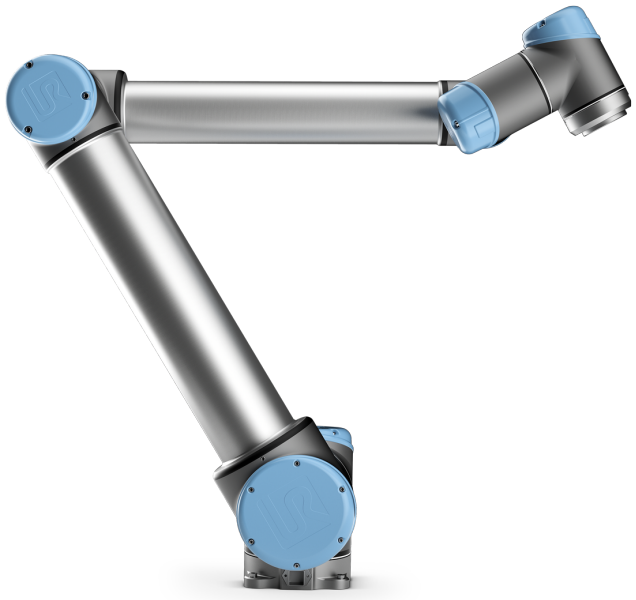


Figure 3.3: Universal Robot 10

Table 3.2: Universal Robot 10 Specifications

Reach	1300 mm
Weight	1.5 kg
Repeatability	0.1 mm
Maximum payload	10 kg
Maximum tool velocity	1 $^{m}/_{s}$
Degrees of freedom	6 rotating joints
Average power consumption	350 W

3.3 Robotiq Three Finger Gripper

The Robotiq three finger adaptive gripper, depicted in Figure 3.4, is the end-effector mounted on the UR10 on the Thing. As specified in Table 3.3, it has four different grip types and allows for force, position and speed control for each finger individually. This makes it suitable for manufacturing, assembly, pick and place and other research applications.

Figure 3.4: Robotiq three finger adaptive robot gripper

Table 3.3: Robotiq 3-Finger Adaptive Robot Gripper Specifications

Weight	2.3 kg
Repeatability	0.1 mm
Maximum payload (encompassing grip)	10 kg
Gripper opening	0 to 155 mm
Object diameter for encompassing	20 to 155 mm
Grip force	30 to 70 N
Grip types	Pinch mode
	Wide mode
	Scissor mode
	Basic mode
Minimum power consumption	4.1 W
Peak power (at maximum gripping force)	36 W

3.4 Robotiq FT 300 Force-Torque Sensor

The FT 300 is a force torque sensor by Robotiq, embedded in a cylindrical and stiff casing. As Figure 3.5 shows, the sensor is compatible with any Robotiq gripper and Universal Robots arm and acts as an intermediate module between the gripper and the end-effector of the robotic arm. Hence, any force or torque which is transferred from gripper to the arm or vice versa can be measured. The sensor measures forces along and torques around the three axis x, y and z of the module, makint it a six dimensional sensor, outputting data at 100 Hz. In Table 3.4, the measuring ranges and signal noises are listed.

Figure 3.5: Robotiq FT 300 Force Torque Sensor

The FT-300 allows for any tasks that make use of force feedback or force control, such as assembly, finishing, insertion, pick and place or teach and repeat applications.

Table 3.4: Robotiq FT 300 Force Torque Sensor Specifications

Measuring range	
Force F_x, F_y, F_z	$\pm 300\,\mathrm{N}$
Moment M_x, M_y, M_z	$\pm 30\,\mathrm{Nm}$
Signal noise[1]	
Force F_x, F_y, F_z	$0.1\,\mathrm{N}$ / $1\,\mathrm{N}$
Moment M_x, M_y	$0.05\,\mathrm{Nm}$ / $0.02\,\mathrm{Nm}$
Moment M_z	$0.03\,\mathrm{Nm}$ / $0.01\,\mathrm{Nm}$
Data output rate	$100\,\mathrm{Hz}$
Weight	$300\,\mathrm{g}$
Outside diameter	$75\,\mathrm{mm}$
Thickness	$37.5\,\mathrm{mm}$

3.5 Hokuyo Laser Range Finder

Figure 3.6 depicts the Hokuyo UST-10LX scanning laser range finder. It scans a single horizontal plane by spinning around its vertical axis at 2400 rpm, which results in a scanning frequency of 40 Hz, also shown in Table 3.5. Compared to any LIDAR system that stacks multiple lasers on top of each other, the Hokuyo UST-10LX is in a disadvantage, because it only outputs two-dimensional range measurements and supplies no information along its vertical axis. However, it outperforms many alternatives in terms of size, weight, power consumption, accuracy and measurement frequency, which makes it highly suitable for a mobile platform.

[1] Signal noise is the standard deviation of the signal measured over a period of one second.

Figure 3.6: Hokuyo UST-10LX laser range finder

Table 3.5: Hokuyo UST-10LX Laser Range Finder Specifications

Scan angle	250 °
Measurement steps	1081
Angular resolution	0.25 °
Detection range	0.06 to 10 m
Scan frequency	40 Hz
Average power consumption	3 W
Weight	130 g
Length	50 mm
Width	50 mm
Height	70 mm

Chapter 4

Admittance Control

The term *admittance control* is closely coupled to the term *impedance control* and essentially they refer to two ways of implementing the same control goal. The goal is to model the dynamic coupling between the end-effector and the environment, which is achieved by establishing a dynamic relationship between the position and force of the end-effector [14].
By definition, an impedance is a physical system, that accepts motion inputs and yields force outputs, whereas a system that accepts force inputs and yields motion outputs is referred to as an admittance [**?**].
Hence in impedance control, the controller acts as the impedance and the end-effector as the admittance, which leads to a stable dynamic interaction with stiff environments but very often to instability in free space. Alternatively in admittance control, the controller acts as the admittance and the end-effector as the impedance. Mathematically, the interaction between force input and position output is modelled as a second order spring mass damper system, as depicted in Figure 4.1. According to Newton's second law of motion, the change in momentum is equal to the sum of all forces F acting on our rigid body

$$M \cdot \ddot{X} = \sum F \tag{4.1}$$

where M is the mass and $\ddot{X}$ the acceleration of the rigid body.
The forces in the case of the mass spring damper system become apparent if we free-cut our system, as seen in Figure 4.2.

$$\sum F = F_D + F_K + F_{ext} \tag{4.2}$$

Which consists of a spring term with spring constant K,

$$F_K = K \cdot X \tag{4.3}$$

a damping term with damping constant D

$$F_D = D \cdot \dot{X} \tag{4.4}$$

and a term $F_e xt$ for external forces acting on the body. Replacing $\sum F$ in Equation 4.1 with Equation 4.2 and dividing by M yields the equation for a second order mass spring damper system:

$$M \cdot \ddot{X} = F_{ext} - D \cdot \dot{X} - K \cdot X \tag{4.5}$$

We can transform Equation 4.5 into state-space representation using

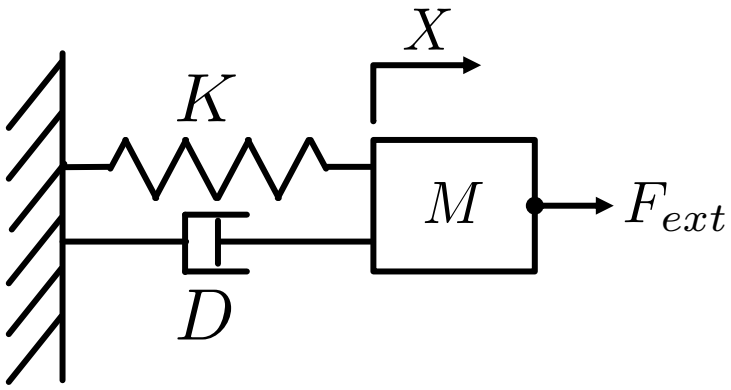


Figure 4.1: Second order mass spring damper system.

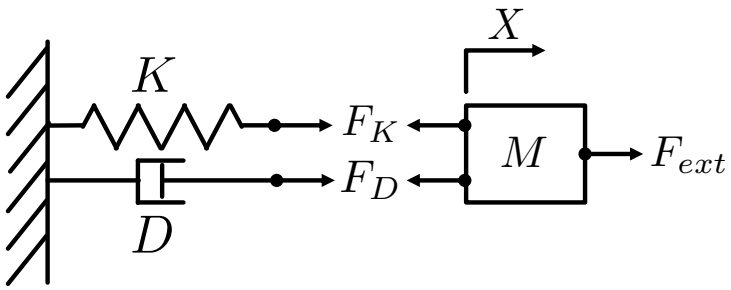


Figure 4.2: Free-cut second order mass spring damper system.

$$Z = \dot{X} \tag{4.6}$$

and we get:

$$M \cdot \dot{Z} = F_{ext} - D \cdot Z - K \cdot X \tag{4.7}$$

Written in matrix form, Equation 4.7 becomes

$$\begin{bmatrix} \dot{X} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{K}{M} & -\frac{D}{M} \end{bmatrix} \begin{bmatrix} X \\ Z \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{F_{ext}}{M} \end{bmatrix} \tag{4.8}$$

Chapter 5

Obstacle Avoidance

Ever since robots faced the task of autonomous navigation, obstacle avoidance has been a crucial element of it. There are numerous approaches to tackle the problem, varying in degrees of foresight and influence on the path planner.
The problem can be seperated into two categories, path planning around obstacles on a global scale and on a local scale. The first category bundles algorithms that take a goal position and a current position of a robot and calculate the optimal path (usually the shortest path) in between, given some objective function and a map. Since the distance to the goal is normally greater than the range of any obstacle detection sensor, these algorithms need a full map of the environment. Widely used examples are A*, Dijkstra and RRT.
In contrast, path planning on a local scale takes obstacle detection sensor information as an input and outputs commands to a drive unit, that meet the given objective, which is usually to avoid collisions and stay clear of an obstacle by a minimum distance.
A typical path planning infastructure on a robot consists of both a global and a local planner, where the global planner outputs a path to follow to the local planner, which in terms fuses that path with online obstacle detection sensor information to ensure it is indeed collision free and deviates from it if necessary.
To find the path planning algorithm that best meets our needs, we must first examine what are the given inputs and the desired outputs of our system. As we already elaborated in chapter 1, we are working in a master-slave scenario, which means that the robot is trying to achieve the goal of its human counterpart and not its own. This manifests in the planner in such a way, that the input is the force and torque applied by the human and the robot has no global goal pose. Hence, we are inherently bound to iteratively updated goals within close proximity and there is no possibility nor need to apply global path planning techniqes and we focus only on local planners in the remainder of this chapter.

5.1 Local Planners

In this chapter, we discuss a selection of common local planning methods and their feasibility for the task at hand.

5.1.1 Dynamic Window Approach

The Dynamic Window Approach (DWA) [15] is a well-known algorithm that produces command velocities for a planar robot given vehicle dynamics and obstacle measurements. The basic assumption is that the robot moves instantaneously on

circular arcs with a translational velocity v and a rotational velocity ω. Thus, the complexity is greatly simplified and calculations are be performed in the 2D velocity space (v, ω). Within this space, we compute three sets of velocity pairs, subsequently called *windows* for every iteration of the algorithm. Figure 5.1 provides a visual example of the functionality of the algorithm.

Static window The static window V_s expresses the constraint velocities of the vehicle, i.e., absolute maximum and minimum velocity. As the name suggests, these parameters are usually static and do not need to be recalculated every step.

Obstacle window The obstacle window V_o are the measurements of any obstacles, e.g. taken by a range laser sensor and transformed from Cartesian to v, ω space.

Dynamic window The dynamic window V_d are the vehicle dynamics, i.e., velocities that are physically feasible for the robot to reach within one time step. It's size is defined by the maximal acceleration and the current velocity of the robot.

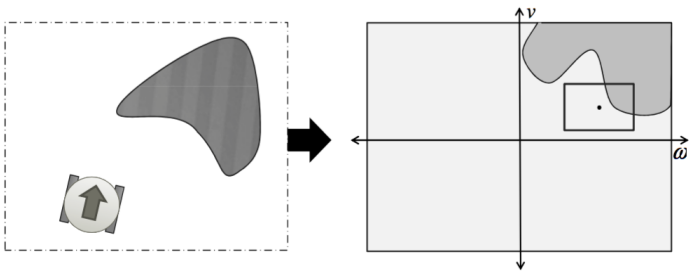


Figure 5.1: DWA working principle [16]. *Left:* Example of a planar robot and one obstacle in cartesian space. *Right:* Representation of the same scenario in (v, ω) velocity space. The large bounding box depicts the static window V_s, the dark grey area depicts the obstacle window V_o, the small bounding box depicts the dynamic window V_d and the black dot represents the current velocity pair of the robot. The resulting window V_r is then the intersection of all three areas.

$$V_r = V_o \cap V_s \cap V_d \tag{5.1}$$

As eq. (5.1) shows, the intersection of these three sets gives us the resulting window V_r of feasible velocity pairs, that guarantee no collision with an obstacle for the next step.
A cost function is then applied to find the (v, ω) pair, that maximizes the objective within V_r. Elements are heading, distance to goal and velocity terms.

5.1.2 Potential Fields

Potential fields are also a common tool in the mobile robotics field used for path planning. A virtual potential $U(q)$ at position $q = (x, y)$ affects the robot and drives him away from any maximum, like a ball rolling downhill [16].
It can be used to attract the robot to a goal by creating a local minimum and repulsing a robot from an obstacle by creating a local maximum. An example of multiple attractive and repulsive potentials is given in Figure 5.2.

The superposition of all attractive and repulsive potential fields yields the overall potential $U(q)$:

$$U(q) = U_{attr}(q) + U_{rep}(q) \tag{5.2}$$

If the potential field is differentiable, we find that the resulting virtual force $F(q)$ that acts on the robot in position q is then defined as

$$F(q) = -\nabla U(q) \tag{5.3}$$

where $\nabla U(q)$ is the gradient of the potential field

$$\nabla U(q) = \begin{pmatrix} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \end{pmatrix} \tag{5.4}$$

Analoguesly, by combining 5.3 and 5.2 we see that the overall force equals the sum of the attractive and repulsive forces:

$$\begin{aligned} F(q) &= F_{att}(q) + F_{rep}(q) \\ &= -\nabla U_{att}(q) - \nabla U_{rep}(q) \end{aligned} \tag{5.5}$$

An **attractive potential**, i.e. a goal is usually defined as a parabolic function

$$U_{att}(q) = \frac{1}{2} k_{att} \cdot {\rho_{goal}}^2(q) \tag{5.6}$$

where k_{att} is the scaling factor and ρ_{goal} is the Euclidian distance to the goal:

$$\rho_{goal} = ||q - q_{goal}|| \tag{5.7}$$

A **repulsive potential**, i.e. an obstacle is zero if a certain distance is exceeded and should rise drastically when the robot gets within close proximity of the obstacle. Hence, a common definition is

$$U_{rep}(q) = \begin{cases} \frac{1}{2} k_{rep} \cdot (\frac{1}{\rho(q)} - \frac{1}{\rho_0})^2 & \text{if } \rho(q) \leq \rho_0 \\ 0 & \text{if } \rho(q) \geq \rho_0 \end{cases} \tag{5.8}$$

where k_{rep} is the scaling factor, ρ_0 is the distance threshold and $\rho(q)$ is the minimal distance from position q to the obstacle.

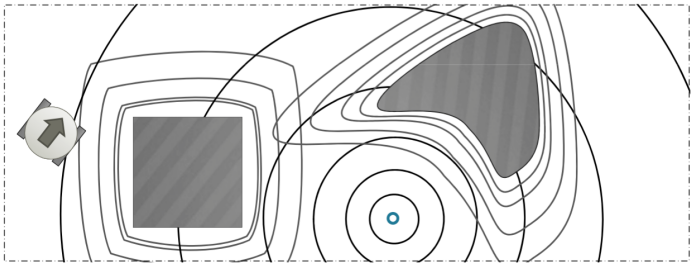


Figure 5.2: Potential field method working principle [16]. Grey areas represent obstacles, the blue circle represents a goal. Depicted in grey and black lines are the equipotential lines of the repulsive and attractive potentials respectively. The robot will follow the direction of the largest gradient, based on his starting position.

The main limitation to this approach is that it is prone to local minima, where the robot can get stuck. However, we are not working with any goals, but rather with an user input in form of a force so even if the robot is within a local minima, the user

can literally pull it away from it. Furthermore, because the output of the potential field method is a force vector, we can elegantly combine it with the admittance controller, which also operates using virtual forces acting on the robot.
So we can conclude that the potential field method is most suitable for a fusion with an admittance controller and we will elaborate on our implementation in the following.

Chapter 6

Implementation

Having the potential field method selected as the desired obstacle avoidance algorithm, we can move on to fusing it together with the admittance controller. Figure 6.1 shows the overall controller schematic as a flowchart, with the nodes used and their messages exchanged, and Table 6.1 summarizes the individual nodes and their ROS messages used.
Firstly, we outline the overall functionality of the system by explaining the functionality of all nodes briefly and dive more deeply into nodes of interest subsequently.

Force-torque sensor The force-torque sensor node continuously outputs the readings from the FT300 (Chapter 3.4) as a wrench F_{ext} of dimension $\mathbb{R}^{6\times1}$. The top three elements are the linear force readings in $[N]$ and the latter the angular torque readings in $[Nm]$. All packed in a ROS message of type *WrenchStamped* [17] and published at a rate of 100 Hz.

LIDAR The output of the LIDAR node are the scans from the Hokuyo laser range finder. Each *LaserScan* ROS message [18] contains the range and intensity of every laser measurement taken from the minimum to the maximum angle of the sensor.

Obstacle avoidance Given the laserscan readings, the obstacle avoidance node creates a costmap [19] around the mobile manipulator, which is explained in detail in Chapter 6.1.2. The costmap is the obstacle potential of all obstacles seen in the laserscans and we derive the resulting force F_{obs} analogously to Equation 5.3 as

$$F_(obs) = -\nabla U_{obs}(q) \tag{6.1}$$

Because the costmap is in two dimensional horizontal space, the resulting force F_{obs} is also planar and has dimension $\mathbb{R}^{2\times1}$.

Admittance control We feed these into the *admittance control*, whose other inputs are the vehicle kinematics, namely the current pose X_{base} [20] and twist $\dot{X}_{base}$ [21] of the ridgeback *mobile base* and the current pose X_{ee} and twist $\dot{X}_{ee}$ of the UR10 *manipulator*. Given these inputs, we calculate the desired twist $\dot{X}_{base_{des}}$ for the mobile base and the desired twist $\dot{X}_{ee_{des}}$ for the manipulator in cartesian space.

Inverse kinematics which in terms needs to be put through the *inverse kinematics* to obtain a feasible velocity $\dot{q}_{ee_{des}}$ in joint space for the manipulator.

Mobile base This node represents the hardware interface and the low level controller of the Ridgeback. It accepts a command velocity as input and using

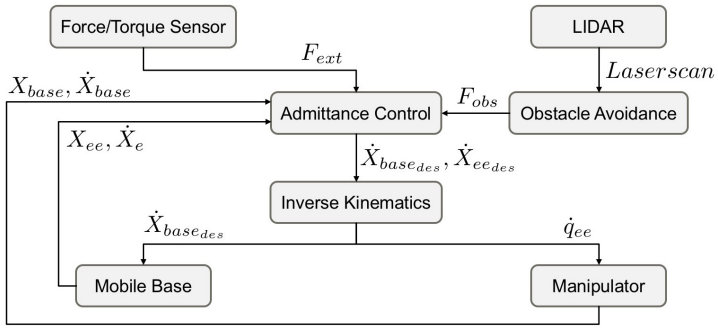


Figure 6.1: Schematic of the controller. Boxes filled in grey are the nodes of the controller and the arrows indicate information flow between the nodes of the control architecture. The top level nodes are the sensor inputs, which are the readings from the *force-torque sensor* and the *LIDAR*. Beneath, we have the nodes which make up the controller, namely the *admittance control*, the *obstacle avoidance* and the *inverse kinematics*. Nodes on the bottom represent the harwarde interfaces for the *mobile base* and the *manipulator*.

the wheel odometry, it can accurately estimate its pose X_{base} and velocity $\dot{X}_{base}$ and publish these.

Manipulator This node represents the hardware interface and the low level controller of the UR10. Joint command inputs can be given through a FollowJointTrajectory action server [22], to which we continuously publish the command joint positions $q_{ee_{des}}$. Similarly to the mobile base node, outputs are the pose X_{ee} and velocity $\dot{X}_{ee}$.

Table 6.1: Controller nodes and their respective outputs.

Node	Output	Dimension	Unit	ROS message type
Force-torque sensor	F_{ext}	$\mathbb{R}^{6\times1}$	$[N]$	WrenchStamped
LIDAR	F_{obs}	$\mathbb{R}^{2\times1}$	$[N]$	WrenchStamped
Admittance control	$\dot{X}_{base}, \dot{X}_{ee}$	$\mathbb{R}^{10\times1}$	$[N]$	Custom twist message
Inverse kinematics	$\dot{X}_{base_{des}}$	$\mathbb{R}^{3\times1}$	$[m/s]$	TwistStamped
	$\dot{q}_{ee_{des}}$	$\mathbb{R}^{6\times1}$	$[rad]$	FollowJointTrajectory
Mobile base	X_{base}	$\mathbb{R}^{7\times1}$	$[m]$	PoseStamped
	$\dot{X}_{base}$	$\mathbb{R}^{6\times1}$	$[m/s]$	TwistStamped
Manipulator	X_{ee}	$\mathbb{R}^{7\times1}$	$[m]$	PoseStamped
	$\dot{X}_{ee}$	$\mathbb{R}^{6\times1}$	$[m/s]$	TwistStamped

6.1 Admittance Control and Obstacle Avoidance Fusion

The key element of the whole algorithm is the fusion of the admittance controller, as described in Chapter 4 and the obstacle avoidance, as described in Chapter 5.1.2. In order to achieve this, we define a virtual two mass spring damper system, depicted in Figure 6.2. Here M_{ee} represents the virtual mass of the EE with pose X_{ee} and M_{base} the virtual mass of the Ridgeback with pose X_{base}. We couple the two masses

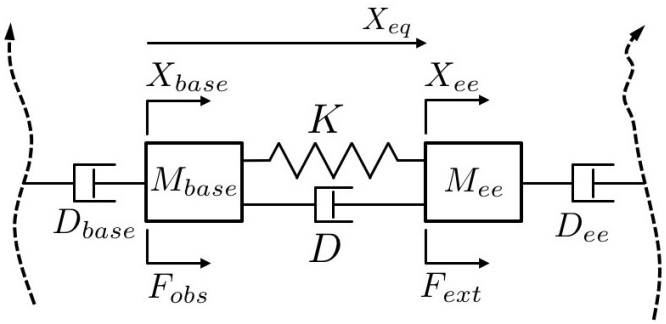


Figure 6.2: Virtual spring mass damper system with two masses. Dotted arrows represent the path of the base (left) and the arm (right) over time, connected by the spring mass damper system.

together using a spring element with a spring constant K and a damping element with a damping constant D. This coupling has an equilibrium position which is defined a priori and is denoted by X_{eq}. As the name suggests, the coupling is in an equilibrium when

$$\overline{\mathrm{X_{base}X_{ee}}} = X_{eq} \tag{6.2}$$

Furthermore, we add additional damping elements D_{ee} and D_{base}, through which we couple the two masses to their desired trajectories, depicted as dotted arrows in Figure 6.2.
Since the EE is where the system is connected to the human partner, any external force F_{ext} exerted by him attacks at M_{ee}. Similarly, because the Ridgeback is running the obstacle avoidance, any force F_{obs} created by it has to attack at M_{base} in the system.
Applying Equation 4.5 to both rigid elements in our virtual system yields the control equations of the fused admittance control and obstacle avoidance algorithm

$$M_{base} \cdot \ddot{X}_{base} = -D_{base} \cdot \dot{X}_{base} - D \cdot \dot{X}_{ee} + K \cdot X_{error} + F_{obs} \tag{6.3}$$

$$M_{ee} \cdot \ddot{X}_{ee} = -(D + D_{ee}) \cdot \dot{X}_{ee} - K \cdot X_{error} + F_{ext} \tag{6.4}$$

where X_{error} is the deviation from the equilibrium pose X_{eq}

$$X_{error} = \overline{\mathrm{X_{base}X_{ee}}} - X_{eq} \tag{6.5}$$

where Equation 6.3 is the force equilibrium for the Ridgeback and Equation 6.4 the force equilibrium for the EE. We can see that through this coupling, any force F_{ext} exerted by the human parter on the EE will also result in a velocity response of the Ridgeback and vice versa will any obstacle force F_{obs} also result in a velocity response of the EE.

6.1.1 Control Parameters

In this chapter, we list and elaborate on the tuned parameters used in Equation 6.3 and 6.4. Since we are in six dimensional space and we are not interested in having any interdependence between the individual axes, all constants are six dimensional diagonal matrices. Therefore, we guarantee a full decoupling of all axes.

We set the equilibrium pose X_eq of the EE in respect to the frame of the armbase to be offset by 1 m in x and 0.5 m in z. Equation 6.6 shows the full pose in $\mathbb{R}^{7\times1}$, consisting of a position and a quaternion[1].

$$X_{eq} = \begin{pmatrix} -1 \\ 0 \\ 0.5 \\ 0 \\ 0 \\ 0.7071067 \\ 0.7071069 \end{pmatrix} \begin{bmatrix} m \\ - \end{bmatrix} \tag{6.6}$$

Since all our parameters are virtual, we can start off by setting the mass of the EE M_{ee} as the identity matrix. From there, we tune the remaining parameters M_{base}, D_{ee},D_{base},D and K, which is essentially the same as tuning a PID controller. The goal is to have a fast response on the EE, so that jerky input from the human partner can be compensated and a slower response by the Ridgeback to follow the overall trajectory.

$$M_{ee} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} kg \\ kg \cdot m^2 \end{bmatrix} \tag{6.7}$$

$$D_{ee} = \begin{pmatrix} 8 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 20 & 0 & 0 \\ 0 & 0 & 0 & 0 & 20 & 0 \\ 0 & 0 & 0 & 0 & 0 & 8 \end{pmatrix} \begin{bmatrix} \frac{Ns}{m} \\ Nms \end{bmatrix} \tag{6.8}$$

An important note here is that although Equation 6.3 is six dimensional, the output sent to the Ridgeback is only three dimensional, as explained in Chapter 3.1. Hence, parameters in linear z, angular x and angular y, i.e., the third through fifth rows of Equation 6.9 and Equation 6.10 are irrelevant for the performance of the controller.

$$M_{base} = \begin{pmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 \end{pmatrix} \begin{bmatrix} kg \\ kg \cdot m^2 \end{bmatrix} \tag{6.9}$$

$$D_{base} = \begin{pmatrix} 20 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 \end{pmatrix} \begin{bmatrix} \frac{Ns}{m} \\ Nms \end{bmatrix} \tag{6.10}$$

[1] The UR10 base frame is rotated by 180 ° around the vertical axis z in respect to the Ridgeback base frame, hence the negative sign in x.

$$D = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{bmatrix} \frac{Ns}{m} \\ Nms \end{bmatrix} \tag{6.11}$$

We set the stiffness of the coupling in Equation 6.12 deliberately to 500 $^{\mathrm{N}}/_{\mathrm{m}}$ in linear z, because we want the jointly carried object to be close to the height of the equilibrium pose, independent of its weight. Similarly, we set the stiffness in angular x and angular y to 500 Nm, in order to prevent rotation of the object around these two axes.

$$K = \begin{pmatrix} 150 & 0 & 0 & 0 & 0 & 0 \\ 0 & 150 & 0 & 0 & 0 & 0 \\ 0 & 0 & 500 & 0 & 0 & 0 \\ 0 & 0 & 0 & 500 & 0 & 0 \\ 0 & 0 & 0 & 0 & 500 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \end{pmatrix} \begin{bmatrix} \frac{N}{m} \\ Nm \end{bmatrix} \tag{6.12}$$

6.1.2 Costmap of the Obstacle Potential

Using the ROS costmap package [19], this node builds a two dimensional occupancy grid based on the incoming laserscans and inflates the cost according to user specified parameters. Table 6.2 lists all parameters and their set values. Since we do only local planning, a local costmap, meaning the center of the costmap is equal to the center of the robot, of size 3×3 m suffices.
The algorithm assigns a value from 0 to 100 to every cell of the occupancy grid, which represents the cost c of that cell. We can distinguish four types of cells:

Lethal A lethal cost is assigned to all cells, which are currently occupied by an obstacle and would therefore guarantee a collision, if the robot center would be at one of these cells.

$$c_{lethal} = 100 \tag{6.13}$$

Inscribed This category applies to all cells, whose distance to an obstacle is less than the radius of the robot, which is defined by the parameter *robot footprint*. Similarly to the lethal cells, a collision is guaranteed if the robot center overlaps with an inscribed cell.

$$c_{inscribed} = 99 \tag{6.14}$$

Obstacle proximity Obstacle proximity groups all cells, that make up the repulsive potential U_{rep} of the obstacle, as defined in Equation 5.8. The distance threshold *inflation radius* defines the size of the potential and the *inflater layer cost scaling factor* defines the steepness of the quadratic curve. These cells represent that the robot center is close to an obstacle but not necessarily colliding.

$$c_{proximity} = \big[1, 98\big] \tag{6.15}$$

Freespace All cells that are further away from an obstacle than the distance threshold *inflation radius* are set to 0, meaning that the robot can travel freely through theses cells.

$$c_{free} = 0 \tag{6.16}$$

Figure 6.3 depicts an example of the costmap of a scenario, where the Thing is between two obstacles.

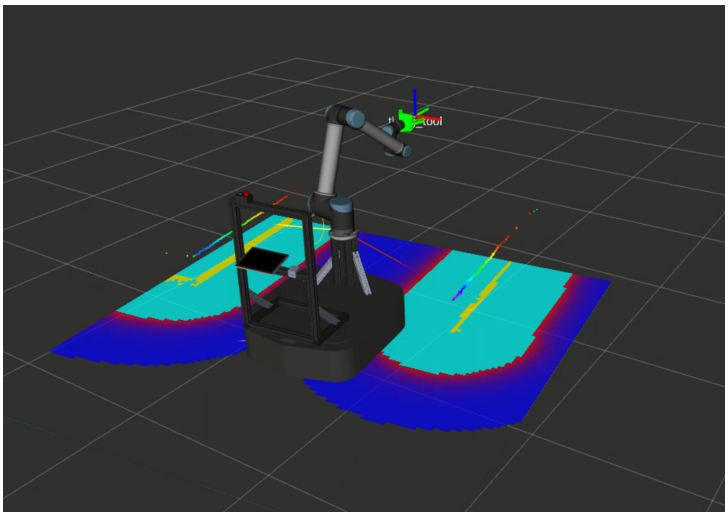


Figure 6.3: Costmap visualization during a test, depicted with the simulation of the Thing. YELLOW: Lethal cells ($c = 100$), TURQUOISE: Inscribed cells ($c = 99$), RED TO BLUE GRADIENT: Proximity to an obstacle ($c = [1, 98]$), TRANSPARENT: Free space ($c = 0$).

6.2 Task Priority Inverse Kinematics

Forward kinematics, which is the mapping from joint coordinates q to the end-effector configuration X_{ee} of a mobile manipulator, is a well understood problem and can be solved using the *Jacobian* $J(q)$ which is defined as

$$J(q) = \begin{bmatrix} \frac{\partial X_1}{\partial q_1} & \cdots & \frac{\partial X_1}{\partial q_{n_j}} \\ \vdots & \ddots & \vdots \\ \frac{\partial X_m}{\partial q_1} & \cdots & \frac{\partial X_m}{\partial q_{n_j}} \end{bmatrix} \tag{6.17}$$

It relates differences from joint coordinates q to EE configuration space X_{ee}:

$$\dot{X}_{ee} = J(q) \cdot \dot{q} \tag{6.18}$$

The inverse kinematics, i.e., the calculation of the joint velocity $\dot{q}$ given an EE velocity $\dot{X}_{ee}$ is already trickier, but can be achieved using the *Moore-Penrose pseudo inverse* J^+ of the Jacobian:

$$\dot{q} = J^+ \cdot \dot{X}_{ee} \tag{6.19}$$

where J^+ is defined as

$$J^+ = J^T (J J^T)^{-1} \tag{6.20}$$

Furthermore, because we are working with a mobile manipulator and not a rigidly fixed robot arm, there are two cartesian velocities $\dot{X}_1$ and $\dot{X}_2$, for which we want to solve simultaneously. Hence, we use a multi-task inverse kinematics control that can prioritize among multiple tasks [23]. This is achieved by utilizing *null-space optimization* to execute a secondary task with Jacobian J_2, given a primary task with Jacobian J_1.

$$\dot{q} = J_1^+ \dot{X}_1 + N_1 J_2^+ N_1 (\dot{X}_2 - J_2 J_1^+ \dot{X}_1) \tag{6.21}$$

Table 6.2: Costmap Parameters.

Type	Local
Update frequency	60Hz
Publish frequency	60Hz
Width	3m
Height	3m
Resolution	0.03m
Global frame	Base link
Static map	False
Rolling window	True
Robot footprint	0.96×0.8m
Plugins used	Obstacles layer, inflater layer
Inflater layer cost scaling factor	10
Inflation radius	2m

Equation 6.21 formulates the task priority approach to inverse kinematics for two tasks, with N_i being the null-space projection matrix of J_i, fulfilling the condition

$$J_i N_i = 0 \tag{6.22}$$

The simplest projection is given by

$$N_i = \mathbb{I} - J_i^+ J_i \tag{6.23}$$

where $\mathbb{I}$ denotes the identity matrix.
Since an accurate execution of the desired Ridgeback velocity $\dot{X}_{base_{des}}$ is crucial to ensure a successful obstacle avoidance, we choose $\dot{X}_{base_{des}}$ to be our primary task and $\dot{X}_{ee_{des}}$ our secondary.

Chapter 7

Results

We test our previously elaborated implementation on the Thing and list the performed tests and their results in this chapter. Figure 7.1 depicts the test setup of the Thing gripping the object to be jointly carried. The object is made from wood and 91.44 cm in length. Furthermore, we have all low level controllers running on the computer embedded in the Ridgeback and the admittance control and obstacle avoidance running on a laptop, connected through Ethernet.

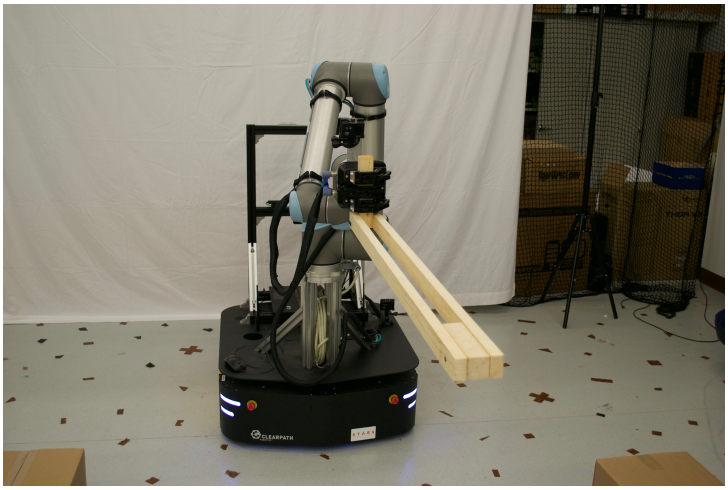


Figure 7.1: The Thing holding the jointly carried object used in the tests. It is a wooden structure 3 feet or 91.44 cm in length and equipped with a handle to facilitate gripping.

7.1 Admittance Control Performance

As initial tests, we analyse the behaviour of the admittance controller in isolation. For this, we place the Thing in free space with no obstacles, i.e., in a region with zero gradient and excite three axes separately by applying force and torque on the object. The three axes are linear x, linear y and angular z, i.e., the three DOF of the Ridgeback. Therefore, we see a response of both the base and the end-effector in all three tests.
Figure 7.2 depicts the velocity response of the system to a force excitement in x, Figure 7.3 the response to a force excitement in y and Figure 7.4 around z, i.e., the angular velocity θ given a torque input around z.

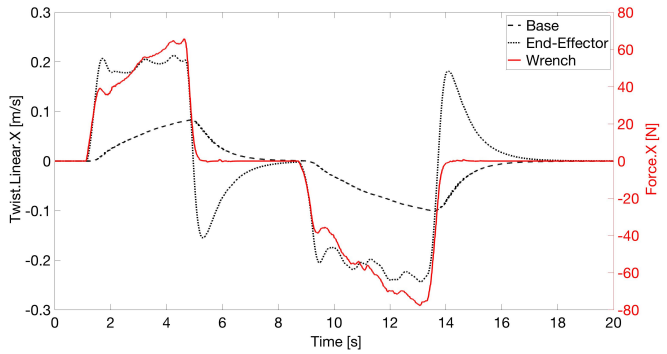


Figure 7.2: Velocity Response to Force Excitement in x

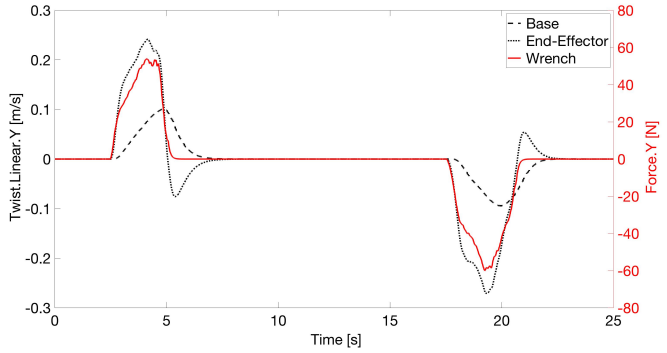


Figure 7.3: Velocity Response to Force Excitement in y

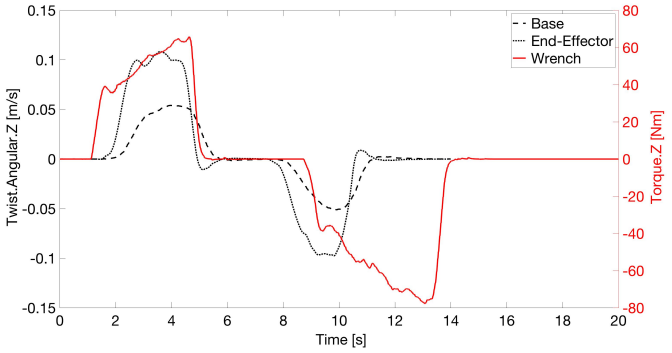


Figure 7.4: Velocity Response to Torque Excitement around z

If we look at the response of the EE in both x, y and θ, we notice that it has a under-critically damped controller, which leads to a quick rise time and overshooting in the opposite direction, as soon as no more force is applied. This behaviour is actually desired, since we want to counteract sudden movement and jerky input by the human partner with the EE and only output smooth behaviour to the Ridgeback. If we have a look at the velocity response in x, y and θ of the latter, we see a vastly slower rise time and no overshooting, i.e., a critically damped controller.
Furthermore, we notice that the linear velocity response of the EE peaks at $0.2\,\mathrm{m/s}$ and the linear velocity response of the Ridgeback at $0.1\,\mathrm{m/s}$.

7.2 Obstacle Avoidance and Admittance Control Performance

Having the admittance controller tuned and working as desired, we can move on to testing the full algorithm where obstacle avoidance comes into play. We use cardboard boxes of dimension $14 \times 14 \times 60$ inches as obstacles, so that in case of a failure of the controller and an occurrence of a crash, the Thing does not get damaged.
Because we have no 360° Lidar coverage, but only a front-facing field of view, we can test the obstacle avoidance in forward and sideways motion of the Thing only. However, we've seen in Chapter 7.1, that the response to the admittance controller is independent of direction. Hence, we can assume that the full algorithm is scalable to a backward motion as well, if a back-facing Lidar were to be mounted.

7.2.1 Frontal Obstacle

The first obstacle avoidance test is the robot being pulled into a obstacle directly ahead of him. The scenario is depicted in Figure 7.5. The velocity response of the system in the direction of pull, which is x, is plotted in Figure 7.6.

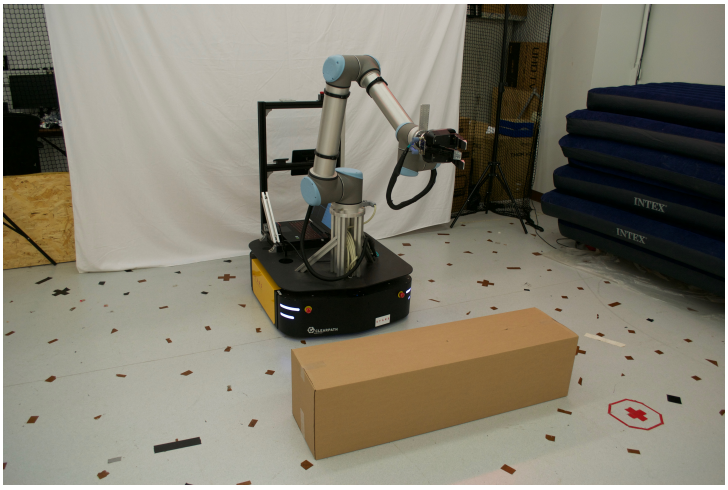


Figure 7.5: Set up of the frontal obstacle test scenario.

We see that the velocity responses of both EE and Ridgeback are constant, if a constant force is applied by the human partner and no obstacle is in range. However, as soon as we close in on the obstacle, the obstacle force pointing away from the obstacle rises and gradually diminishes the velocity respones to zero, although there is still a constant external force applied.

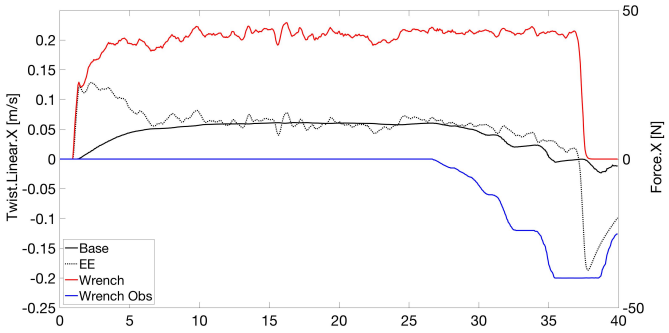


Figure 7.6: Velocity response in X axis of the robot to external and obstacle force input while being pulled in a frontal obstacle.

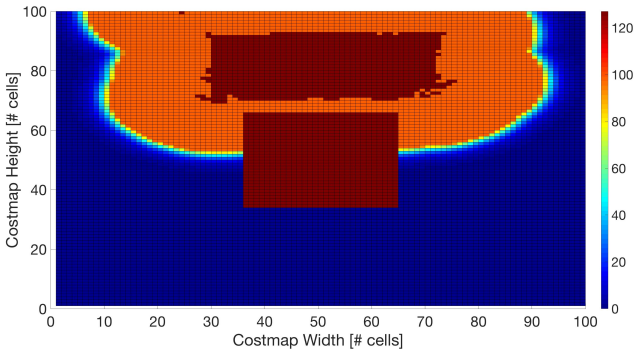


Figure 7.7: Costmap at 34s of the frontal test. Dark red indicate the obstacle and the robot base.

Figure 7.7 depicts the cost map of the algorithm at the end of the experiment, i.e., at the robots closest position to the obstacle. We see the robot comes no closer than four grid cells to the obstacle, which equals a minimum obstacle clearance of 12 cm. Together with the width of the Ridgeback of 0.8 m, the lower bound on a narrow passage is thus slightly above1 m.

7.2.2 Corner

In this experiment, we asses the performance of the algorithm in a corridor with a corner. The width of corridor is 2m and we jointly carry the object through it. The set-up at the beginning of the test is depicted in Figure 7.8.
As we can see in Figure 7.9, there is a peak in the negative y direction, which is caused by the robot drifting close to the outer wall as it is making the corner. This is due to the fact, that it is difficult for the human partner to apply torque in angular z, which makes the Thing rotate in place, without applying force in linear y, which makes the Thing move sideways. Hence, the trajectory of Ridgeback does not follow the center line of the hallway in a corner scenario, but rather utilizes the full width of the hallway.

7.2.3 Cluttered Space

As a final experiment, we test the behaviour in a cluttered environment, as seen in Figure 7.8. It consists of four boxes in a perpendicular arrangement with the last

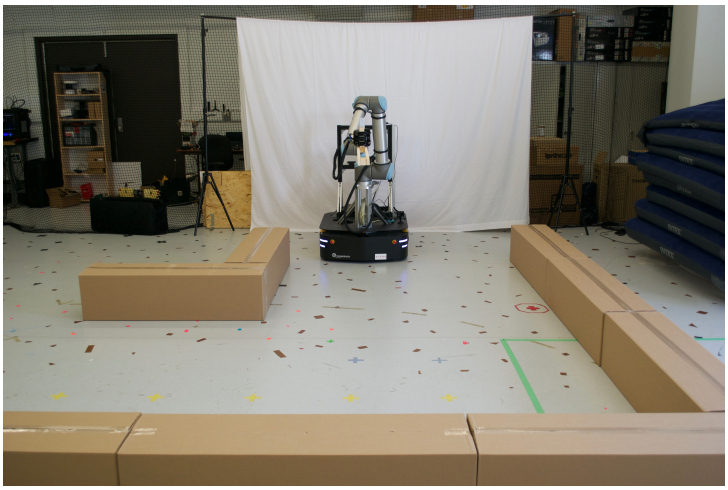


Figure 7.8: Setup of the 2m wide corner scenario.

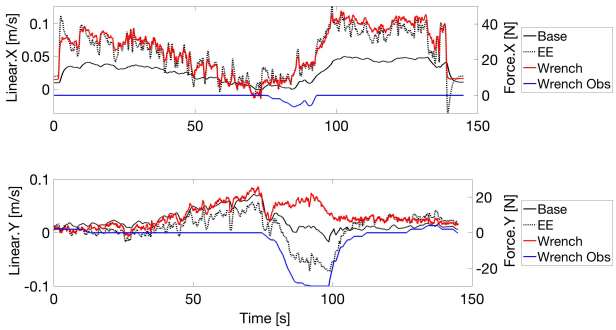


Figure 7.9: Velocity response to external and obstacle force input in 2m wide corner scenario.

two boxes forming a passage of 120 cm width.

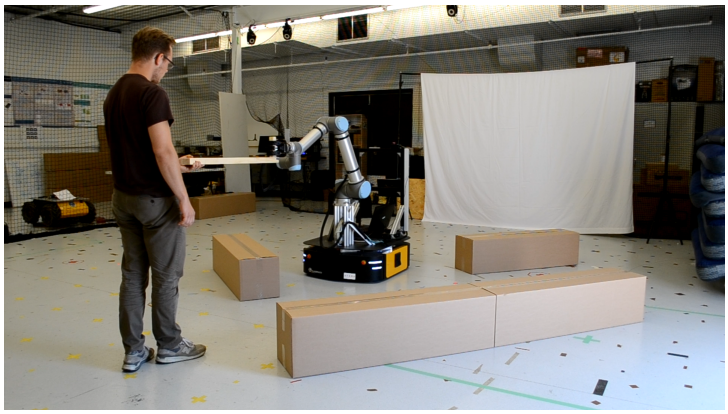


Figure 7.10: Execution of a test in the cluttered environment scenario.

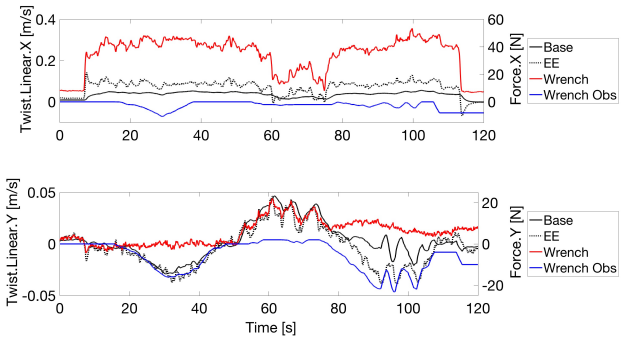


Figure 7.11: Velocity response to external and obstacle force input in the cluttered environment scenario.

Chapter 8

Conclusion

Chapter 9

Einige wichtige Hinweise zum Arbeiten mit LaTeX

Nachfolgend wird die Codierung einiger oft verwendeten Elemente kurz beschrieben. Das Einbinden von Bildern ist in LaTeX nicht ganz unproblematisch und hängt auch stark vom verwendeten Compiler ab. Typisches Format für Bilder in LaTeX ist EPS[1] oder PDF[2].

9.1 Gliederungen

Ein Text kann mit den Befehlen `\chapter{.}`, `\section{.}`, `\subsection{.}` und `\subsubsection{.}` gegliedert werden.

9.2 Referenzen und Verweise

Literaturreferenzen werden mit dem Befehl `\citep{.}` und `\citet{.}` erzeugt. Beispiele: ein Buch [**?**], ein Buch und ein Journal Paper [**?** **?**], ein Konferenz Paper mit Erwähnung des Autors: **?**].
Zur Erzeugung von Fussnoten wird der Befehl `\footnote{.}` verwendet. Auch hier ein Beispiel[3].
Querverweise im Text werden mit `\label{.}` verankert und mit `\cref{.}` erzeugt. Beispiel einer Referenz auf das zweite Kapitel: chapter 9.

9.3 Aufzählungen

Folgendes Beispiel einer Aufzählung ohne Numerierung,

- Punkt 1
- Punkt 2

wurde erzeugt mit:

```
\begin{itemize}
  \item Punkt 1
  \item Punkt 2
\end{itemize}
```

[1] Encapsulated Postscript
[2] Portable Document Format
[3] Bla bla.

Folgendes Beispiel einer Aufzählung mit Numerierung,

1. Punkt 1
2. Punkt 2

wurde erzeugt mit:

```
\begin{enumerate}
  \item Punkt 1
  \item Punkt 2
\end{enumerate}
```

Folgendes Beispiel einer Auflistung,

P1 Punkt 1

P2 Punkt 2

wurde erzeugt mit:

```
\begin{description}
  \item[P1] Punkt 1
  \item[P2] Punkt 2
\end{description}
```

9.4 Erstellen einer Tabelle

Ein Beispiel einer Tabelle:

Table 9.1: Daten der Fahrzyklen ECE, EUDC, NEFZ.

Kennzahl	Einheit	ECE	EUDC	NEFZ
Dauer	s	780	400	1180
Distanz	km	4.052	6.955	11.007
Durchschnittsgeschwindigkeit	km/h	18.7	62.6	33.6
Leerlaufanteil	%	36	10	27

Die Tabelle wurde erzeugt mit:

```
\begin{table}[h]
\begin{center}
 \caption{Daten der Fahrzyklen ECE, EUDC, NEFZ.}\vspace{1ex}
 \label{tab:tabnefz}
 \begin{tabular}{ll|ccc}
 \hline
 Kennzahl & Einheit & ECE & EUDC & NEFZ \\ \hline \hline
 Dauer & s & 780 & 400 & 1180 \\
 Distanz & km & 4.052 & 6.955 & 11.007 \\
 Durchschnittsgeschwindigkeit & km/h & 18.7 &  62.6 & 33.6 \\
 Leerlaufanteil & \% & 36 & 10 & 27 \\
 \hline
 \end{tabular}
\end{center}
\end{table}
```

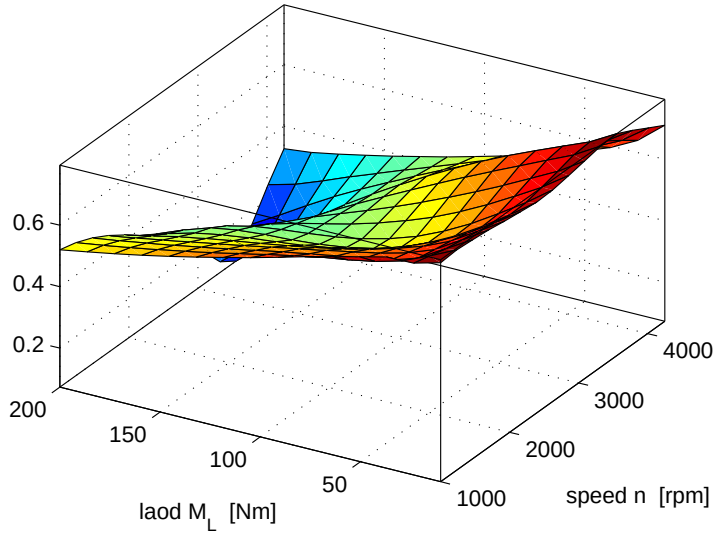


Figure 9.1: Ein Bild

9.5 Einbinden einer Grafik

Das Einbinden von Graphiken kann wie folgt bewerkstelligt werden:

```
\begin{figure}
   \centering
   \includegraphics[width=0.75\textwidth]{images/k_surf.pdf}
   \caption{Ein Bild.}
   \label{fig:k_surf}
\end{figure}
```

oder bei zwei Bildern nebeneinander mit:

```
\begin{figure}
  \begin{minipage}[t]{0.48\textwidth}
    \includegraphics[width = \textwidth]{images/cycle_we.pdf}
  \end{minipage}
  \hfill
  \begin{minipage}[t]{0.48\textwidth}
    \includegraphics[width = \textwidth]{images/cycle_ml.pdf}
  \end{minipage}
  \caption{Zwei Bilder nebeneinander.}
  \label{pics:cycle}
\end{figure}
```

9.6 Mathematische Formeln

Einfache mathematische Formeln werden mit der equation-Umgebung erzeugt:

$$p_{me0f}(T_e, \omega_e) \;=\; k_1(T_e) \cdot (k_2 + k_3 S^2 \omega_e^2) \cdot \Pi_{\max} \cdot \sqrt{\frac{k_4}{B}}\,. \tag{9.1}$$

Der Code dazu lautet:

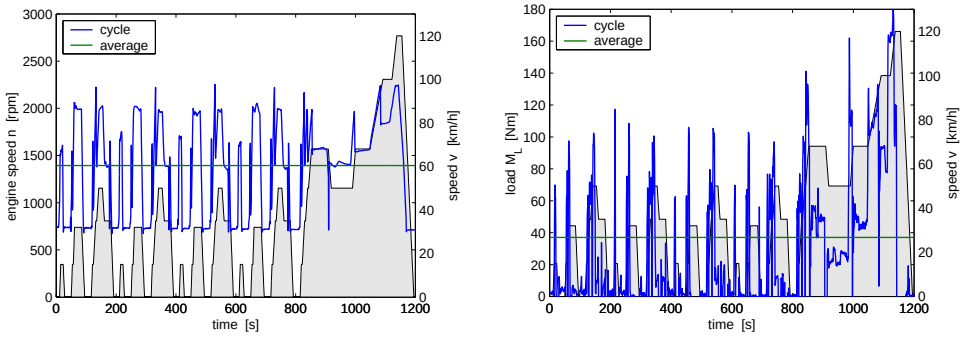


Figure 9.2: Zwei Bilder nebeneinander

```
\begin{equation}
 p_{me0f}(T_e,\omega_e) \ = \ k_1(T_e) \cdot (k_2+k_3 S^2
 \omega_e^2) \cdot \Pi_{max} \cdot \sqrt{\frac{k_4}{B}} \, .
\end{equation}
```

Mathematische Ausdrücke im Text werden mit \$formel\$ erzeugt (z.B.: $a^2+b^2=c^2$). Vektoren und Matrizen werden mit den Befehlen `\vec{.}` und `\mat{.}` erzeugt (z.B. $\boldsymbol{v}$, $\boldsymbol{M}$).

9.7 Weitere nützliche Befehle

Hervorhebungen im Text sehen so aus: *hervorgehoben*. Erzeugt werden sie mit dem `\epmh{.}` Befehl.
Einheiten werden mit den Befehlen `\unit[1]{m}` (z.B. 1 m) und `\unitfrac[1]{m}{s}` (z.B. 1 m/s) gesetzt.

Bibliography

[1] N. Jarrassé, T. Charalambous, and E. Burdet, "A framework to describe, analyze and generate interactive motor behaviors," *PloS one*, vol. 7, no. 11, p. e49945, 2012.

[2] J. Bütepage and D. Kragic, "Human-robot collaboration: From psychology to social robotics," *arXiv preprint arXiv:1705.10146*, 2017.

[3] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[4] K. B. Reed and M. A. Peshkin, "Physical collaboration of human-human and human-robot teams," *IEEE Transactions on Haptics*, vol. 1, no. 2, pp. 108–120, 2008.

[5] K. B. Reed, M. Peshkin, M. J. Hartmann, J. Patton, P. M. Vishton, and M. Grabowecky, "Haptic cooperation between people, and between people and machines," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on.* Ieee, 2006, pp. 2109–2114.

[6] P. Evrard and A. Kheddar, "Homotopy switching model for dyad haptic interaction in physical collaborative tasks," in *EuroHaptics conference, 2009 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics 2009. Third Joint.* IEEE, 2009, pp. 45–50.

[7] A. Tagliabue, M. Kamel, S. Verling, R. Siegwart, and J. Nieto, "Collaborative transportation using mavs via passive force control," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on.* IEEE, 2017, pp. 5766–5773.

[8] M. Lawitzky, A. Mörtl, and S. Hirche, "Load sharing in human-robot cooperative manipulation," in *RO-MAN, 2010 IEEE.* IEEE, 2010, pp. 185–191.

[9] "Clearpath ridgeback," https://www.clearpathrobotics.com/ridgeback-indoor-robot-platform/, accessed: 2018-08-31.

[10] "Universal robot 10," https://www.universal-robots.com/products/ur10-robot/, accessed: 2018-08-31.

[11] "Robotiq three finger adaptive robot gripper," https://robotiq.com/products/3-finger-adaptive-robot-gripper, accessed: 2018-08-31.

[12] "Hokuyo ust-10lx," https://www.hokuyo-usa.com/products/scanning-laser-rangefinders/ust-10lx, accessed: 2018-08-31.

[13] "Robotiq ft 300," https://robotiq.com/products/ft-300-force-torque-sensor, accessed: 2018-08-31.

[14] N. Hogan, "Impedance control: An approach to manipulation: Part ii—implementation," *Journal of dynamic systems, measurement, and control*, vol. 107, no. 1, pp. 8–16, 1985.

[15] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[16] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, "Autonomous mobile robots," *Massachusetts Institute of Technology*, 2004.

[17] "Wrenchstamped message definition," http://docs.ros.org/indigo/api/geometry_msgs/html/msg/WrenchStamped.html, accessed: 2018-08-31.

[18] "Laserscan message definition," http://docs.ros.org/indigo/api/sensor_msgs/html/msg/LaserScan.html, accessed: 2018-08-31.

[19] "Costmap 2d," http://wiki.ros.org/costmap_2d.

[20] "Posestamped message definition," http://docs.ros.org/indigo/api/geometry_msgs/html/msg/PoseStamped.html, accessed: 2018-08-31.

[21] "Twiststamped message definition," http://docs.ros.org/indigo/api/geometry_msgs/html/msg/TwistStamped.html, accessed: 2018-08-31.

[22] "Followjointtrajectory action definition," http://docs.ros.org/api/control_msgs/html/action/FollowJointTrajectory.html, accessed: 2018-08-31.

[23] Y. Nakamura, H. Hanafusa, and T. Yoshikawa, "Task-priority based redundancy control of robot manipulators," *The International Journal of Robotics Research*, vol. 6, no. 2, pp. 3–15, 1987.

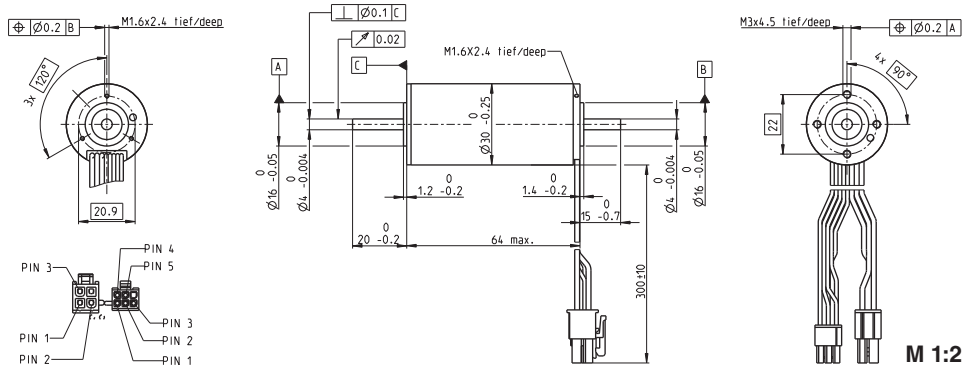
Appendix A

Code

Appendix B

Datasheets

EC-max 30 ∅30 mm, brushless, 60 Watt



Stock program
Standard program
Special program (on request)

Motor Data		Part Numbers 272762	272763	272764	272765
Values at nominal voltage					
1 Nominal voltage	V	12	24	36	48
2 No load speed	rpm	7980	9340	9490	9350
3 No load current	mA	302	191	130	95.4
4 Nominal speed	rpm	6590	8040	8270	8130
5 Nominal torque (max. continuous torque)	mNm	63.6	60.7	63.7	64.1
6 Nominal current (max. continuous current)	A	4.72	2.66	1.88	1.4
7 Stall torque	mNm	381	458	522	519
8 Starting current	A	26.8	18.8	14.5	10.7
9 Max. efficiency	%	80	81	82	82
Characteristics					
10 Terminal resistance phase to phase	Ω	0.447	1.27	2.48	4.49
11 Terminal inductance phase to phase	mH	0.049	0.143	0.312	0.573
12 Torque constant	mNm/A	14.2	24.3	35.9	48.6
13 Speed constant	rpm/V	672	393	266	197
14 Speed/torque gradient	rpm/mNm	21.2	20.6	18.4	18.2
15 Mechanical time constant	ms	4.86	4.73	4.21	4.17
16 Rotor inertia	gcm²	21.9	21.9	21.9	21.9

Specifications

Thermal data

17	Thermal resistance housing-ambient	7.4 K/W
18	Thermal resistance winding-housing	0.5 K/W
19	Thermal time constant winding	2.76 s
20	Thermal time constant motor	1000 s
21	Ambient temperature	-40…+100°C
22	Max. permissible winding temperature	+155°C

Mechanical data (preloaded ball bearings)

23	Max. permissible speed		15000 rpm
24	Axial play at axial load	< 6.0 N	0 mm
		> 6.0 N	0.14 mm
25	Radial play		preloaded
26	Max. axial load (dynamic)		5 N
27	Max. force for press fits (static)		98 N
	(static, shaft supported)		1300 N
28	Max. radial loading, 5 mm from flange		25 N

Other specifications

29	Number of pole pairs	1
30	Number of phases	3
31	Weight of motor	305 g

Values listed in the table are nominal.

Connection motor (Cable AWG 20)

red	Motor winding 1	Pin 1
black	Motor winding 2	Pin 2
white	Motor winding 3	Pin 3
	N.C.	Pin 4

Connector	Part number
Molex	39-01-2040

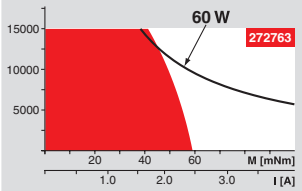
Connection Sensors (Cable AWG 26)

yellow	Hall sensor 1	Pin 1
brown	Hall sensor 2	Pin 2
grey	Hall sensor 3	Pin 3
blue	GND	Pin 4
green	V_{Hall} 3…24 VDC	Pin 5
	N.C.	Pin 6

Connector	Part number
Molex	430-25-0600

Wiring diagram for Hall sensors see p. 35

Operating Range



Comments

Continuous operation
In observation of above listed thermal resistance (lines 17 and 18) the maximum permissible winding temperature will be reached during continuous operation at 25°C ambient.
= Thermal limit.

Short term operation
The motor may be briefly overloaded (recurring).

Assigned power rating

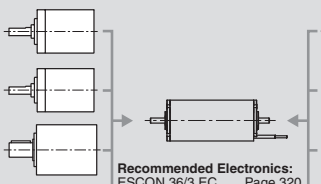
maxon Modular System

Overview on page 20 - 25

Planetary Gearhead
∅32 mm
8.0 Nm
Page 266

Koaxdrive
∅32 mm
1.0 - 4.5 Nm
Page 268

Planetary Gearhead
∅42 mm
3 - 15 Nm
Page 271



Recommended Electronics:

ESCON 36/3 EC	Page 320
ESCON 50/5, Module 50/5	321
ESCON 70/10	321
DECS 50/5	324
DEC Module 24/2	325
DEC Module 50/5	325
EPOS2 24/5, 50/5	331
EPOS2 P 24/5	334
EPOS3 70/10 EtherCAT	337
Notes	**24**

Encoder MR
500/1000 CPT,
3 channels
Page 302

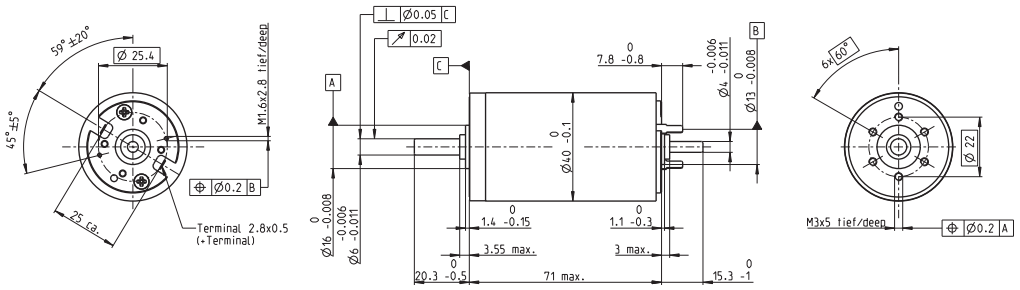
Encoder HEDL 5540
500 CPT,
3 channels
Page 308

Brake AB 20
24 VDC
0.1 Nm
Page 346

RE 40 ∅40 mm, Precious Metal Brushes, 25 Watt

NEW

maxon DC motor



M 1:2

Stock program
Standard program
Special program (on request)

			Part Numbers						
			448588	448589	448590	448591	448592		
Motor Data									
	Values at nominal voltage								
1	Nominal voltage	V	9	18	24	42	48		
2	No load speed	rpm	2850	2850	2780	2920	2690		
3	No load current	mA	49.7	24.8	18.1	11	8.62		
4	Nominal speed	rpm	2610	2600	2480	2640	2410		
5	Nominal torque (max. continuous torque)	mNm	87.8	87.8	88.2	87.6	87.6		
6	Nominal current (max. continuous current)	A	2.96	1.48	1.09	0.65	0.524		
7	Stall torque	mNm	873	956	794	895	818		
8	Starting current	A	29	15.9	9.66	6.53	4.81		
9	Max. efficiency	%	92	92	92	92	92		
	Characteristics								
10	Terminal resistance	Ω	0.311	1.14	2.49	6.43	9.97		
11	Terminal inductance	mH	0.0824	0.33	0.613	1.7	2.62		
12	Torque constant	mNm/A	30.2	60.3	82.2	137	170		
13	Speed constant	rpm/V	317	158	116	69.7	56.2		
14	Speed / torque gradient	rpm/mNm	3.27	2.98	3.51	3.27	3.3		
15	Mechanical time constant	ms	4.85	4.29	4.36	4.14	4.13		
16	Rotor inertia	gcm²	142	137	119	121	120		

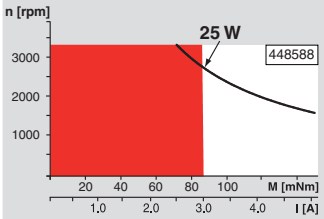
Specifications

	Thermal data	
17	Thermal resistance housing-ambient	4.65 K/W
18	Thermal resistance winding-housing	1.93 K/W
19	Thermal time constant winding	41.5 s
20	Thermal time constant motor	809 s
21	Ambient temperature	-20…+85°C
22	Max. permissible winding temperature	+100°C
	Mechanical data (ball bearings)	
23	Max. permissible speed	3330 rpm
24	Axial play	0.05 - 0.15 mm
25	Radial play	0.025 mm
26	Max. axial load (dynamic)	5.6 N
27	Max. force for press fits (static)	110 N
	(static, shaft supported)	1200 N
28	Max. radial loading, 5 mm from flange	28 N
	Other specifications	
29	Number of pole pairs	1
30	Number of commutator segments	13
31	Weight of motor	480 g

Values listed in the table are nominal.
Explanation of the figures on page 71.

Option
Preloaded ball bearings

Operating Range



Comments

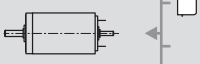
Continuous operation
In observation of above listed thermal resistance (lines 17 and 18) the maximum permissible winding temperature will be reached during continuous operation at 25°C ambient.
= Thermal limit.

Short term operation
The motor may be briefly overloaded (recurring).

Assigned power rating

maxon Modular System

Overview on page 20 - 25



Encoder MR
256 - 1024 CPT,
3 channels
Page 303

Encoder HED_ 5540
500 CPT,
3 channels
Page 305/307

Recommended Electronics:

ESCON 36/2 DC	Page 320
ESCON 50/5	321
ESCON Module 50/5	321
EPOS2 24/2	330
EPOS2 Module 36/2	330
EPOS2 24/5	331
EPOS2 50/5	331
EPOS2 P 24/5	334
EPOS3 70/10 EtherCAT	337
Notes	**22**