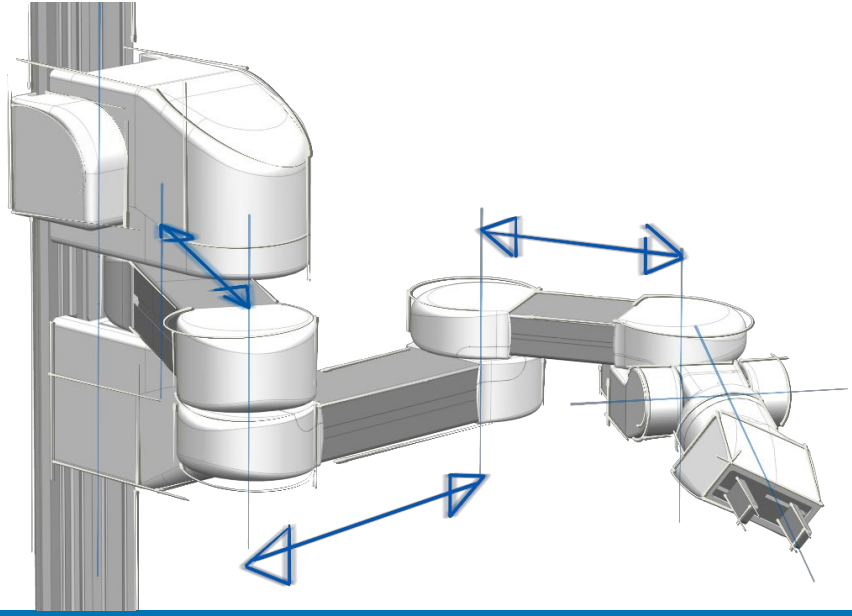
MECHANISM THEORY, MACHINE DYNAMICS AND ROBOTICS



Veronika Zumpe

Semi-automated Task-based Synthesis and Design of Serial Robotic Systems

IGMR Institute of Mechanism Theory, Machine Dynamics and Robotics

RWTH AACHEN UNIVERSITY

Semi-automated Task-based Synthesis and Design of Serial Robotic Systems

Teilautomatisierte, taskbasierte Synthese und Entwicklung von seriellen Robotersystemen

Von der Fakultät für Maschinenwesen
der Rheinisch-Westfälischen Technischen Hochschule Aachen
zur Erlangung des akademischen Grades einer
Doktorin der Ingenieurwissenschaften
genehmigte Dissertation

vorgelegt von

Veronika Zumpe

Berichter: apl. Prof. Dr.-Ing. Mathias Hüsing
Univ.-Prof. Dr.-Ing. Dr. h. c. (UPT) Burkhard Corves
Assoz.-Prof. Priv.-Doz. Mag.rer.nat. Dr.techn. Martin Pfurner

Tag der mündlichen Prüfung: 03.02.2023

DOI: 10.18454/RWTH-2023-02136

Vorwort

Die vorliegende Arbeit entstand während meiner Tätigkeit als Ingenieurin in der Konzernforschung der KUKA Deutschland GmbH mit Betreuung der wissenschaftlichen Arbeit durch das Institut für Getriebetechnik, Maschinendynamik und Robotik (IGMR) der RWTH Aachen.

Zunächst danke ich besonders meinem Doktorvater Prof. Mathias Hüsing, der mir durch die Betreuung das Vorhaben überhaupt ermöglichte und mich stets unterstützte. Die Zusammenarbeit hat mir große Freude bereitet und ich bin dankbar über die gemeinsamen Momente und den wertvollen Austausch. Ebenso bedanke ich mich bei Prof. Burkhard Corves, dass ich bei ihm am IGMR promovieren durfte und die Möglichkeit hatte, an den spannenden Themen zu arbeiten. Großer Dank gilt Prof. Martin Pfurner für die Berichtsübernahme und die wertvollen Diskussionen über Robotik, Kinematik und Geometrie. Weiterhin bedanke ich mich bei Prof. Julia Kowalski für die Übernahme des Prüfungsvorsitzes und ihr Interesse an meiner Arbeit.

Herzlicher Dank gilt der Firma KUKA und insbesondere Dr. Martin Riedel, denn durch ihn und unsere Zusammenarbeit während des Bachelors habe ich meine Begeisterung für die Robotik entdeckt. Ich bin froh, dass sich unsere Wege nach dem Master wieder getroffen haben und ich in seinem Team neben meiner Arbeit als Ingenieurin meine wissenschaftliche Arbeit vorantreiben konnte. Ich bedanke mich weiterhin bei dem gesamten REFILLS Projektteam und den Kolleginnen und Kollegen für die spannende und lehrreiche Zeit. Auch bei den wissenschaftlichen Mitarbeitenden des IGMRs möchte ich mich für deren wunderbare Unterstützung bedanken und dafür, dass ich die familiäre Atmosphäre des Instituts miterleben durfte.

Weiterhin bedanke ich mich bei meinen Freundinnen und Freunden sowie meinen großartigen Schwestern, meinen Eltern und Schwiegereltern für deren Interesse an meiner Arbeit sowie die aufbauenden Worte in anstrengenden Zeiten. Abschließend danke ich von ganzem Herzen meiner wunderbaren Frau Zisa für ihre starke Unterstützung, Geduld und Liebe.

Veronika Zumpe Augsburg, im Februar 2023

Table of Content

List of Symbols and Indices

In the following work, bold lower-case letters represent vectors and bold upper-case letters are matrices.

Latin Lower Case

a	Joint axis	$[-]$
$\boldsymbol{a}$	Joint axis orientations	$[-]$
c_{φ}	Cosine of angle φ	$[-]$
d	Joint variable for prismatic joints	$[m]$
$\boldsymbol{d}$	Offset vector	$[m]$
det	Determinant	$[-]$
dof	Degree of freedom	$[-]$
$\boldsymbol{error}$	Error vector	$[rad\ or\ m]$
f	Degree of freedom of a joint	$[-]$
$\boldsymbol{g}$	Goal type for desired position vector	$[-]$
$gain$	Gain factor	$[-]$
i	Counter	$[-]$
j	Counter	$[-]$
k	Counter, Number of IK calculation calls	$[-]$
l	Scalar link length	$[m]$
$\boldsymbol{l}$	Link length set	$[m]$
m	Dimension, counter, number of possible link length sets, mass	$[-\ or\ kg]$
n	Counter, Number of joints	$[-]$

nr	Number of solution sets	$[-]$
$\boldsymbol{o}$	Axis around a vector must be rotated	$[-]$
$\boldsymbol{o}_i^j$	Position vector, transformation vector from i to j	$[m]$
$\boldsymbol{p}^i$	Desired position, position vector expressed in frame i	$[m]$
$\tilde{\boldsymbol{p}}$	Homogenous coordinates of point $\boldsymbol{p}$	$[m]$
$\boldsymbol{q}$	Joint parameters	$[rad\ or\ m]$
r	Possible kinematic schemes	$[-]$
$\boldsymbol{r}_{i,j}$	lever length, vector pointing from origin of frame i to origin of frame j	$[m]$
s	Step size for link length discretization, task scaling factor	$[m\ or\ -]$
s_φ	Sine of angle φ	$[-]$
t	Time	$[s]$
$\boldsymbol{v}$	Velocity	$[m/s]$
x	Value of the referring coordinate axis	$[m]$
$\boldsymbol{x}$	x unit vector of the coordinate system	$[m]$
y	Value of the referring coordinate axis	$[m]$
$\boldsymbol{y}$	y unit vector of the coordinate system	$[m]$
z	Value of the referring coordinate axis	$[m]$
$\boldsymbol{z}$	z unit vector of the coordinate system	$[m]$

Latin Upper Case

Ai	Joint axis i	$[-]$
$\boldsymbol{A}_i^j$	Homogenous transformation matrix for transformation from i to j	$[-]$
$\boldsymbol{C}_{ji}$	Matrix with stored kinematic schemes, first join is j type with axis in i direction	$[-]$
E	Energy	$[Nm]$
$\boldsymbol{I}$	Identity matrix	$[-]$
$\boldsymbol{J}$	Jacobian	$[-]$
$\boldsymbol{L}$	Matrix for link length combination sets	$[m]$
$\boldsymbol{O}_i$	World frame, coordinate system origin i	$[-]$
P_i	Prismatic joint with axis in i direction	$[-]$
$\boldsymbol{P}$	Matrix for the desired goal positions	$[-]$
R_i	Revolute joint with axis in i direction	$[-]$
$\boldsymbol{R}_i^j$	Rotation matrix, matrix for reorientation from frame i to frame j	$[-]$
SE	Special Euclidian group	$[-]$
SO	Special orthonormal group	$[-]$
$\boldsymbol{S}$	Skew symmetric matrix	$[-]$
$\boldsymbol{T}_i^j$	Transformation matrix from frame i to j	$[-]$
$\boldsymbol{W}$	Matrix specifying the saturated joints	$[-]$

Greek Lower Case

α	Angle for rotation of coordinate system	$[rad]$
β	Angle for rotation of coordinate system	$[rad]$
γ	Angle for rotation of coordinate system	$[rad]$
$\boldsymbol{\gamma}$	Joint forces and torques	$[N\ or\ Nm]$
θ	Angle between two vectors	$[rad]$
$\boldsymbol{\tau}$	End-effector forces and torques	$[N\ or\ Nm]$
φ	Joint variable for revolute joints	$[rad]$
ϕ	Angular error	$[rad]$
ω	Angular velocity	$[rad/s]$

Indices

In the following work, indices printed in normal type are invariant and indices printed in italics are variable.

3	Rank of identity matrix equals three
all	All elements of the referring parameter
b	Base, drive concept with relocated components
cur	Current value of referring parameter
des	Desired parameter value
e	End-effector
g	Number of goal positions
h	Human
i	Counter

id	Identical
j	Counter
joints	Robot joints
kin	kinetic
lim	Limits
l	Payload
links	Robot links
max	Maximum value
min	Minimum value
n	Number of joints
N	Null space
nr	Counter, number of referring parameter
o	Angular component
ori	Orientation
p	Linear component
pr	Parameter for calculation preparation
prio	Priority kinematic schemes
r	Robot
rel	Relative
s	Step, standard drive concept
SNS	Parameter being obtained by SNS algorithm
TCP	Tool center point
tol	Tolerated value of referring parameter

type	Joint type
x	Projection on the x axis, orientation of axis aligned with x axis, x component of referring parameter
y	Projection on the y axis, orientation of axis aligned with y axis, y component of referring parameter
z	Projection on the z axis, orientation of axis aligned with z axis, z component of referring parameter

List of Abbreviations

3d	Three dimensional
CAD	Computer-aided design
CHS	Circular hollow section
dof	Degree of Freedom
F	Fixed joint
FK	Forward kinematics
GUI	Graphical user interface
HRC	Human robot collaboration
HRI	Human robot interaction
IFR	International Federation of Robotics
IGMR	Institute of Mechanism Theory, Machine Dynamics and Robotics
IK	Inverse kinematics
nnz	Non-zero elements
P	Prismatic joint
R	Revolute joint
RHS	Rectangular hollow section
SCARA	Selective compliance assembly robot arm
SLM	Selective laser melting
STL	Stereolithography
TCP	Tool center point
wrt	With respect to

1 Introduction

Over the last years and especially with focus on the future, service robotics and home robotics play an important role in the field of automation. Compared to the market of industrial robots, the service robotic market shows a significantly greater growth potential for the following years. New robotic systems in the field of professional service robotics, service robotics and home robotics will be needed and a great increase of applications in non-industrial environment is predicted by trend analyses and studies such as [KM20; MGP20; WPK18]. Consequently, the need for suitable systems will grow in the coming years and in the long term we will also encounter such systems in our everyday lives.

Since for these applications the environment and set-up of the robot changes from industrial surroundings to other business environments and private households, suitable robotic systems meeting the resulting requirements face different challenges and another focus in the development compared to the design of industrial robots. Aspects such as safe human-robot collaboration are becoming even more important, so that in addition to the use of sensors, a system that is inherently safer from mechanical point of view also offers great advantages, and this is for instance approachable by means of lightweight construction design. Another challenge is the cost pressure, as the robotic systems in the service robotics and especially in the home robotics area must become significantly cheaper compared to industrial systems, so that they are more accessible. This leads to a design approach with the focus that the robot must not reach the highest possible performance, but instead be good enough to meet the requirements. In general, robots are known for their flexibility of use, however, the suitability of a system for a specific task is highly dependent on the robot's kinematics. To save costs on the one hand and to develop the system specifically for the applications and surroundings on the other, it is obvious that kinematics other than for instance the standard six-axis articulated robot can offer a promising solution. Due to ever shorter development times and to be able to compete on the market, efficient development of new systems and the set-up of corresponding prototypes is indispensable.

The following work deals with these important issues and offers a solution approach for semi-automated task-based robot design that can be integrated into standard processes in terms of methodology and workflow to enable rapid and targeted development and to support the engineer during the design process. In the subsequent Chapter 1.1, the challenges are discussed first from an overarching perspective, followed by Chapter 1.2 in which the approach of this work is explained to master the challenges mentioned.

1.1 Problem definition

The design of robotic systems requires multidisciplinary expertise since they are complex and highly developed mechatronic systems making use of synergies primarily between mechanical design, electrical design and information technology [Mai19]. However, this work mainly addresses the development of such systems from a mechanical point of view.
The development of new robot kinematics involves the following steps: First, the motion task must be defined based on the requirements. This is followed by the type synthesis in which suitable kinematics consisting of links and joints are determined and the corresponding drive positions can be defined. In the subsequent dimensional synthesis, the design parameters are determined so that the qualitative solution is further defined and for instance concrete link lengths are specified so that the rough dimensioning of the system is carried out. Afterwards, the system is further developed regarding a production- and load-oriented design and a first prototype can be generated. During the process, the intermediate results must always be analyzed, and the work steps may need to be repeated iteratively until an appropriate system has been developed.
To design suitable systems, it is therefore not only necessary to have know-how in component development, but also a profound knowledge in the field of robot kinematics, so that the developed robot arm can perform the required motion task under consideration of the environmental conditions. To make the required expertise accessible for the engineers and to fulfil the higher-level requirements mentioned in the introduction, it is necessary to support them in the development of new systems, both in terms of methodology and tools provided. Despite digitalization and the increase in

computing power, today's development processes and tools still have great potential for optimization. Kinematics development has been a field of research for many decades, but without the appropriate expertise, application of that is often challenging and, in most cases, it requires a serious amount of time. To address this issue, a methodology and tool for semi-automated task-based robot design has been developed within this work and the chosen approach is described in the following Chapter 1.2.

1.2 Approach

The objective of this work is to develop a method for task-based, semi-automated kinematic development of serial robot structures for the service robotics sector, and to implement it in a tool to support the engineer and make the development process more efficient. To better understand the context, Chapter 2 first discusses the basic terms of the robotic systems under consideration and the mathematical and kinematic principles. Subsequently, the solution structure of the developed methodology is presented in Chapter 3, so that based on an initial general overview, the concrete design and implementation can be dealt with in the following. To better understand the basic challenges for the development of new robotic systems in the service robotics sector, Chapter 4 first provides a rough overview of the corresponding requirements that must generally be kept in mind during the entire design process. This is followed by the three main chapters that address the core of the developed semi-automated approach for robotic systems design. The pre-processor (Chapter 5) helps to describe the motion task, the solver (Chapter 6) processes the data automatically and performs both, the type synthesis and dimensional synthesis, and finally the post-processor (Chapter 7) presents the solution options to be evaluated and selected. The three chapters are all structured in the same way. First, the methodological contexts are discussed and then the contents are described in more detail, followed by a simple application example to understand the presented approach. To evaluate the method and the implementation, the procedure developed was applied for the design of a robotic system within the REFILLS funding project as described in Chapter 8.

Chapter 9 gives a summary of the most important aspects of this work and Chapter 10 provides an outlook on how the potential of semi-automated robot development can be exploited in the future and which topics should be further explored.

After the structure of the following work has been described, the next Chapter 2 deals with the corresponding terminological and mathematical basics required.

2 Fundamentals of serial robotic systems

The methodology and implementation of task-based serial robot design developed in this thesis is based on the fundamentals of robotics and kinematics, so the essential basics are briefly described in this chapter. It is divided into two main parts, namely Chapter 2.1, which deals with the systems from a component point of view and explains basic kinematic terms and Chapter 2.2, which deals with the mathematical background.
In Chapter 2.1.1 the basic components of the robotic systems are described, and a brief overview of performance characteristics is provided in Chapter 2.1.2. Further, the nomenclature and ways of describing and visualizing kinematic structures are mentioned and the different types of workspaces are explained in Chapter 2.1.3.
The mathematical relationships are then discussed in Chapter 2.2. It provides an overview on the degree of freedom of kinematics, followed by describing the general pose of a rigid body in space, and describing the homogeneous transformation. This is followed by inverse kinematics (IK) and forward kinematics (FK) explanations and finally, a brief overview of the basics of differential kinematics is given in the last sub-chapter.
After the structure of the chapter has been outlined, robotic systems and basic characteristics are discussed below to provide a general overview.

2.1 Robotic systems and basic characteristics

This chapter gives an overview of robotic systems, their main components and set-up. Also, it points out some differences between industrial robotics and service robotics and illustrates some of their characteristics and their degree of fulfilment. The ISO/DIS 8373 draft international standard [ISO8373] defines the terms industrial robot and service robot as follows:

Industrial robot
An industrial robot is a reprogrammable multipurpose manipulator, including robot actuators controlled by the robot controller. It is programmable and includes any auxiliary axes that are integrated into the kinematic solution.

The system can either be stationary or mobile and is used in industrial automation applications.

Service robot

A service robot is a "robot that performs useful tasks for humans or equipment excluding industrial automation applications. Industrial automation applications include, but are not limited to manufacturing, inspection, packaging, and assembly. While articulated robots used in product lines are industrial robots, similar articulated robots used for serving food are service robots." [ISO8373, p. 2]

To have a better understanding while working on robotic systems, in the following the term industrial robots refers to systems in a classical industrial environment, while the term service robotics refers to all other applications to be automated by robotic systems. Applications of service robotics are different from industrial ones, which leads to various system requirements. While in industrial robotics characteristics like accuracy, greater payloads, high velocity, and acceleration are of great importance, for service robotics aspect like safety, low-cost and lightweight design are more relevant. Figure 2.1 gives an overview of the qualitative demands of some characteristics for service robotics compared to industrial robotics.

In general, industrial robots, as the name suggest, can be found in industrial environment with clearly defined and well-known surroundings. The systems have a long lifetime, and usually they provide a high accuracy and process speed. Most of the time they are closed systems with safety features like safety fences or light barriers. A physical interaction between the moving robot and a human is usually not foreseen. With service robotics this can be different. Here, human robot collaboration and interaction can be of importance, which leads to considerably changed requirements and system characteristics as depicted exemplarily in Figure 2.1. Furthermore, there is a higher demand on the flexibility and mobility of the system, since on the one hand the environmental conditions can change and on the other hand the system is to be used in a more versatile way. The ratio of payload to own weight is supposed to be lower, so that light weight design becomes even more important for service robotics systems. In addition, service robotics

systems need to be cheaper to be affordable, and according to the application, it must be evaluated which requirements must be prioritized.

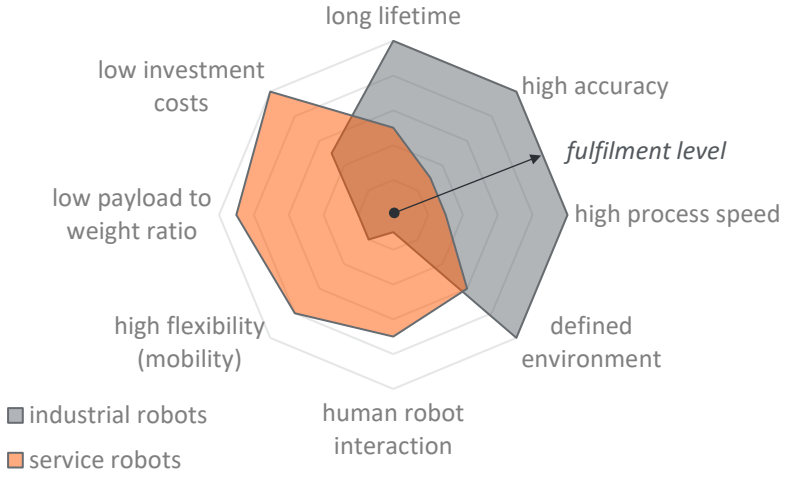


Figure 2.1 Qualitative characteristics of industrial and service robotics

To summarize, new markets and application possibilities lead to altered boundary conditions and the need for systems with characteristics different from those of classic industrial robotics as mentioned before. Still, there are a lot of similarities of the systems, especially regarding general structure and components as well as mathematical contents. The components and contents that are important for the following work are briefly described below in Chapter 2.1.1.

2.1.1 Components of serial robotic systems

On a higher level a serial robotic system can be divided into the robot arm, robot tool, control pc, program interface, power supply and, if necessary external sensor devices [Mai19]. The components are visualized in Figure 2.2. Having a closer look at the robot arm itself, it is composed of further components such as the robot base, joint units and drive trains, structural parts of links, and the flange at the last link of the robot. The most important components for this work are briefly described in the following.

Drive train

The drive train can be subdivided into actuator, gear unit and sensor devices. Typical gear units for industrial robots are harmonic drive gearboxes, cyclo gear units or planetary gear units. To monitor the motion of the robot joints and to control the robot, encoders or resolvers can be used.

Sensor devices

In addition to encoders to detect the error between target and actual position of the robot, more sensor devices like proximity sensors or joint torque sensors can be used. This is especially essential for service robots that work side by side with humans to detect collisions or limit the maximum torque that can be applied. Also, they can be used to detect process forces.

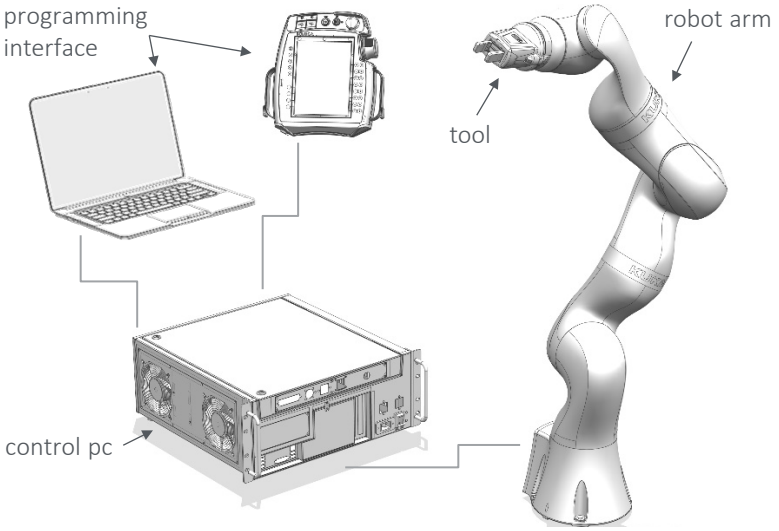


Figure 2.2 Components of a robotic system

Structural parts

Structural parts of the robot are components that link one joint of the kinematic structure to the following joint. They have a decisive influence on the shape and weight of the robot and guide forces and torques from the base to the robot's flange or tool, where the process forces are to be applied.

Additionally, the structural parts are the basis to mount all robot components and optionally covers, and they define the robot's links.

Regional structure and robot wrist

A general serial robot kinematic allows the positioning and orientation of the end-effector in space, depending on the degree of freedom and joint arrangement. According to the application and purpose, it may be useful to divide the kinematic scheme into two parts, namely the regional structure, also called main axes, for positioning the end-effector as highlighted in orange in Figure 2.3, and the secondary axes visualized in blue, that form the robot's wrist, which enables to change the orientation of the end-effector [VDI2861-2]. Depending on link lengths and joint arrangement, a joint angle movement of secondary axes can also lead to a change in the end-effector's position and not only to a different orientation. Therefore, it is important to consider whether a subdivision is appropriate and what requirements must be fulfilled by the wrist to ensure the desired behavior.

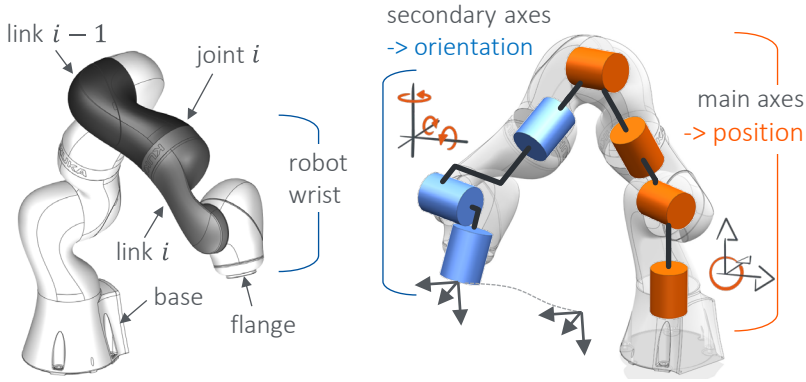


Figure 2.3 Components of a robot arm

In the context of this work, the focus is intentionally on the regional structure of kinematics, as it is primarily the reachability in the context of given applications that is investigated. Depending on the task to be automated, the wrist as well as the tool can vary considerably and are dependent on a variety of boundary conditions, therefore are not considered in detail in the context

of the presented and implemented methodology. More information on the kinematic scheme and the resulting workspaces are given in Chapter 2.2.1.

After briefly explaining the most important robot components, the following Chapter 2.1.2 discusses characteristic properties of robotic systems and how these can be used to identify a suitable system for a given application.

2.1.2 Robot performance characteristics

If a task must be automated by a robotic system, one approach is to choose a proper, already existing robot for the handling tasks. Depending on the application, the given constraints and the environmental conditions, some robots are more appropriate than others. To identify if a system fits the given technical requirements, there are performance characteristics that can be consulted to ensure that the chosen robot is suitable. Some of the important performance characteristics following [Raa86; Vol86] are briefly described below.

Degree of freedom (dof)

The degree of freedom of a serial robot indicates the number of independent movements that are possible for a rigid body in relation to the reference system. In general, the degree of freedom for serial robots corresponds to the number of axes to be driven and can be calculated using Grübler's equation (2.1). Further explanation as well as the difference between the degree of freedom of a robot arm and the operated degree of freedom in space is given in Chapter 2.2.1.

Kinematic scheme

The kinematic scheme of a robot determines the mobility of the robot arm and is basically defined by number, type, and arrangement of the joints. A more detailed description is given in Chapter 2.2.1.

Joint limits

Due to cable guiding through the robot structure and the mechanical set-up, joints often have axes limits. In case of revolute joints (R), the axis range is often limited to less than $360°$ and in case of prismatic joints (P), the maximum stroke length is limited, for instance, by the length of the linear guide. Joint limits reduce the actual workspace that would theoretically be possible with the kinematic structure under consideration.

Workspace

Depending on the degree of freedom, the kinematic scheme and the link lengths, the workspace of the robot arm is defined. Including joint limits and self-collision of the robot's interference structure leads to the robot's reachable workspace, which is the workspace that can be achieved even under the physical boundary conditions that exist. A more detailed explanation as well as different types of workspaces are given in Chapter 2.1.3.

Payload

The payload of a robot determines the maximum permissible load at a specific point of the robot. The nominal load is the combination of the tool load and the payload that is often referred to the tool center point (TCP) and can range from a few kilograms to more than a ton in the industrial robotics sector. Depending on the payload of the robot, correspondingly heavy objects can be handled, or process forces can be applied.

Accuracy

Depending on the application, the robotic system must maintain certain accuracies. The positioning accuracy, path accuracy and repeatability of the system are often considered to verify whether it meets the specified requirements for the task to be automated.

Velocity and acceleration

The cycle time of the system depends on the velocity and acceleration, so that high speeds lead to a better amortization. On the other hand, higher

speed reduces accuracy, so it is important to decide which factor should be prioritized regarding the application requirements.

Robot control

The robot control determines the system's application possibilities and integration options. Depending on the application, certain functionalities such as Cartesian motion may be required, so this must be supported by the robot control and the required interfaces must be provided.

Safety requirements

When humans and robots work together or side by side, safety plays a more important role than it already does within classical application set-ups. Thus, different safety features such as lightweight design or additional sensors may be required on the system.

The evaluation criteria or performance characteristics can be sorted depending on the prioritization of the given application and can be classified in two different groups, which are the necessary requirements and the desirable criteria. In summary, the specifications of the robot must match the requirements of the application to be automated. To facilitate the selection, there are on the one hand classifications of jobs that can be assigned to the corresponding performance characteristics or on the other hand catalogues to support this process, which are both based on the characteristics but are not explained in detail here. More information about performance characteristics can be found in the literature [Raa86; VDI2861-2; Vol86] and in [CDE12; Hes16; Sch87; TS84; ZLZ10] focusing on selecting suitable robot systems for a specific task or application.

If no suitable robot can be identified or a new development must take place for other reasons, the system must be designed so that the specifications of the robot match the given requirements. In this case, the performance characteristics mentioned above can be used to define the list of requirements. If also the task is directly considered during kinematic design and development, this is referred to as task-based robot design, which is

explained in more detail in Chapter 3 and Chapter 5, as the method developed in the context of this work focuses on that approach.
The most important criteria in this work are the constraints regarding mechanical restrictions like minimum or maximum link lengths, approved joints, and joint arrangements, what can be summarized on higher-level characteristics that significantly influence the workspace and the robot's motion behavior. Therefore, kinematic chains and workspaces of robots are discussed among others in the following Chapter 2.1.3.

2.1.3 Kinematic structures and workspaces

The capability of a robot to fulfill tasks and to change the pose of the tool center point in space is enabled by its kinematic scheme. It defines the robot type and basic characteristics like mobility as well as the qualitative workspace. There are three basic visualization possibilities for kinematic relationships and the design of systems, namely the kinematic chain, the kinematic scheme, and the dimensioned drawing. The first two are mainly for quick and intuitive understanding of the motion context, and the last one is aimed at realistic design.

Kinematic chain

A kinematic chain is mainly used for a quick and easy understanding of the kinematic interrelationship of a system and comprises information like the number of links, the number and type of joints and their degrees of freedom, as well as their connection and arrangement to each other. Also, it defines which links are forcibly guided. However, it contains no information about the function of the links or their kinematic dimensions. [Vol78]
Thus, a kinematic chain is a sequence of links that are connected by joints, which define the axes of motion of the robot. They can be divided into two main groups – open chains which are also called serial kinematics, and closed chains, that describe parallel kinematics. Within this work only serial kinematics are considered. Based on the kinematic chain the conceptual design of a kinematic can be further defined by a kinematic scheme, which is explained in the following.

Kinematic scheme

The functionality of the kinematics can be illustrated and described by using a kinematic scheme or also called kinematic diagram. It is a simplified representation of the system and does not give any information on the concrete design but only of the respective function of the links. [Vol78]

For example, the base link and the end-effector link can be defined. With serial kinematics, the first link is connected to the base frame or base link, links in between are connected to two joints and thus to two other links, and the last link is on one side connected to the last joint of the chain and on the output the reference frame of the end-effector is attached. In parallel kinematics, each link member is connected to at least two other members and the sequence of the links form a closed loop. Also, there are systems that combine both types of kinematic chains, such as a serial articulated robot arm that is extended by a parallelogram at its regional structure [KCH15; SVS09].

As already mentioned in reference to Figure 2.3, robot kinematics can be divided into the regional structure and the local structure, based on their different functions. The axes of the robot joints that mainly contribute to positioning the end-effector are called main axes [Raa86], primary axes, or principle axes and are highlighted orange in Figure 2.3. The spatial position can be defined in Cartesian space via the three coordinates x, y and z.

For the orientation of the end-effector in space, the secondary axes or minor axes are responsible and the orientational transformation can be described for instance using a rotation matrix. The minor axes specify the local structure, which is the robot's wrist as visualized blue in Figure 2.1. The wrist equips the robotic manipulator with dexterity.

Depending on the arrangement of the joint axes of a mechanism, it can be classified in different types of construction that allow a certain mobility. The kinematic diagram helps to understand principal solutions, their motion capabilities and to visualize them without much effort. Once the kinematic scheme is defined, the actual design of the system can be developed and represented in the dimensioned drawing that is shortly explained in the following.

Dimensioned drawing

Unlike the kinematic chain or the kinematic scheme, the dimensioned drawing or design is not about understanding the motion relationships as intuitively as possible. Instead, it is about the actual design, respecting the loads, taking constructive aspects into account and visualize how the system will look like in reality.

An example of a kinematic chain, the kinematic scheme and the final design visualized by the dimensioned drawing are shown in Figure 2.4.

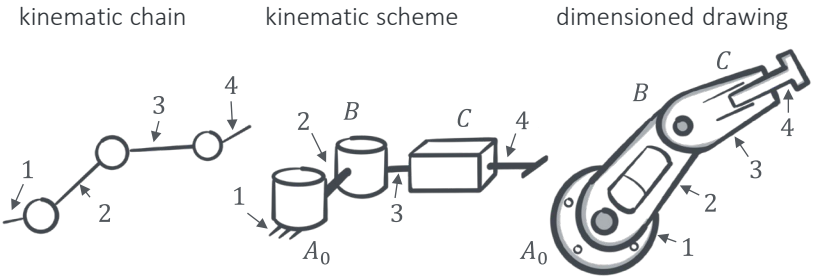


Figure 2.4 Kinematic chain, kinematic scheme, and dimensioned drawing

Since the robot joints have a significant influence on the motion capabilities and kinematic properties, they are briefly discussed below.

For serial kinematic chains the relative movement of two consecutive links is only dependent on their connecting joint and in general each joint adds one further degree of freedom to the kinematic structure, if the degree of freedom of that joint equals one [SVS09].

The most common joints are revolute joints R and prismatic joints P with both one degree of freedom. By using a prismatic joint, a translational motion is obtained between the two links connected to the joint and by using a revolute joint, the connected links can move rotationally relative to each other around the referring joint axis. Since in most systems only revolute and prismatic joints are used reasoned in their reliability and compactness characteristics [SVS09] and due to the advantages in terms of complexity and use of standard components, the following work focuses on kinematics built up by combinations of those two joint types and the others are not

considered. Via combination of R and P joints with different arrangements in order and orientation, a set of kinematic schemes can be generated [Hes98; SVS09].

In the following, typical kinematics for the regional structure determined by their main axes are listed up and the kinematic scheme as well as the resulting qualitative workspace are visualized in Figure 2.5.

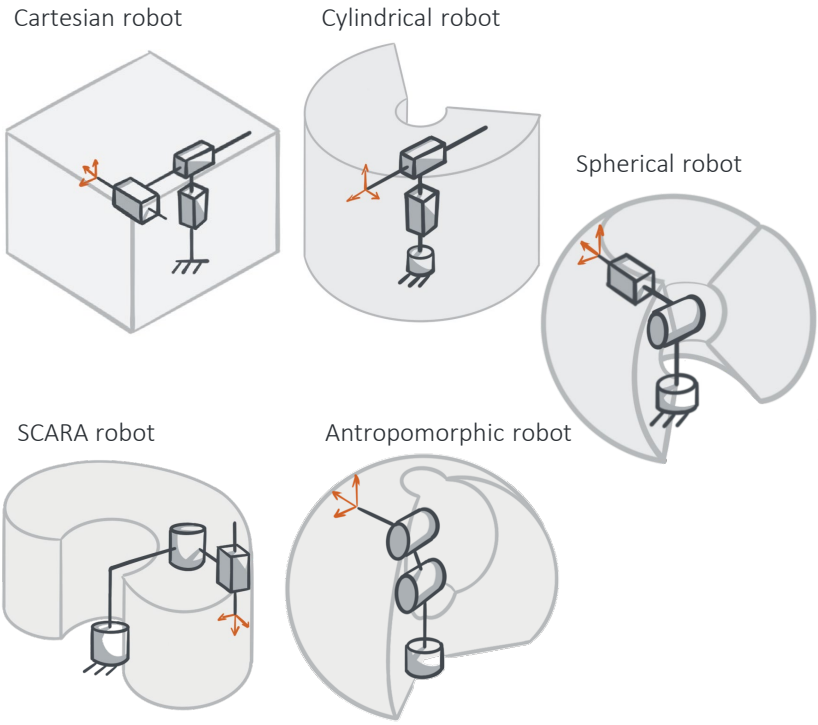


Figure 2.5 Kinematic scheme and qualitative workspace of typical robots

Cartesian robot

A cartesian robot consists of three consecutive orthogonal prismatic joints, leading to a cuboid workspace with good stiffness properties and a constant accuracy throughout the workspace. Typical applications for cartesian robots are handling or assembly tasks. [SVS09]

Cylindrical robot

A cylindrical robot structure consists of a revolute joint in z direction and two consecutive perpendicular prismatic joints, and the qualitative workspace has a cylindrical shape as the kinematics designation implies. With increasing displacement between the robot's end-effector and base, the generally good stiffness properties are decreasing. Cylindrical robots are often used for handling of large objects and to execute fast movements. [SVS09]

Spherical robot

Spherical robots consist of two consecutive perpendicular revolute joints and an orthogonal prismatic joint with the joint values corresponding to the parameters in spherical axis denotation. The workspace has the shape of an ellipsoid and the robot is typically used for machining applications [SVS09].

SCARA

The selective compliance assembly robot arm (SCARA) is a kinematic structure that consists of two revolute and one prismatic joint with all joint axes parallel to each other. Typically, the prismatic joint is the last joint in the kinematic chain, however, it often also makes sense to design the first axis as prismatic joint. The workspace has a cylindrical like shape as depicted in Figure 2.5.

Anthropomorphic / articulated robot

The anthropomorphic or articulated robot arm consists of three revolute joints, with the second and third joint axis parallel to each other and both perpendicular to the first joint axis. The workspace has a spherical volume, and this robot structure is used for a wide range of applications.

According to report [KM20] of the International Federation of Robotics (IFR), the articulated robot is the most widely used robotic system in the industrial sector with $251{,}287\ (67.3\ \%)$ new units installed worldwide in 2019, while from SCARA systems $54{,}033\ (14.5\ \%)$ units and from cartesian robots a total of $50{,}773\ (13.6\ \%)$ units were installed in 2019. As mentioned in the introduction, automation in new areas and the development of new service robots may have different requirements that lead to special

kinematics than those most widely used in the industry, so that it may be reasonable and necessary to deviate from the standard designs.

As depicted in Figure 2.5, the robot's workspace has different shapes, sizes, and characteristics depending on the robot kinematic and its dimensions. The development of new kinematics consists of the structural synthesis and the dimensional synthesis, among other things as further explained within this work. By finding a suitable kinematic scheme the structural kinematic synthesis leads to a qualitative workspace. The dimensional synthesis results in the dimensions of the design parameters so that based on this, the quantitative workspace is defined. According to the required tasks that must be done by the robot, some kinematics and their resulting workspaces are more suitable than others, which makes workspace considerations essential. There are different kinds of workspaces that vary by respecting diverse constraints and the most important classifications regarding the content of this work are explained in the following and refer to the literature [Mer06; SVS09; Vol78], in which further information can be found.

Workspace

A robot's workspace is defined by all positions that can be reached by the end-effector when executing all possible motions of the principal axes without respecting the physical geometry such as the robot's outer shape or joint limits. Thus, it is formed by all configurations that the end-effector of the robot can reach and is completely independent from a particular task.

Collision space

The collision space is the volume that is occupied not only by the end-effector but also by the robot's interference structure in all possible positions while moving all axes of the robot. To maintain a good space utilization the ratio of workspace to collision space must be as big as possible. For physical reasons, the collision space is always larger than the workspace.

Reachable workspace

The reachable or also called secondary workspace is the volume spanned by all positions that can be reached by the robot taking physical boundary

conditions like joint limits and the robot's interference structure into account. Thus, it includes all positions that can be reached with at least one orientation of the end-effector by moving the principal axis under realistic aspects.

Dexterous workspace

The dexterous or also called primary workspace is a subset of the reachable workspace. It includes all possible positions of the end-effector that can be reached with any arbitrary orientation.

Constant orientation workspace

The constant orientation workspace is the workspace that can be reached by the end-effector under one specified orientation.

Workspaces can be determined using analytical, geometric, or numerical methods. Depending on the complexity of the kinematics, the appropriate procedure must be selected. In the tool for serial robot kinematic development presented in Chapter 7.1, the workspace determination is carried out analytically, based on a discretization of the possible joint angles. Early work for workspace determination and characteristics is provided for instance in [GR82; Gup86; KW81; TS81], the impact of the design parameters like link lengths on the workspace and how to use them for optimization are presented in [CCP10; CL04; COC07; KB06; TS84; YL84], computer-aided workspace generation and evaluation is given in [ZBH07; ZCY19], and further methods for workspace generation and evaluation as well as a more detailed overview of the state of the art is provided in [COC08].

In this work, the reachable workspace and the constant orientation workspace are essential and become a decisive factor in Chapter 5.1 for the task-based robot design.

After having discussed robotic systems from a component point of view and their characteristics, kinematic representations and resulting workspaces, the mathematical interrelationships of serial kinematic structures are described below in Chapter 2.2.

2.2 Mathematical fundamentals for serial robot kinematics

For the implementation of the methodology developed in this thesis for the task-based generation of serial robot kinematics, some mathematical principles are used, which are briefly summarized in the following chapter. First, the determination of the degree of freedom of serial structures is addressed, followed by a general introduction to the pose of a rigid body in space. Afterwards a short overview of the homogeneous transformation, the forward kinematics (FK), inverse kinematics (IK), and finally the basics to calculate the geometric Jacobian are given. The contents are mainly based on the literature [SVS09] and more detailed explanations and derivations can be found there.

2.2.1 Degrees of freedom of serial robots

The degree of freedom (dof) of a spatial mechanism with n_{links} links including the base, n_{joints} joints with respective degree of freedom f, can be calculated using equation (2.1). The number of identical joint degrees of freedom f_{id} must be subtracted for joints with more than two links attached but equals zero in case of serial structures. If a serial kinematic system consists only of revolute and prismatic joints with $dof = 1$, the degree of freedom can be obtained directly by equation (2.2) and at the same time corresponds to the number of independent drives required to actuate the system. [Vol86]

$$dof = 6(n_{links} - 1) - 6n_{joints} + \sum_{i=1}^{n_{joints}} f_i - f_{id} \tag{2.1}$$

$$dof = n_{links} - 1 \tag{2.2}$$

As described in more detail in the next section 2.2.2, a rigid body without constraints has six degrees of freedom in space. This results in the fact that a robot arm needs at least six dof to fully define position and orientation of the end-effector's reference frame in space. If a robot has more than six degrees of freedom, such as the LBR iiwa, the system is called redundant.

Besides the classical redundancy with more than six degrees of freedom, a robot can be task redundant, specifically when it offers more degrees of freedom than are required to fulfill the given task. Redundant degrees of freedom can be used to consider additional constraints such as collision avoidance or efficient robot motion planning. It should be noted that the degree of freedom of the robot arm does not necessarily correspond to the degree of freedom that can be operated by it in space, as this depends on the axis orientation among each other. If, for example, one considers a kinematic scheme with four parallel axes of rotation, the system can position its end-effector in a plane orthogonal to the joint axes and orient it around the direction of the joint axes. The fourth joint axis, however, does not provide an additional degree of freedom in space, but a redundant degree of freedom in the spanned plane.

2.2.2 Pose of a rigid body

A rigid body in space has six degrees of freedom, three are translational and can be expressed via values of x, y and z along the coordinate axes, and three are rotational that can be expressed by rotation around the coordinate system axes. Thus, the pose of a body in space can be determined by specification of its position and orientation as visualized in Figure 2.6.

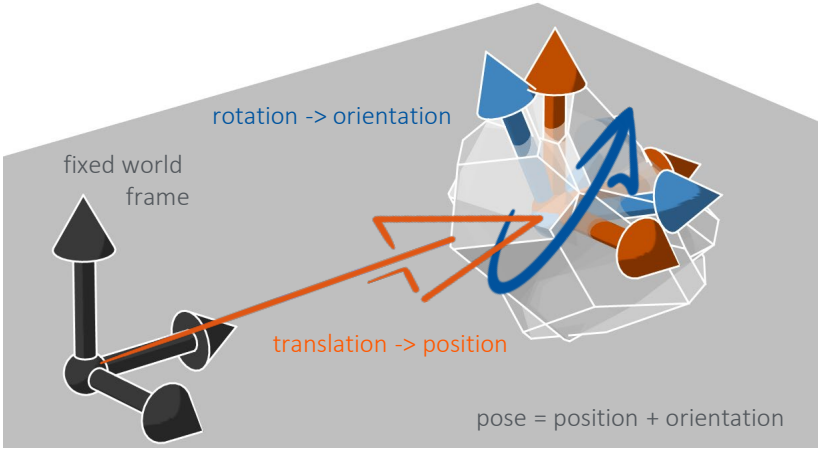


Figure 2.6 Pose of a rigid body in space

To describe poses of robot links or the end-effector with respect to the fixed world frame $\boldsymbol{O}$ or in relation to each other, orthonormal reference frames can be sticked to the links that are assumed as rigid bodies. Their position and orientation then can be mathematically expressed regarding the desired coordinate frame. The expression for the position $\boldsymbol{o'}$ of a reference frame $\boldsymbol{O'}$ with respect to the world frame's unit vectors $\boldsymbol{x}, \boldsymbol{y}$ and $\boldsymbol{z}$ is given in (2.3) and visualized in Figure 2.7.

$$\boldsymbol{o'} = o'_x \cdot \boldsymbol{x} + o'_y \cdot \boldsymbol{y} + o'_z \cdot \boldsymbol{z} = \begin{bmatrix} o_x' \\ o_y' \\ o'_z \end{bmatrix} \tag{2.3}$$

The components of the unit vectors $\boldsymbol{x'}, \boldsymbol{y'}$ and $\boldsymbol{z'}$ in (2.4) of the reference frame $\boldsymbol{O'}$ are the projections to the unit vectors $\boldsymbol{x}, \boldsymbol{y}$ and $\boldsymbol{z}$ of frame $\boldsymbol{O}$ as shown in Figure 2.7.

$$\begin{aligned} \boldsymbol{x'} &= x'_x \cdot \boldsymbol{x} + x'_y \cdot \boldsymbol{y} + x'_z \cdot \boldsymbol{z} \\ \boldsymbol{y'} &= y'_x \cdot \boldsymbol{x} + y'_y \cdot \boldsymbol{y} + y'_z \cdot \boldsymbol{z} \\ \boldsymbol{z'} &= z'_x \cdot \boldsymbol{x} + z'_y \cdot \boldsymbol{y} + z'_z \cdot \boldsymbol{z} \end{aligned} \tag{2.4}$$

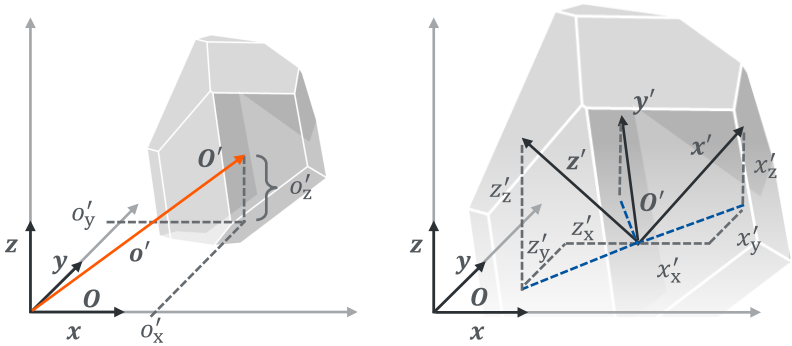


Figure 2.7 Pose of frame $\boldsymbol{O'}$ with respect to frame $\boldsymbol{O}$

The unit vectors of frame $\boldsymbol{O'}$ can be written in matrix notation with its components being the dot product of the referring unit vectors of frame $\boldsymbol{O}$ and $\boldsymbol{O'}$, which results in the rotation matrix $\boldsymbol{R}$ belonging to the special

orthonormal group $SO(m)$ with $m = 2$ for planar rotations, $m = 3$ for spatial rotations and for the second case given in equation (2.5). [SVS09]

$$\boldsymbol{R} = [\boldsymbol{x}' \quad \boldsymbol{y}' \quad \boldsymbol{z}'\,] = \begin{bmatrix} x'_{\mathrm{x}} & y'_{\mathrm{x}} & z'_{\mathrm{x}} \\ x'_{\mathrm{y}} & y'_{\mathrm{y}} & z'_{\mathrm{y}} \\ x'_{\mathrm{z}} & y'_{\mathrm{z}} & z'_{\mathrm{z}} \end{bmatrix} \tag{2.5}$$

$$= \begin{bmatrix} \boldsymbol{x}'^{T} \cdot \boldsymbol{x} & \boldsymbol{y}'^{T} \cdot \boldsymbol{x} & \boldsymbol{z}'^{T} \cdot \boldsymbol{x} \\ \boldsymbol{x}'^{T} \cdot \boldsymbol{y} & \boldsymbol{y}'^{T} \cdot \boldsymbol{y} & \boldsymbol{z}'^{T} \cdot \boldsymbol{y} \\ \boldsymbol{x}'^{T} \cdot \boldsymbol{z} & \boldsymbol{y}'^{T} \cdot \boldsymbol{z} & \boldsymbol{z}'^{T} \cdot \boldsymbol{z} \end{bmatrix}$$

Since the rows of the rotation matrix $\boldsymbol{R}$ consist of orthonormal unit vectors, their dot product equals zero (2.6) and they have unit norm (2.7), which defines $\boldsymbol{R}$ as orthogonal matrix with $det(\boldsymbol{R}) = 1$ and its identity $\boldsymbol{I}$ as defined in equation (2.8). Further, the inverted rotation matrix equals the transposed rotation matrix (2.9), what results from post-multiplying the inverted rotation matrix $\boldsymbol{R}^{-1}$ on both sides of equation (2.8).

$$\boldsymbol{x}'^{T}\boldsymbol{y}' = 0, \qquad \boldsymbol{y}'^{T}\boldsymbol{z}' = 0, \qquad \boldsymbol{z}'^{T}\boldsymbol{x}' = 0 \tag{2.6}$$

$$\boldsymbol{x}'^{T}\boldsymbol{x}' = 1, \qquad \boldsymbol{y}'^{T}\boldsymbol{y}' = 0, \qquad \boldsymbol{z}'^{T}\boldsymbol{z}' = 0 \tag{2.7}$$

$$\boldsymbol{R}^{T}\boldsymbol{R} = \boldsymbol{I}_3 \tag{2.8}$$

$$\boldsymbol{R}^{T} = \boldsymbol{R}^{-1} \tag{2.9}$$

Arbitrary rotations can be described a product of elementary rotations, which refer to rotations around the axes x, y and z of a reference frame and are given in (2.10) for a rotation around the x axis, (2.11) for rotation around y, and (2.12) for rotation around the z axis, respectively. A visualization of the three elementary rotations is given in Figure 2.8.

$$\boldsymbol{R}_{\mathrm{x}}(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) \\ 0 & \sin(\gamma) & \cos(\gamma) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\gamma & -\mathrm{s}_\gamma \\ 0 & s_\gamma & c_\gamma \end{bmatrix} \tag{2.10}$$

$$\boldsymbol{R}_{\mathrm{y}}(\beta) = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix} = \begin{bmatrix} c_\beta & 0 & s_\beta \\ 0 & 1 & 0 \\ -s_\beta & 0 & c_\beta \end{bmatrix} \tag{2.11}$$

$$\boldsymbol{R}_{\mathrm{z}}(\alpha) = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_\alpha & -s_\alpha & 0 \\ s_\alpha & c_\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.12}$$

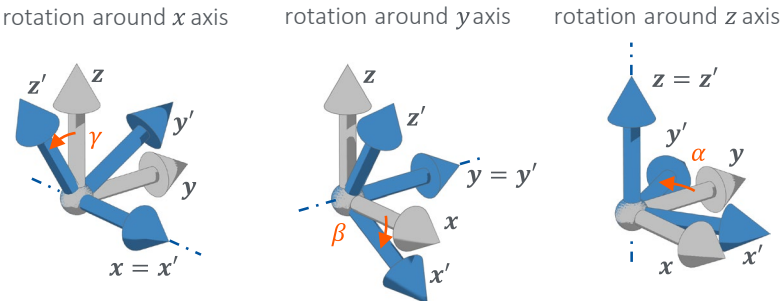


Figure 2.8 Elementary rotations

Summing up the previous findings and following [SVS09] there are three equivalent geometrical meanings of the rotation matrix as given in the following.

First, with its column vectors being the direction cosines of a rotated coordinate frame ***O'*** with respect to its reference frame ***O*** it characterizes the orientation between these previously mentioned frames (2.13).

$$\boldsymbol{O'} = \boldsymbol{RO} \tag{2.13}$$

Secondly, the rotation matrix maps the expression of a point in two different coordinate systems with the same origin and thus enables coordinate transformation (2.14).

$$\boldsymbol{p} = \boldsymbol{Rp'} \Leftrightarrow \boldsymbol{p'} = \boldsymbol{R^T p} \tag{2.14}$$

Thirdly, as operator, the rotation matrix $\boldsymbol{R}$ maps a vector expressed in a coordinate frame $\boldsymbol{O'}$ to a vector that is expressed in another coordinate frame $\boldsymbol{O}$ (2.15) and is also called rotation transformation or passive rotation.

$$\boldsymbol{p^O} = \boldsymbol{R^O_{O'}} \cdot \boldsymbol{p^{O'}} \tag{2.15}$$

In case of several consecutive rotations, rotation matrices can be summarized in one overall rotation (2.16), however, the sequence of operations must be followed.

$$\boldsymbol{R^0_2} = \boldsymbol{R^0_1 R^1_2} \tag{2.16}$$

To sum up the previous findings, a vector $\boldsymbol{p}$ described in a coordinate system $\boldsymbol{O'}$ can be expressed in the reference frame $\boldsymbol{O}$ by equation (2.17).

$$\boldsymbol{p^O} = \boldsymbol{o^O_{O'}} + \boldsymbol{R^O_{O'}} \cdot \boldsymbol{p^{O'}} \tag{2.17}$$

Based on these principles, the following Chapter 2.2.3 describes the homogeneous transformation, which can be used to describe poses of rigid bodies in relation to each other and to reference frames.

2.2.3 Homogenous Transformation

The pose of a rigid body or a point on this very body as depicted in Figure 2.9 can be described by sticking a coordinate frame $\boldsymbol{O_1}$ to that body and formulating the displacement with respect to a reference frame $\boldsymbol{O_0}$. As mentioned earlier, the pose consists of a translational and a rotational part and instead of looking at the individual transformations one by one, they can be combined in the homogenous transformation.

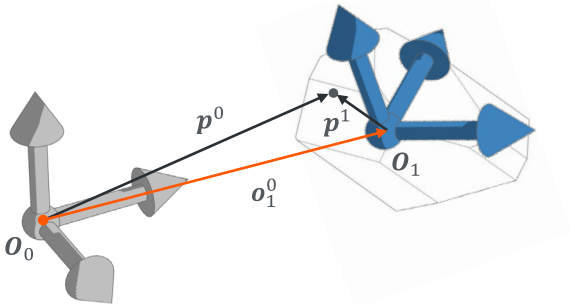


Figure 2.9 Rigid body transformation

To enable this, a rigid body transformation or transformation of points or vectors on that body can be represented by matrices and vectors in $\mathbb{R}^4$. Enlarging a point in $\mathbb{R}^3$ by 1 to the coordinates yields to a vector in $\mathbb{R}^4$ of the form $\tilde{\boldsymbol{p}}$, which are the homogeneous coordinates of a point defined by vector $\boldsymbol{p}$ in $\mathbb{R}^3$ as given in (2.18). [MLS94]

$$\tilde{\boldsymbol{p}} = \begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix} = \begin{bmatrix} p_{\mathrm{x}} \\ p_{\mathrm{y}} \\ p_{\mathrm{z}} \\ 1 \end{bmatrix} \tag{2.18}$$

The (4×4) homogenous transformation matrix $\boldsymbol{A}$ as given in equation (2.19) is part of the Special Euclidian group $SE(3) = \mathbb{R}^3 \times SO(3)$ and can describe the coordinate transformation between two frames, which consists of a translational and rotational part. [SVS09]

$$\boldsymbol{A}_0^1 = \begin{bmatrix} \boldsymbol{R}_0^1 & \boldsymbol{o}_0^1 \\ \boldsymbol{0}^T & 1 \end{bmatrix} \tag{2.19}$$

Based on equations (2.18) and (2.19), the affine transformation can then be represented by its linear form in equation (2.20).

$$\tilde{\boldsymbol{p}}^0 = \boldsymbol{A}_1^0 \tilde{\boldsymbol{p}}^1 = \begin{bmatrix} \boldsymbol{R}_0^1 & \boldsymbol{o}_0^1 \\ \boldsymbol{0}^T & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}^1 \\ 1 \end{bmatrix} \tag{2.20}$$

A combination of successive homogenous transformations can be summarized according to the given sequence. With the configuration $\boldsymbol{A}_2^1 \in SE(3)$ of frame $\boldsymbol{O}_2$ relative to frame $\boldsymbol{O}_1$, and $\boldsymbol{A}_1^0 \in SE(3)$ being the configuration of frame $\boldsymbol{O}_1$ relative to frame $\boldsymbol{O}_0$, respectively, the configuration of $\boldsymbol{O}_2$ relative to $\boldsymbol{O}_0$ can be denoted as given in equation (2.21).

$$\boldsymbol{A}_2^0 = \boldsymbol{A}_1^0 \boldsymbol{A}_2^1 = \begin{bmatrix} \boldsymbol{R}_1^0 \boldsymbol{R}_2^1 & \boldsymbol{R}_1^0 \boldsymbol{o}_2^1 + \boldsymbol{o}_1^0 \\ \boldsymbol{0}^T & 1 \end{bmatrix} \tag{2.21}$$

It follows from equation (2.21) that a point in any coordinate system can be expressed in the reference coordinate system $\boldsymbol{O}_0$ via several transformations by means of equation (2.22).

$$\tilde{\boldsymbol{p}}^0 = \boldsymbol{A}_1^0 \boldsymbol{A}_1^0 \dots \boldsymbol{A}_n^{n-1} \tilde{\boldsymbol{p}}^n \tag{2.22}$$

A reverse transformation is achieved by use of the inverted homogenous transformation matrix $\boldsymbol{A}^{-1}$ and is denoted in (2.23). Unlike the rotation matrix, the inverted homogeneous transformation matrix is not identical to its transposed one and must therefore be determined arithmetically.

$$\tilde{\boldsymbol{p}}^1 = (\boldsymbol{A}_1^0)^{-1} \tilde{\boldsymbol{p}}^0 = \begin{bmatrix} \boldsymbol{R}_1^{0T} & -\boldsymbol{R}_1^{0T} \boldsymbol{o}_1^0 \\ \boldsymbol{0}^T & 1 \end{bmatrix} \tag{2.23}$$

The homogeneous transformation provides the basics for determining the forward kinematics of serial robots and a brief overview is given in the following Chapter 2.2.4.

2.2.4 Forward kinematics

A serial kinematic scheme as explained in Chapter 2.1.3 describes the number, type, and arrangement of joints to each other, what significantly determines the degree of freedom and the mobility of the resulting robot. In a standard setup, each joint of the structure can be actuated individually, defining the configuration of the robot, and thus positioning and orienting its end-effector in space. The task of forward kinematics (FK) or also called direct kinematics is to describe this relationship mathematically so that the pose of the end-effector's frame $\boldsymbol{O}_{\mathrm{e}}$ can be expressed in a fixed coordinate frame $\boldsymbol{O}_{\mathrm{b}}$ sticked to the base. The transformation is a function of the joint variables $\boldsymbol{q} = [q_1, q_2, \dots, q_n]^T$ and is given in equation (2.24).

$$\boldsymbol{T}_{\mathrm{e}}^{\mathrm{b}}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{x}_{\mathrm{e}}^{\mathrm{b}}(\boldsymbol{q}) & \boldsymbol{y}_{\mathrm{e}}^{\mathrm{b}}(\boldsymbol{q}) & \boldsymbol{z}_{\mathrm{e}}^{\mathrm{b}}(\boldsymbol{q}) & \boldsymbol{p}_{\mathrm{e}}^{\mathrm{b}}(\boldsymbol{q}) \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{\mathrm{e}}^{\mathrm{b}}(\boldsymbol{q}) & \boldsymbol{p}_{\mathrm{e}}^{\mathrm{b}}(\boldsymbol{q}) \\ \boldsymbol{0}^T & 1 \end{bmatrix} \tag{2.24}$$

So, the forward kinematic equation maps joint variables from the joint space to the pose of the end-effector in cartesian space with respect to the robot kinematics.

For open or serial kinematic chains with $n + 1$ links being connected by n joints, link 0 can be defined as robot base, that is fixed according to the installation site in the base frame, and link n can be defined as the robot's last link with an end-effector attached to it.

The displacement and rotation between these coordinate frames of a generic serial chain can be described recursively by combining the homogenous transformations between the individual link coordinate frames in ascending order as given in (2.25) and visualized in Figure 2.10.

$$\boldsymbol{T}_n^0(\boldsymbol{q}) = \boldsymbol{A}_1^0(q_1)\boldsymbol{A}_2^1(q_2) \dots \boldsymbol{A}_n^{n-1}(q_n) \tag{2.25}$$

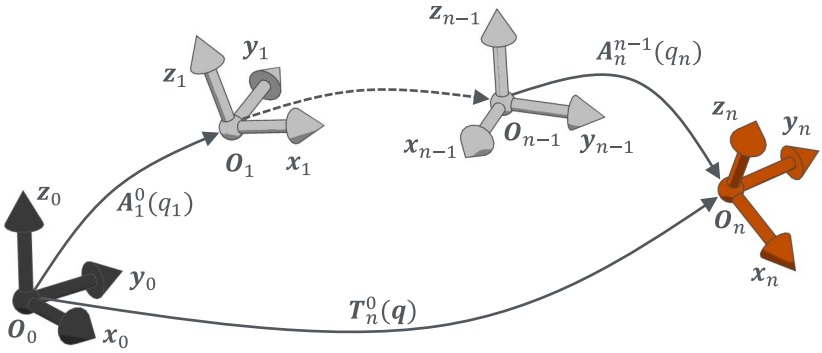


Figure 2.10 Direct kinematic relation following [SVS09]

The complete transformation between the base frame $\boldsymbol{O}_\mathrm{b}$ and the robot's end-effector frame $\boldsymbol{O}_\mathrm{e}$ is generated by pre-multiplying the generally constant transformation $\boldsymbol{T}_0^\mathrm{b}$, and postmultiplying the in most cases constant transformation $\boldsymbol{T}_n^\mathrm{e}$ to (2.25) which yields equation (2.26). With this equation, the end-effector pose of any serial kinematics can be determined for given joint values. Some examples of forward kinematics for typical robots can be found in [SVS09; Vol86].

$$\boldsymbol{T}_\mathrm{e}^\mathrm{b}(\boldsymbol{q}) = \boldsymbol{T}_0^\mathrm{b}\boldsymbol{T}_n^0(\boldsymbol{q})\boldsymbol{T}_\mathrm{e}^n \tag{2.26}$$

If a robotic system is to be developed or a suitable robot is to be identified, it must be ensured that the system fulfils the required task. In this case, the focus is therefore on determining the corresponding joint values for known end-effector poses or positions that describe the robot's task. This can be done using inverse kinematics and is briefly explained in the following Chapter 2.2.5.

2.2.5 Inverse kinematics

As mentioned in the last chapter, the direct kinematics equation enables the calculation of the robot's end-effector and this can be done, for example, based on the homogenous transformation as a function of the joint variables. Thus, a given set of joint variables leads to a unique pose of the end-effector. On the other hand, the inverse kinematic (IK) problem involves the reverse task, which means that the joint variables that lead to a desired pose of the end-effector are searched for. In general, there can exist an infinite number, a specific number, one or no solutions to the inverse kinematic problem.
To solve the inverse kinematic problem, there are basically two fundamental approaches, that are analytical closed-form solutions, and numerical solutions.

Analytical IK

To solve the inverse kinematic problem analytically, the spatial problem can be divided up by the principle of kinematic decoupling into solving the positioning problem and the orientation problem individually or dividing the main task into several planar sub-problems for which there are analytical solutions that can be determined by use of geometric laws [SHV20; SVS09]. Another possibility is to determine algebraic solutions based on the transformation matrices.
Analytical solutions are dependent on the robot kinematic, thus there is no standard solution for all arbitrary robot kinematics, and it is not guaranteed to always determine a closed-form solution. Other characteristics of this solution method are that all existing solutions can be determined, and the calculation is faster compared to numerical methods. Methods to solve the analytical IK, examples and detailed derivations and explanations are given in the literature [MLS94; Rie92].

Numerical IK

Numerical inverse kinematic solutions are iterative approaches, that in general require more computational effort compared to analytical calculation and only one of the possible solutions is determined [SVS09]. Furthermore, the solution found depends on the algorithm and the initial

joint values. Still, with increase of computational power and since this approach is applicable to all robot kinematics, it is a sensible method for solving the inverse kinematic problem. A detailed overview and review of numerical techniques is presented in [AL09].
Since an essential part of this work is the kinematic synthesis for specific tasks, which can result in several different suitable kinematic solutions, the numerical calculation on velocity level is an appropriate approach and is thus implemented in the design tool as further explained in Chapter 6.2 and related necessary fundamentals are explained in the following Chapter 2.2.6 in which the topic of differential kinematics is addressed.

2.2.6 Differential kinematics

The homogeneous transformation as outlined above in Chapter 2.2.3 describes the relationship between joint parameters and end-effector pose, so that the calculations of FK and IK are performed at position level. In this chapter, the relationship between joint velocities and end-effector velocities is described, since this is an essential part to solve the IK as used for the dimensional synthesis that follows in Chapter 6.2.
First, the rotation matrix $\boldsymbol{R}$ is derived, then the link velocities are discussed, so that finally the differential kinematics equation based on the geometric Jacobian $\boldsymbol{J}$ can be obtained, which maps the joint velocities to the end-effector velocities. Some advantages of the Jacobian are that it is an efficient tool to analyze robot kinematics, especially regarding their characteristics of mobility and the transmission behavior. It helps to detect singularities, analyze redundancy, generate solutions for inverse kinematic problems on velocity level and offers a direct relationship between joint torques and end-effector forces [SVS09].

Derivation of the rotation matrix

The velocity level is reached by derivation of the robot's pose (2.17) over time. Therefore, the rotation matrix is derived according to time, as given in the literature [SVS09] and explained in the following.
With the matrix product of the rotation matrix and its transposed resulting in its identity as given in (2.8), and the time dependence leading to the

expression $\boldsymbol{R}(t)\boldsymbol{R}^T(t) = \boldsymbol{I}$, equation (2.27) is obtained by derivation according to time. Defining the skew-symmetric matrix $\boldsymbol{S}$ as given in (2.28) and post-multiplying both sides of (2.28) with $\boldsymbol{R}(t)$ leads to equation (2.29) in which the time derivative $\dot{\boldsymbol{R}}(t)$ is expressed depending on $\boldsymbol{R}(t)$ itself [SVS09].

$$\dot{\boldsymbol{R}}(t)\boldsymbol{R}^T(t) + \boldsymbol{R}(t)\dot{\boldsymbol{R}}^T(t) = \boldsymbol{O} \tag{2.27}$$

$$\boldsymbol{S}(t) = \dot{\boldsymbol{R}}(t)\boldsymbol{R}^T(t) \tag{2.28}$$

$$\dot{\boldsymbol{R}}(t) = \boldsymbol{S}(t)\boldsymbol{R}(t) \tag{2.29}$$

Transforming a constant vector $\boldsymbol{p}'$ as given in (2.14) with respect to a time dependent rotation matrix and calculating its time derivative of this resulting relation $\boldsymbol{p}(t) = \boldsymbol{R}(t)\boldsymbol{p}'$ returns the expression (2.30) as a function of $\boldsymbol{R}(t)$. The expression $\boldsymbol{\omega}(\boldsymbol{t}) = \left[\omega_{\mathrm{x}}(t), \omega_{\mathrm{y}}(t), \omega_{\mathrm{z}}(t)\right]^T$ is the time dependent angular velocity of $\boldsymbol{R}(t)$ and, incorporating the mechanical background, leads to equation (2.31). Thus, the skew-symmetric vector product matrix $\boldsymbol{S}(t)$ (2.32) describes the vector product between $\boldsymbol{\omega}$ and the vector $\boldsymbol{p}(t) = \boldsymbol{R}(t)\boldsymbol{p}'$, which is a linear mapping. Consequently, equation (2.29) can be described as a function of $\boldsymbol{\omega}$ as given in (2.33), where the right-hand side of the equation can be rewritten as specified in (2.34) and will be used later for equation (2.39). [SVS09]

$$\dot{\boldsymbol{p}}(t) = \dot{\boldsymbol{R}}(t)\boldsymbol{p}' = \boldsymbol{S}(t)\boldsymbol{R}(t)\boldsymbol{p}' \tag{2.30}$$

$$\boldsymbol{v}(t) = \dot{\boldsymbol{p}}(t) = \boldsymbol{\omega} \times \boldsymbol{r} = \boldsymbol{\omega} \times \boldsymbol{R}(t)\boldsymbol{p}' \tag{2.31}$$

$$\boldsymbol{S}(t) = \mathbf{S}(\boldsymbol{\omega}(t)) = \begin{bmatrix} 0 & -\omega_{\mathrm{z}}(t) & \omega_{\mathrm{y}}(t) \\ \omega_{\mathrm{z}}(t) & 0 & -\omega_{\mathrm{x}}(t) \\ -\omega_{\mathrm{y}}(t) & \omega_{\mathrm{x}}(t) & 0 \end{bmatrix} \tag{2.32}$$

$$\dot{\boldsymbol{R}} = \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R} \tag{2.33}$$

$$\boldsymbol{R}\boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}^T = \boldsymbol{S}(\boldsymbol{R}\boldsymbol{\omega}) \tag{2.34}$$

These relationships can now be used to describe link velocities as explained in the following.

Link velocities

An arbitrary link i of a serial robot kinematic is connected to its prior link $i-1$ or base via a joint i and connected to its following link $i+1$ or end-effector by joint $i+1$ (see Figure 2.11).

The mathematical correlation between joint i and joint $i+1$ is given in (2.35) with frame $i-1$ connected to link $i-1$ at joint i of an arbitrary link i and frame i attached to joint $i+1$ of the aforesaid link i.

$$\boldsymbol{p}_i = \boldsymbol{p}_{i-1} + \boldsymbol{R}_{i-1}\boldsymbol{r}_{i-1,i}^{i-1} \tag{2.35}$$

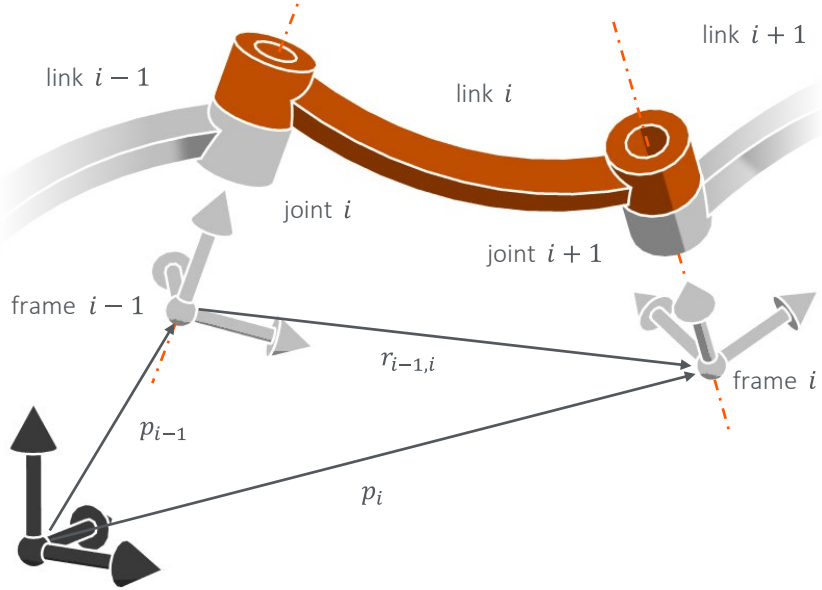


Figure 2.11 Generic link i of the robot's structure following [SVS09]

Linear velocities of link $\boldsymbol{i}$

Derivation of (2.35) after time leads to the linear velocity of the considered link $\boldsymbol{i}$ as expression (2.36) in dependence on translational and rotational velocities of link $\boldsymbol{i-1}$.

$$\dot{\boldsymbol{p}}_i = \dot{\boldsymbol{p}}_{i-1} + \boldsymbol{R}_{i-1}\dot{\boldsymbol{r}}_{i-1,i}^{i-1} + \dot{\boldsymbol{R}}_{i-1}\boldsymbol{r}_{i-1,i}^{i-1}$$

$$= \dot{\boldsymbol{p}}_{i-1} + \boldsymbol{v}_{i-1,i} + \boldsymbol{\omega}_{i-1} \times \boldsymbol{r}_{i-i,i} \tag{2.36}$$

Angular velocities of link $\boldsymbol{i}$

The rotation of link $\boldsymbol{i}$ with respect to a fixed base frame $\boldsymbol{O}$ can be expressed by the product of the referring rotation matrices as given in (2.37).

$$\boldsymbol{R}_i = \boldsymbol{R}_{i-1}\boldsymbol{R}_i^{i-1} \tag{2.37}$$

With the explanations delivered earlier for the derivation of the rotation matrix, the time derivative of (2.37) can be expressed as specified in (2.38).

$$\dot{\boldsymbol{R}}_i = \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}_i = \dot{\boldsymbol{R}}_{i-1}\boldsymbol{R}_i^{i-1} + \boldsymbol{R}_{i-1}\dot{\boldsymbol{R}}_i^{i-1}$$

$$= \boldsymbol{S}(\boldsymbol{\omega}_{i-1})\boldsymbol{R}_{i-1}\boldsymbol{R}_i^{i-1} + \boldsymbol{R}_{i-1}\boldsymbol{S}\left(\boldsymbol{\omega}_{i-1,i}^{i-1}\right)\boldsymbol{R}_i^{i-1} \tag{2.38}$$

The rotation matrices on the left side of equation (2.38) can be summarized and the right side can be rewritten by expanding with $\boldsymbol{I} = \boldsymbol{R}_{i-1}^T\boldsymbol{R}_{i-1}$ and by use of the relationship from equation (2.34) and yields in equation (2.39). The reformulation of the angular link velocity in (2.39) shows that the angular velocity of link $\boldsymbol{i}$ can be described as a function of the angular velocity of link $\boldsymbol{i-1}$ plus the angular velocity of link $\boldsymbol{i}$ in relation to link $\boldsymbol{i-1}$ (2.40). [SVS09]

$$\boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}_i = \boldsymbol{S}(\boldsymbol{\omega}_{i-1})\boldsymbol{R}_i + \boldsymbol{R}_{i-1}\boldsymbol{S}\left(\boldsymbol{\omega}_{i-1,i}^{i-1}\right)\boldsymbol{R}_{i-1}^{T}\boldsymbol{R}_{i-1}\boldsymbol{R}_i^{i-1}$$

$$= \boldsymbol{S}(\boldsymbol{\omega}_{i-1})\boldsymbol{R}_i + \boldsymbol{S}\left(\boldsymbol{R}_{i-1}\boldsymbol{\omega}_{i-1,i}^{i-1}\right)\boldsymbol{R}_i \tag{2.39}$$

$$= \boldsymbol{S}(\boldsymbol{\omega}_{i-1})\boldsymbol{R}_i + \boldsymbol{S}\left(\boldsymbol{\omega}_{i-1,i}\right)\boldsymbol{R}_i$$

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \boldsymbol{\omega}_{i-1,i} \tag{2.40}$$

Summarizing the previous content, the link velocity consists of a linear and an angular component. The linear velocity component is dependent on the translational and rotational velocity of the previous link $i-1$ and the relative translational velocity between link i and link $i-1$ (2.36). Its angular velocity, on the other hand, is determined by the angular velocity of the previous link $i-1$ and the relative angular velocity between link i and link $i-1$ (2.40). Dependent on the kinematic relationship of the links to each other, which is determined by their connecting joint, the expressions of the relative velocity components vary, as shown in Table 1 and briefly explained in the following.

Prismatic joint velocities

For prismatic joints there is no relative angular velocity between the links connected by the joint, while the relative translational velocity component is dependent on the joint axis $\boldsymbol{a}_{i-1}$ and the velocity of its joint variable $\dot{d}$. If these relationships are taken into account in the equations (2.36) for the translational and (2.40) for the angular component, the expression for the link velocities are given in (2.41) and (2.42), respectively.

$$\dot{\boldsymbol{p}}_i = \dot{\boldsymbol{p}}_{i-1} + \dot{d}_i\boldsymbol{a}_{i-1} + \boldsymbol{\omega}_i \times \boldsymbol{r}_{i-i,i} \tag{2.41}$$

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} \tag{2.42}$$

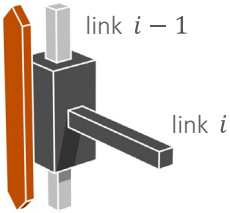
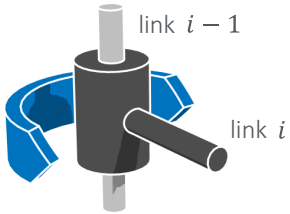
Revolute joint velocities

For revolute joints, the angular velocity results from the motion of the joint $\dot{\boldsymbol{\varphi}}_i$ and the joint axis $\boldsymbol{a}_{i-1}$, which in turn also influences the translational speed as given in Table 1. If this relationship is inserted into the equations for the linear (2.36) and rotational (2.40) velocity, equations (2.43) and (2.44), respectively, are derived.

$$\dot{\boldsymbol{p}}_i = \dot{\boldsymbol{p}}_{i-1} + \boldsymbol{\omega}_i \times \boldsymbol{r}_{i-i,i} \tag{2.43}$$

$$\boldsymbol{\omega}_i = \boldsymbol{\omega}_{i-1} + \dot{\boldsymbol{\varphi}}_i \boldsymbol{a}_{i-1} \tag{2.44}$$

Table 1 Velocity components with respect to joint type

prismatic joint P	revolute joint R
	
joint variables q_i	
$q_i = d_i$	$q_i = \varphi_i$
relative angular velocity between link i and link $i-1$	
$\boldsymbol{\omega}_{i-1,i} = 0$	$\boldsymbol{\omega}_{i-1,i} = \dot{\varphi}_i \boldsymbol{a}_{i-1}$
relative linear velocity between link i and link $i-1$	
$\boldsymbol{v}_{i-1,i} = \dot{d}_i \boldsymbol{a}_{i-1}$	$\boldsymbol{v}_{i-1,i} = \boldsymbol{\omega}_{i-1,i} \times \boldsymbol{r}_{i-1,i}$

Summarizing the previous contents, based on the time derivative of the description of the pose of link $\boldsymbol{i}$, its velocity is a function of the velocities of the previous link $\boldsymbol{i-1}$ and the motion of their connecting joint. These

correlations help to determine the kinematic dependent Jacobian, which is described in the following.

Geometric Jacobian $\boldsymbol{J}$

To determine the Jacobian, the linear component $\boldsymbol{J}_{\mathrm{p}}$ and the angular component $\boldsymbol{J}_{\mathrm{o}}$ are calculated separately as described in the following. The linear component consists of the linear end-effector velocities, which are generated by the respective joint $i = 1, \ldots, n$ when the remaining joints are stationary. The referring equation (2.45) shows the time derivative of $\boldsymbol{p}_{\mathrm{e}}(\boldsymbol{q})$. [SVS09]

$$\dot{\boldsymbol{p}}_{\mathrm{e}} = \sum_{i=1}^{n} \frac{\partial \boldsymbol{p}_{\mathrm{e}}}{\partial q_i} \dot{q}_i = \sum_{i=1}^{n} \boldsymbol{J}_{\mathrm{p}i} \dot{q}_i \tag{2.45}$$

With equation (2.45) and the joint variables and relative linear velocity as given in Table 1, the contribution of a prismatic joint P to the linear Jacobian $\boldsymbol{J}_{\mathrm{p}i}$ can be derived as given in (2.46).

$$\dot{q}_i \, \boldsymbol{J}_{\mathrm{p}i} = \dot{d}_i \boldsymbol{a}_{i-1} \overset{\dot{q}_i = \dot{d}_i}{\Longrightarrow} \boldsymbol{J}_{\mathrm{p}i} = \boldsymbol{a}_{i-1} \tag{2.46}$$

For revolute joints, the same applies with the information from Table 1 and equation (2.45) for the linear part of $\boldsymbol{J}$ and yields to (2.47).

$$\begin{aligned} \dot{q}_i \, \boldsymbol{J}_{\mathrm{p}i} &= \omega_{i-1,i} \times \boldsymbol{r}_{i-1,\mathrm{e}} = \dot{\varphi}_i \boldsymbol{a}_{i-1} \times (\boldsymbol{p}_{\mathrm{e}} - \boldsymbol{p}_{i-1}) \\ &\overset{\dot{q}_i = \dot{d}_i}{\Longrightarrow} \boldsymbol{J}_{\mathrm{p}i} = \boldsymbol{a}_{i-1} \times (\boldsymbol{p}_{\mathrm{e}} - \boldsymbol{p}_{i-1}) \end{aligned} \tag{2.47}$$

Thus, dependent on the joint type, the translational part $\boldsymbol{J}_{\mathrm{p}i}$ can be determined, and the angular part $\boldsymbol{J}_{\mathrm{o}i}$, generated by the prismatic joint or revolute joint, will be elaborated in the following.

With equation (2.40) the angular velocity of the end-effector can be calculated as given in (2.48).

$$\boldsymbol{\omega}_{\mathrm{e}} = \boldsymbol{\omega}_n = \sum_{i=1}^{n} \boldsymbol{\omega}_{i-1,i} = \sum_{i=1}^{n} \boldsymbol{J}_{\mathrm{o}i}\dot{q}_i \tag{2.48}$$

Using the relationship for angular velocities from Table 1, equation (2.48) yields the expression (2.49) for a prismatic joint and equation (2.50) for a revolute joint, respectively.

$$\dot{q}_i \boldsymbol{J}_{\mathrm{o}i} = \boldsymbol{0} \implies \boldsymbol{J}_{\mathrm{o}i} = \boldsymbol{0} \tag{2.49}$$

$$\dot{q}_i \boldsymbol{J}_{\mathrm{o}i} = \dot{\varphi}_i \boldsymbol{a}_{i-1} \implies \boldsymbol{J}_{\mathrm{o}i} = \boldsymbol{a}_{i-1} \tag{2.50}$$

Based on the last explanations, the Jacobian equation (2.51), which combines the linear and angular parts, is obtained with the entries of $\boldsymbol{J}_{\mathrm{p}i}$ and $\boldsymbol{J}_{\mathrm{o}i}$ dependent on the joint type as given in (2.52).

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{J}_{\mathrm{p}1} & \cdots & \boldsymbol{J}_{\mathrm{p}n} \\ \boldsymbol{J}_{\mathrm{o}1} & & \boldsymbol{J}_{\mathrm{o}n} \end{bmatrix} \tag{2.51}$$

$$\begin{bmatrix} \boldsymbol{J}_{\mathrm{p}i} \\ \boldsymbol{J}_{\mathrm{o}i} \end{bmatrix} = \begin{cases} \begin{bmatrix} \boldsymbol{a}_{i-1} \\ \boldsymbol{0} \end{bmatrix} & for\ P\ joint \\ \begin{bmatrix} \boldsymbol{a}_{i-1} \times (\boldsymbol{p}_{\mathrm{e}} - \boldsymbol{p}_{i-1}) \\ \boldsymbol{a}_{i-1} \end{bmatrix} & for\ R\ joint \end{cases} \tag{2.52}$$

In this way, based on the relationships of the direct kinematics, the Jacobian $\boldsymbol{J}$ can be determined for arbitrary points along the robot structure. The Jacobian is expressed with respect to the base frame that is assumed as fixed, but also can be expressed with respect to other frames if needed. Further details can be found in the literature [SVS09].

Differential kinematics equation

The Jacobian describes in the first order differential kinematics the mapping from the joint velocity vector to unique cartesian velocity vector of the end-effector $\boldsymbol{v}_{\mathrm{e}}$, consisting of a 3×1 linear, translational part $\dot{\boldsymbol{p}}_{\mathrm{e}}$ and a 3×1 angular end-effector velocity part $\boldsymbol{\omega}_{\mathrm{e}}$, describing the spatial motion of the end-effector. [SVS09]
The linear and angular part of the end-effector velocities can be derived using $\boldsymbol{J}_{\mathrm{p}}$ and $\boldsymbol{J}_{\mathrm{o}}$ with both dimensions of $3 \times \boldsymbol{n}$ as given in equations (2.53) and (2.54).

$$\dot{\boldsymbol{p}}_{\mathrm{e}} = \begin{bmatrix} \dot{p}_{\mathrm{x}} \\ \dot{p}_{\mathrm{y}} \\ \dot{p}_{\mathrm{z}} \end{bmatrix} = \boldsymbol{J}_{\mathrm{p}}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{2.53}$$

$$\boldsymbol{\omega}_{\mathrm{e}} = \begin{bmatrix} \omega_{\mathrm{x}} \\ \omega_{\mathrm{y}} \\ \omega_{\mathrm{z}} \end{bmatrix} = \boldsymbol{J}_{\mathrm{o}}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{2.54}$$

In combination, these components describe the end-effector's velocity as the robot's differential kinematics equation (2.55) by using the geometric Jacobian $\boldsymbol{J}$ as a function of the joint parameters as given in (2.52). [SVS09]

$$\boldsymbol{v}_{\mathrm{e}} = \begin{bmatrix} \dot{\boldsymbol{p}}_{\mathrm{e}} \\ \boldsymbol{\omega}_{\mathrm{e}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}_{\mathrm{p}}(\boldsymbol{q}) \\ \boldsymbol{J}_{\mathrm{o}}(\boldsymbol{q}) \end{bmatrix} \dot{\boldsymbol{q}} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{2.55}$$

Based on the relationships presented in this chapter, the Jacobian can be determined for any serial robot kinematics and, subsequently, the inverse kinematics can be determined using various numerical methods, as described in [AL09] in detail. For the dimensional synthesis of the tool developed, which is presented in Chapter 6.2, the saturation in the null space (SNS) algorithm from [FLK12] is implemented for this purpose and can be found in the literature. Having described the basic kinematic relationships, the following chapter deals with the method developed for task-based kinematics design.

3 Method for task-based synthesis of robot kinematics

This chapter describes the methodology developed for the task-based development of serial robots. First, an overview of the state of the art is given and the possibilities and approaches that already exist in these areas are shown for the three main components of the methodology, which are the pre-processor (Chapter 5), the solver (Chapter 6) and the post-processor (Chapter 7). In addition, the disadvantages of the state of the art are addressed shortly. Then, the solution structure is explained and the developed method as well as the interrelationships of the corresponding components are discussed more detailed in Chapter 3.2.

3.1 State of the art and its drawbacks

As briefly mentioned in Chapter 1.1, the development of new robotic systems can be divided into different work packages. These include the definition of the task, the type synthesis with subsequent dimensional synthesis and, based on this, the final design. The fields of type and dimensional synthesis have been researched extensively for several decades and a variety of different approaches was developed. There are methods for mapping tasks to suitable robot structures, analytical approaches, numerical methods, and optimization approaches. In the following, however, those works will be discussed that take a more holistic view of the subject and focus on task-based synthesis.

To develop a suitable system for a given task respecting further constraints like environmental conditions, the requirements of the application must be defined, and the motion task must be planned. Many approaches that deal with task-based kinematic design describe the task mathematically via goal-frames in cartesian space by defining the origin of the coordinate frame and the frame's orientation with respect to the world frame. Thereby, complete poses are defined, consisting of the position and orientation specifications and they will be consulted for the continuing synthesis process. Table 2 and Table 3 provide a selection and short overview of literature dealing with the task-based synthesis of serial kinematics.

Table 2 Selection of task-based design approaches from the state of the art - I

Reference	Task description and constraints	Design parameter	Optimization parameter
[CB95]	Set of points or frames	Robot assembly configuration	Performance measures based on the Jacobian
[CR96]	Trajectory within a high constrained environment, set of kinematics	Selection of one of the predefined kinematics and base position	Trajectory accessibility within environment
[CB97]	Set of frames and obstacles within the environment	Robot structure and link lengths	Reachability, deviations from the target pose, obstacle proximity, link length, number of joints, dexterity
[LB99; Leg99]	Trajectory and obstacles within the environment	Base position, robot structure, link dimensions	Deviations from the target pose, deflection of links, actuator saturation, mass, efficiency
[MA00; McC00; PSM04]	Limited number of frames along a trajectory	Robot structure and link length	Deviations from the target pose
[SKK02]	Positions, robot structure and joint limits	Link parameters and cross-sectional characteristics	Torque, deflection, efficiency
[KA06]	Robot structure	Link lengths	Condition number of the dimensionally homogenous Jacobian, workspace
[ODM09]	Robot structure, constraints for points within workspace	Variation of kinematic structure, link lengths	Reach workspace points respecting constraints

Table 3 Selection of task-based design approaches from the state of the art - II

Reference	Task description and constraints	Design parameter	Optimization parameter
[CCP10]	Robot structure	Link lengths, cross section areas	Workspace, mass, stiffness, safety
[AJM13]	Robot structure	Link length, trajectory, mass	Task time, link length, torque
[PS14; PS15a]	Positions, frames, orientations along task trajectory	DH parameters	Reach targets while avoiding singularities
[RKO14]	Motion vector	Robot structure, joint orientation, and link length	Fulfilment of the desired motion task
[MSV16]	Not explained	Modular robot structure, placement of task within workspace	Simplicity of structure, analytical solvability of IK, assembly index matrix
[IA16]	Carry payload from initial to goal position respecting obstacles	Modular robot structure and link dimensions	Path execution time
[SO19]	Trajectory	Joint displacement parameters	Deviations from target points along the trajectory, energy
[HWP20]	Sampling points of joint angles	Structural dimensions, parameterized joint components	Natural frequency, Cartesian stiffness, mass
[HKP21]	Set of frames or positions	DH parameters	Manipulability, total link length
[PBH21]	Trajectory	Type of links, link dimensions and curvature	Deviations from target pose respecting collisions, torque

In addition, it is stated in Table 2 and Table 3 how the task is described, which design parameters are used and for which parameters and criteria the optimization is performed. The result is either the structure and the optimal dimensions regarding the selected criteria or only the link lengths if the robot structure must be specified beforehand.

Based on the input data and boundary conditions, the solution field is restricted, which on the one hand can limit the complexity and shorten the calculation time, on the other hand there is the danger of unnecessarily restricting the solution field if more specifications are made than necessary. A robot structure, for instance, should only be predetermined if this kinematic scheme must be used and not more or less degrees of freedom or a different joint arrangement is more appropriate under the evaluation criteria considered.

Further, if a complete frame resulting in six constraints is specified to describe a task as it is done in many of the approaches listed in Table 2 and Table 3, it soon becomes obvious that the robot system to be developed must have at least six degrees of freedom to be able to achieve the specifications of all poses mathematically, since otherwise the system would be over constrained. In general, there is no specific number of parameters needed to completely define an arbitrary task since the description and constraints required are highly dependent on the task itself. Thus, not always all six degrees of freedom are needed, and the task can better be defined by a smaller number of parameters instead of the frame leading to six constraints. The space in which the robot's task is defined is called operational space and the dimension is dependent on the task description. In case that the dimension of the operational space is smaller than the dimension of the joint space, the robot kinematic is task redundant or also called functionally redundant, which means that it has more degrees of freedom than required to fulfill the task. Redundancy can be used to fulfill side conditions like collision avoidance or optimization with respect to the energy consumption or regarding other optimization criteria. Summarized, since not all tasks require all six degrees of freedom in space to be executed, it makes sense to define only the necessary constraints to have more flexibility and optimization possibilities in the synthesis process and to avoid unnecessarily

limiting the solution field or generating more complex kinematics than necessary.

Criteria according to which the structure and dimension synthesis can be carried out are, for instance: Minimum link lengths to save weight and costs, permissible deformation of the structure, effective task execution by the kinematics, avoidance of singularities and further manipulability or performance measures. A comprehensive literature survey for manipulator performance measures is given in [PS15b]. A suitable system can therefore be generated by means of multi-objective optimization. The problems can be complex and the solution procedures non-trivial. Deterministic methods such as the hill climbing method or heuristic approaches can be used to solve the problem. Another approach involves using stochastic methods such as genetic algorithms or particle swarm optimization to find a suitable solution. Depending on the formulation of the objective function, these global procedures can be well suited for multi-objective optimization under constraints. However, it should be selected according to the particular use case and requirements. Despite all the methods that are available, it is still not directly intuitive how the developer has to describe the task in a meaningful way, which boundary conditions must be defined and how the results are evaluated. It therefore makes sense to support the engineer with useful tools that can be integrated into the computer-aided design process, but only [Leg99] and [PSM04] from Table 2 offer a graphical user interface. According to the state of the art, the methods and synthesis procedures are versatile and well researched, but an intuitive accessibility of the contents is still limited and requires expertise and time investment. Table 4 provides an overview of available tools with a graphical user interface that support the developer in the synthesis and analysis of serial robot structures consisting of revolute and prismatic joints. The robot can be parameterized which is mostly done via the definition of Denavit-Hartenberg (DH) parameters and thus the structure can be built with the desired dimensions and subsequently animated and simulated to validate if the kinematic is suitable for the desired task. The tools given in Table 4 offer a good approach for the synthesis and evaluation of kinematics, but there is still potential for optimization. It is desirable for the development of new robot systems to be able to methodically integrate synthesis and analysis into the development process

and to provide suitable interfaces. Furthermore, only [LB99; Leg99] and [PSM04] automatically generate suitable robot structures based on the task description. With the other user interfaces, a system can be modelled and analyzed, but this must be built manually by the developer and it is therefore not guaranteed that the system can fulfil the task in a meaningful way automatically, and this must still be evaluated.

Table 4 Software with user interface for robot modelling, analysis, and simulation

Reference	Functionalities
Darwin2K [LB99; Leg99]	• Robot configuration synthesis based on task specifications • Simulation • Optimization of kinematics, dynamics, structural geometry, actuators
ROBOT BUILDER [Ruo16]	• Test robot kinematics while modelling • Models can be generated and stored to a library • Support for robot dimensioning and selection
SYNTHETICA 2.0 [PSM04]	• Synthesis of constrained serial chains • Interactive task definition and topology selection • Generation of intermediate frames for interpolated trajectory • Animation of generated solutions performing the task
ARAT [OKB19]	• Toolbox for kinematic and dynamic analysis and modeling of redundant serial robots • Stationary or mobile base possible • Simulation of complex structures defined via DH parameter
RoboAnalyzer [Sah20]	• Learn and teach robotic physics • Video lectures • Model, analyze and simulate serial robots defined via DH parameter

A partial automation of the sub-areas of kinematics development offers great potential to shorten development times and costs and is only represented to a limited extent in the state of the art presented here, so that it can be beneficial for an efficient development process to take a closer look at this topic. Thus, in the following Chapter 3.2, the developed methodology for semi-automated task-based robot development is presented in more detail.

3.2 Solution structure for task-based robot design

The development of robotic systems requires in-depth expertise in areas such as mechanics, electrical engineering, and computer science, making it a complex and interdisciplinary challenge. To increase the efficiency of the development process, it can be helpful to automate individual process steps or parts of them, so that a semi-automated overall process is provided, which requires user input at appropriate points.

The method and tool for task-based robot design, which is presented in the following chapters, helps the engineer to develop new robotic systems efficiently and to quickly identify and work out suitable solutions for specific tasks with manageable effort.

As mentioned in Chapter 1.1, this is particularly useful in the field of service robotics since there is a need for new kinematics and different requirements compared to industrial robots are given, so the referring boundary conditions should be considered. To be able to integrate the procedure into existing development processes, it is useful that it is oriented on the company structure, internal company processes or general guidelines.

One such guideline making a standard development process available is [VDI2221-1; VDI2221-2] for the design of technical products and systems. An overview of the process steps and the outcome of these steps are shown in Table 5. These process steps can be gone through one by one and to meet the requirements and solve the given task, an iterative approach may be necessary.

Starting with clarifying and describing the problem or task, first requirements can be derived. The more precisely the task can be formulated and elaborated, the more detailed the requirements can be derived, which is essential for the development process. As a result of the first process step, a list of requirements can be generated. This list is a collection of requirements that accompanies the development process and must be updated as new insights are gained and should be revised throughout the development process so that the developed system fulfils the required task in the best possible way. [VDI2221-1]

To facilitate a better understanding, some requirements in the field of robot development are mentioned in Chapter 4.

Table 5 Activities and results of the design process following [VDI2221-1]

development and design process		activities	results
	1	clarification of problem or task	requirements
	2	determination of functions and their structures	function models
	3	search for solution principles and their structures	basic solution concepts
	4	assessment and selection of solution concepts	solution concepts
	5	subdivision into modules and interface definition	system architecture
	6	shaping of modules	partial designs
	7	integration of product as a whole	overall design
	8	elaboration of execution and usage requirements	product documentation

Based on the results of the first step of the procedure shown in Table 5, functions and their structures can be deduced. These functional structures reflect what the technical system should be able to do. In process step three, solution principles can be generated, and effective structures are identified. Solution concepts can be delivered by combination of effective solution principles. A common tool for combination and generation is the morphological box that is described in [VDI2222-2] more precisely. From many of the derived concepts, the best solutions must then be determined in step four. This can be done by evaluating the fulfilment of the requirements and weighting them on basis of additional evaluation criteria. Once the preferred solution structure has been identified, a corresponding system architecture can be determined by dividing it into appropriate modules and interfaces. On this basis, the first design of the individual modules can take place, resulting in the so-called rough design of the system in step six. After finalization of the design and joining the results together to

the overall system, a solution for the task that was given at the beginning of the process is generated. Depending on the partial success of the development process and the degree of fulfilment of the requirements, these steps can be carried out iteratively. [VDI2221-1]

In addition to the methodical procedure for the development of general technical products and systems, Table 6 shows a procedure for the development of mechanisms following [Cor13]. The depicted process summarizes the steps from Table 5 from the perspective of mechanism development. The results of the individual work packages are given on the left for the synthesis' perspective and on the right for the analysis' point of view. This process is a helpful guideline for the development of novel mechanisms in a structured way and can also be consulted for the development of robot kinematics.

Table 6 Procedure to develop novel mechanisms following [Cor13]

synthesis	process steps	analysis
task and boundary conditions	1 planning and task definition	market analysis guidelines, regulations
drives and control joint types, arrangement	2 structural synthesis	control parameters kinematics, dynamics
analytical solution or optimization task	3 dimensional synthesis	
load-oriented design finalization for production	4 elaboration and design	other demands costs
	realization	

The method for task-based robot design presented in this work is based on the processes mentioned above and is geared towards the development of serial robot structures, optimized for the required tasks and boundary conditions. Since the design procedure follows [VDI2221-1], it can be integrated in existing structures with minimal effort. The general process is followed but is specified for the task-based development of serial robots. Some process steps are automated or semi-automated and, on some stages, specific user input is required. The procedure as well as the reference to the

existing processes indicated by the numbers in the diamonds and circles are shown in Figure 3.1.

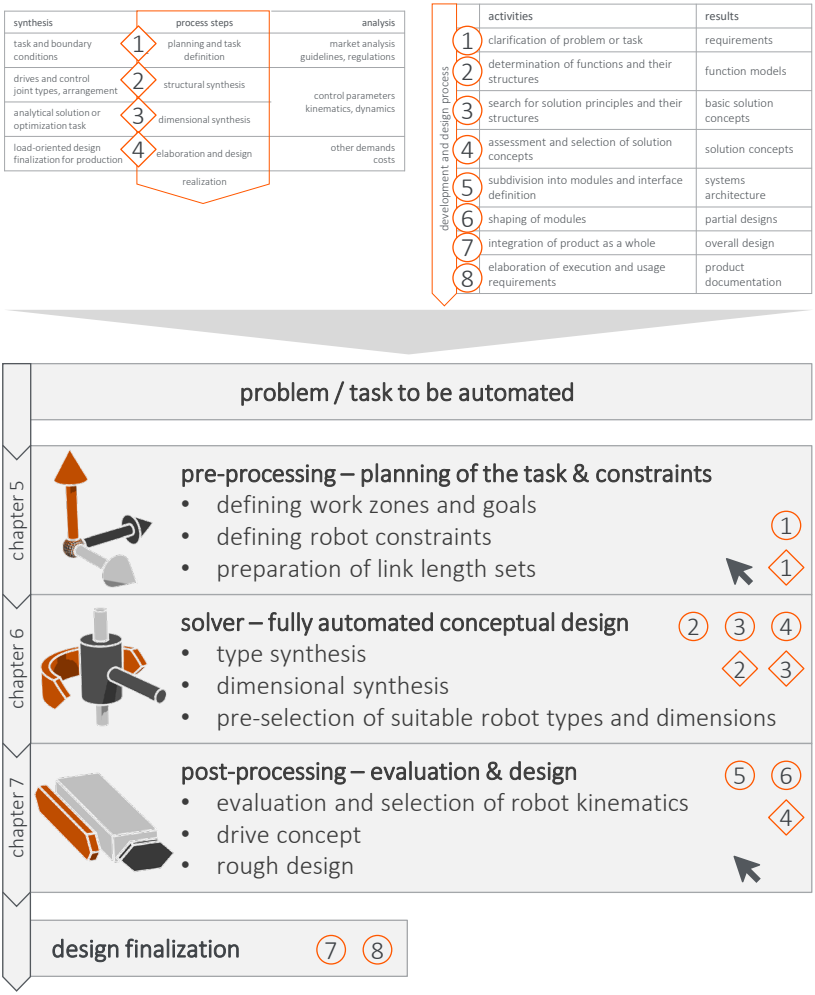


Figure 3.1 Method for task-based robot design

Starting from any given task that is to be automated by a robot, the three main steps pre-processing, solver, and post-processing can be followed, so that the rough design of the system is provided as a result. User interaction

is only required for pre-processing and post-processing, which is visualized by the mouse pointer in Figure 3.1. Based on this, the rough design, detailed design, and finalization of the development can be done.

Analogous to the first step of the processes in Table 5 and Table 6, within the pre-processor (Chapter 5) the given problem is planned and clarified. The problem here can be defined as a motion or handling task by the user, specifying what is to be reached by the robotic system. To define the task and environmental conditions as precisely as possible, additional constraints can be specified. Based on this user input, the necessary information for the solver is generated and prepared. Within the solver (Chapter 6), the type and dimensional synthesis are performed fully automated, which means that no intervention or editing by the user is required at this point. The output of the solver is then used as the basis for post-processing (Chapter 7), where the pre-sorted possible solutions can be evaluated and modified by the user, so that one solution, or more, if necessary, can be selected as the basis for further design. This process is carried out by the user and can, for example, be included in the procedure according to [VDI2221-1] and the development process can be completed. The results can then be used for further design. Depending on the intermediate results, the steps can be iterated if necessary, so that a suitable solution is achieved by adjusting the input data. The final design is not integrated into the methodology, but an overview of aspects that are essential for the design of service robotic systems is provided in Chapter 7.2.

The methodology presented in this work, and the tool developed are geared towards the design of suitable regional structures of serial robots and their rough design, which fulfil the required motion tasks in a targeted manner. Consequently, the focus is on a system consisting of the main axes of the kinematics and the robot's wrist development is not integrated in the methodological procedure. A detailed motivation for this approach as well as an overview of the corresponding advantages are given in Chapter 5.1.

In addition to the requirements of the motion task, there can be further wishes or requirements that should ideally be considered during the development process to help with the creation and selection of concepts. Since this should be taken into account from the very beginning of the development, a short overview is given in the following Chapter 4.

4 Requirements for conceptual and rough design

Before the individual process steps are discussed in the further sections of this work, this chapter first provides a brief overview of possible requirements for robotic systems. Most of the requirements for the system to be developed should already be known before pre-processing, so that the contents can be directly considered there. For this reason, this chapter gives an overview on how the requirements can influence each other and how they can be classified to ensure prioritization and better understanding. The overall goal in defining the requirements and taking them into account during the development process is to ensure that the developed system fulfills the task to be automated under all given constraints.

The requirements must cover both the external customer-oriented aspects such as the required performance specifications of the system, as well as the internal aspects of the company. Thus, they can be divided into techno-economic and organizational requirements. The technical and economic aspects include for instance technical requirements, interfaces, costs, and standards to be respected. Organizational requirements include for example time, personnel, and necessary tools. [FG13]

Since the tool developed in scope of this work is primarily intended to support the developer in conceptualization and rough design, the focus lies on the technical requirements. Nevertheless, the presented methodology and the developed tool have a great positive impact on the requirements especially for the resources, as development time and effort can be saved, and thus leading to a more efficient and cost reduced development process, from which both the internal organization and externally the customer will benefit.

The requirements that will be discussed in the following can be divided into functional and non-functional requirements. Functional requirements, as the name implies, refer to the functionality of the system, such as the payload or reach of a robotic arm and they have the highest priority, because only if they are fulfilled, the system will deliver the required performance. A non-functional requirement for instance is the color of the robot, which might be a requirement of the customer but has no influence on the functionality of the system but should also be respected.

From the customer's point of view, there are three different categories of requirements. Basic requirements like functionality of the system are mandatory and must be fulfilled but they do not generate enthusiasm and are taken for granted. Performance requirements such as payload and reach are defined by the customer and must be met by the developed system. Overachievement has a positive effect. Excitement requirements like unexpected features are not explicitly demanded or defined by the customer but can lead to great enthusiasm and have a positive effect, for example, through the resulting customer loyalty. The requirements dealt with in this work are primarily performance requirements since as described above, this work is about concept development and the rough robot design and the consideration of special features is not primarily addressed. Nevertheless, Chapter 10 briefly outlines how the methodology and tool can provide added value for the customer and the company.

To get a first definition of the systems' requirements and specifications, the characteristics presented in Chapter 2.1.2 can be used as a reference. These support the engineer to make sure that the developed robotic system fulfills the necessary properties. To maintain clarity and not to go into too much detail, six general higher-level requirements are identified that are particularly important for almost all applications for which the developed methodology can be used. Table 7 provides an overview of the selected higher-level requirements and corresponding examples of characteristics that can be quantitatively defined by the customer. In addition, it is indicated in which chapter these topics are further addressed.

Table 7 shows that the requirement of task fulfilment must be met in any case and ideally the degree of freedom of the system should for instance remain low, although the latter is not a direct requirement but a preference. To use the workspace efficiently, the design space of the robot should be kept small and, depending on the application, the requirements regarding safety can be specified and implementation can have a particular impact on the design of the system. Ideally, the complexity of the system is low and, depending on the boundary conditions, integrability is provided.

Even this small excerpt of possible requirements quickly shows that they often influence each other, which can lead to a conflict of objectives. For this reason, it is important to know the effects among each other, to prioritize the

requirements, and to find the best compromise for each application since there is not one universal optimum.

Table 7 Exemplary requirements for a robotic system

Functional requirements	Example	Chapter
Suitability for the given task (basic requirement)	• Reach all desired positions specifications • Necessary accuracy	5.1 8.3
Kinematic requirements (desirable requirement)	• Degree of freedom • Joint types and arrangement	5.2 8.3
Design space (desirable requirement)	• Collision contour of the robot arm • Flexibility of the system	7.2 8.6
Inherent safety (desirable requirement)	• Shaping of the robot arm • Lightweight design	7.2 8.6
Non-functional requirements		
Complexity	• Number of components • Simple mechanical design	7.2
Integrity	• Integration of stock components • Flange design	8.6

If the application is for instance in the field of service robotics and human-robot collaboration takes place, the primary requirements are the accomplishment of the desired motion task as well as the fulfillment of the necessary safety aspects. An exemplary extract of possible related requirements is given in Table 8 with a representation of the influences on each other given in the rows and a prioritization of the characteristics given by the numbers on the diagonal. The arrow in the first column indicates in which direction the parameter must be influenced to achieve a positive effect. For example, to get a positive impact on the accuracy, the accuracy must be increased, which is represented by an arrow pointing upwards. Furthermore, it is generally good if the system has the smallest possible collision contour, which is indicated by the arrow pointing down. In the other columns, the arrows show how the characteristics in the referring columns

change when the corresponding property in the row is improved, with a green arrow indicating a positive effect and a red arrow indicates a negative one.

Table 8 Selection of requirements and their influence on each other

Extract of possible requirements (priority indicated on diagonal)		Accuracy	Degree of freedom	Collision contour	Effective mass	Number of components
Accuracy	⇧	5	⇩	⇧	⇧	⇩
Degree of freedom	⇩	⇧	3	⇩	⇩	⇩
Collision contour	⇩	⇩	⇩	2	⇩	
Effective mass	⇩	⇩	⇩	⇩	1	
Number of components	⇩	⇧	⇩			4

If for instance the effective mass must be reduced to enable safe human-robot collaboration, this could be done by reducing the degrees of freedom, as shown in Table 8. However, this is often not possible since the degree of freedom cannot be reduced arbitrarily, as the basic requirement is that the robot must reach all specified positions while taking optional boundary conditions into account. Therefore, another way must be found to reduce the effective mass, for example by modifying the collision contour. Relocating the components of the robotic arm closer to its base might help to reduce the masses in movement. If as in this example the application for which the system is to be developed is in the field of service robotics, it is important that the required accuracies are maintained, but these are in a completely different range than in applications in the industrial sector, where a high accuracy is often crucial. This means that a compromise can be made in which the effective mass of the system is decreased while accepting a concomitant reduction in accuracy. However, if there was a high requirement for accuracy, this approach would obviously not be possible and other ways would have to

be evaluated to improve and achieve the desired criteria. An application example with such considerations is given in Chapter 8 based on the development of a robotic system within the scope of a research project that is dealing with robotics enabling fully integrated logistics lines for supermarkets (REFILLS).

To summarize this chapter, depending on the application under consideration and the resulting requirements and their prioritization, different solution concepts may be appropriate to reach a desired specification. Usually, this does not work without compromising on other criteria. In the end, a suitable trade-off must be found that meets all basic requirements and addresses as many desired criteria as possible. Once the main technical requirements have been defined, the concept development can be initiated.

The definition of first requirements regarding the task to be automated and additional mechanical constraints for the novel robotic system can be defined within the first step of the presented method, the so-called pre-processing. The necessary input data, processing within this step, as well as the results are discussed in the following Chapter 5.

5 Pre-processing – planning of the task and defining constraints

The development tool can be used as an assistance for the kinematics synthesis and rough mechanical design if a new robotic system is to be developed for a special motion or handling task. As described more detailed in this chapter, pre-processing is the first step of the presented process and within its scope, the task is defined, and additional boundary conditions are determined. Input, steps within the pre-processor and the output as well as the integration into the overall process are shown in Figure 5.1

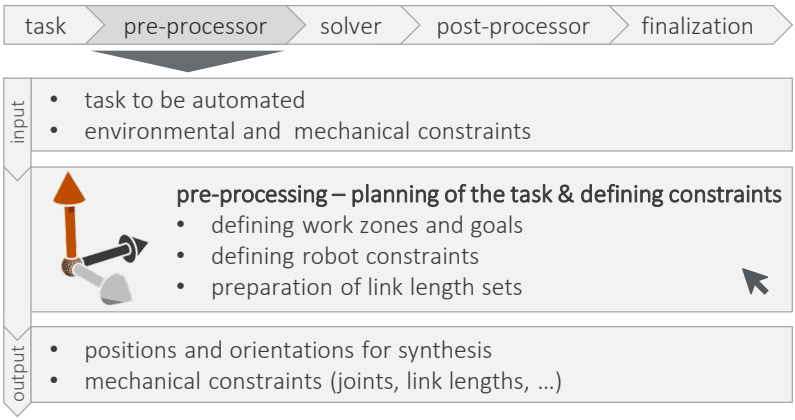


Figure 5.1 Pre-processing within the development process

As depicted in Figure 5.1, input for the pre-processor is the task to be realized as well as constraints related to the environment and kinematics. The task is described by specifying target positions and optional orientation conditions which is further described in Chapter 5.1. In terms of kinematics, the degree of freedom, permitted joint types and their axis orientations must be specified as explained in Chapter 5.2. In addition, the minimum and maximum link length are defined for each robot link as well as their step size for discretization. An overview of the properties to be specified by the user during the pre-processing is shown in Table 9 and Table 10. Output of the pre-processor are the kinematics to be considered, link length sets as well as the target positions and orientations, based on the user's input. After the

structure of the pre-processor was given, the next Chapter 5.1 addresses the task description.

5.1 Task definition and description

This chapter describes the approach used for task description and explains how it is implemented within the framework of the design tool, using an example for illustration.

As described in Chapter 2.1, the kinematic structure of a serial robot is composed of the regional and local structure. While the regional structure is primarily responsible for the positioning of the end-effector, the local structure determines its orientation. In summary, the pose of the end-effector is a combination of the position, determined by the main axes, and the orientation, specified by the secondary axes of the robot.

Since this work is about the rough design of new robotic systems and the first step is mainly to ensure that the target positions can be reached without collisions, the design task is broken down to the development of the regional structures of the serial kinematics. If the end-effector is also to be oriented in a certain way, this additional mobility can be provided by adding a wrist to the kinematic structure and the referring processing steps could be integrated into the tool in the future but are not considered at this stage.

According to the overview of state of the art in Chapter 3.1, in most cases of task-based robot design complete frames consisting of their position and their orientation are given as input data for the kinematic synthesis. This leads to the situation that more conditions are given than are relevant in the actual application and therefore any task redundancies cannot be exploited, or the system might be over dimensioned. If it is not yet precisely defined in which orientations the end-effector must reach the target positions, it is not advisable to specify randomly selected orientations, as this would unnecessarily restrict the solution field and thus theoretically suitable kinematics cannot be considered. In conclusion, a significant advantage of only considering the regional structure is that the motion task is only described as precisely as necessary, but it can be ensured that the robot reaches all desired positions without over constraining the task. Therefore, only the regional structure of the robot that positions the end-effector in

space is considered within the methodology presented and the approach to describe the task is explained in more detail below.
The developed tool is implemented in MATLAB and besides a graphical user interface, which will be explained later in Chapter 7.1, the main program and two classes are the core elements of the tool, one being the class *c_robotlist* which is created within the pre-processor. Some of its properties referring the robot's task are shown in Table 9 and they must be defined to prepare all necessary data for the solver. To describe the task at position level, the robot base position and the desired target positions must be specified. This is done by the user and can be based either interactively on computer-aided design (CAD) data or by specifying absolute coordinates in the fixed world frame.

Table 9 Properties to define the task

Property	Symbol	Description
Base position	$\boldsymbol{p}_{\mathrm{b}}$	Robot's base position
Positions	$\boldsymbol{P}$	Matrix for the desired goal positions
Goal type	$\boldsymbol{g}$	Vector for goal type definition for each position
Desired orientation	$\boldsymbol{R}_{\mathbf{des}}$	Matrix that describes the desired orientation

If the environment in which the robot is to be used is already known and a CAD model of the important components is available, it can be used within the pre-processor to describe the task as visualized in Figure 5.2. The user can provide the data from any CAD system via the standard stereolithography (STL) interface for further use in the development tool. It is important to note here that the file is processed in MATLAB with the unit meter, but the dimensions in most CAD systems are defined in millimeters, so that scaling with the factor 0.001 may be necessary.
Depending on the handling task to be automated, the CAD data may not be available. In this case, however, such a model can rapidly be generated for further processing using a LiDAR sensor, which nowadays is integrated in some smartphones or tablets on the market. Figure 5.3 shows an exemplary

scenario that can be digitized and converted into an STL file within minutes and can thus be easily utilized in the process presented in the following.

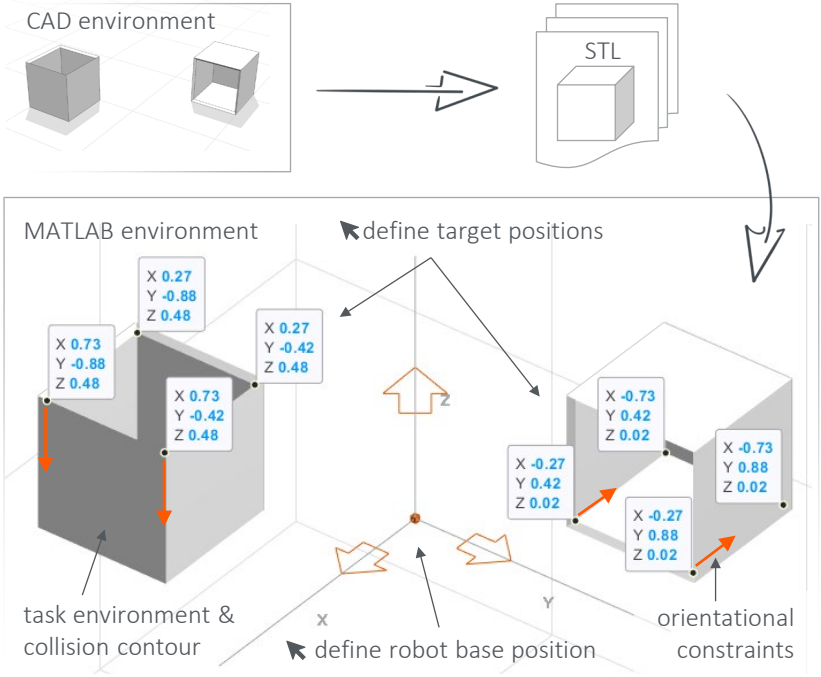


Figure 5.2 Data preparation and interaction

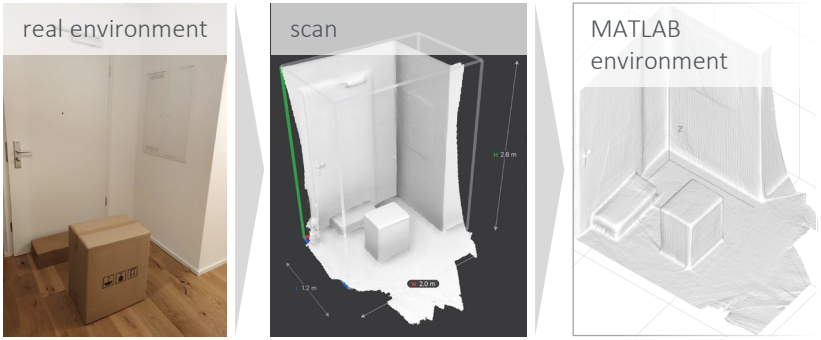
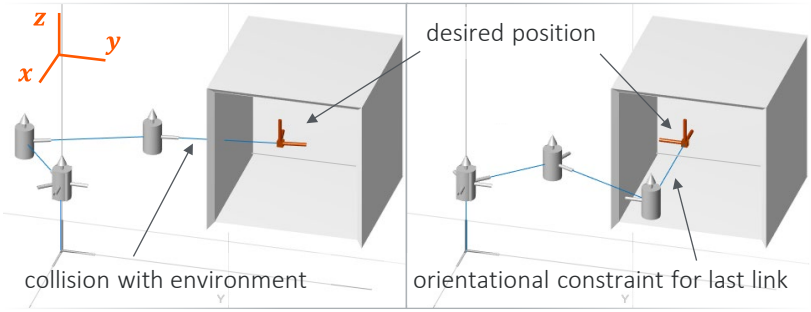


Figure 5.3 Possibilities to generate the environmental model

After importing the STL environmental model into the MATLAB environment, the robot base position can be defined by specifying the coordinates with respect to the world frame and results in $\boldsymbol{p}_{\mathrm{b}} = [p_{\mathrm{b,x}}, p_{\mathrm{b,y}}, p_{\mathrm{b,z}}]^T$, or by interactive positioning in relation to the STL model as shown in Figure 5.2, so that the data is stored in the base position vector automatically.
The desired target positions can be defined interactively by selecting them. If the points are specified by interactive selection, their coordinates are automatically saved as entries in the matrix $\boldsymbol{P}$. Alternatively, or in addition, points can be described by specifying their coordinates $\boldsymbol{p}_i = [p_{i,\mathrm{x}}, p_{i,\mathrm{y}}, p_{i,\mathrm{z}}]$. The number of rows of $\boldsymbol{P}$ then corresponds to the number of target positions and the number of columns equals three, for the x, y and z coordinates.
In addition to specifying the target positions, the task can be described even more precisely by considering information about the obstacle contours of the environment. If the robot moves in an environment to a desired target position or pose, it must be ensured that it does not collide with itself or with the environment. The collision check in general is done within the motion planning. Still, during the development of kinematics it should be considered, and it must be ensured that all specified positions can be reached without collisions with the environment, or at least the risk should be limited.
If no further boundary conditions except the target position are specified, the robot kinematic may approach a point with the end-effector in such a way that the robot collides with the environment as shown in Figure 5.4 (a).



(a) No boundary condition (b) With orientational constraint

Figure 5.4 Orientational boundary conditions for collision avoidance

To counteract this and to be able to reach the target position without collision, part of the orientation of the last robot link can be specified. In this way information about the environment is integrated directly into the task description. Figure 5.4 (b) shows how the robot kinematics can reach the target position without colliding with the interfering contour of the box. For this purpose, the boundary condition was defined that the x unit vector of the end-effector frame must point in the negative direction of the x unit vector of the world coordinate system. In this way it can be ensured that the x vector of the coordinate system of the last link of the regional structure moves parallel to the side and bottom of the box and is aligned normally to the box opening plane. If the robot moves within the box under these specifications, the risk of a collision is reduced. Nevertheless, only the fewest possible boundary conditions are given, so that otherwise existing task redundancies are not eliminated, which would have been the case if complete frames were specified by defining the six referring parameters. By the specifications as in this example, only the direction of the x unit vector of the end-effector coordinate system is given, but the frame itself can still be rotated around this same vector $\boldsymbol{x}_{\mathrm{e},1}$ but not around any other axes as shown in Figure 5.5.

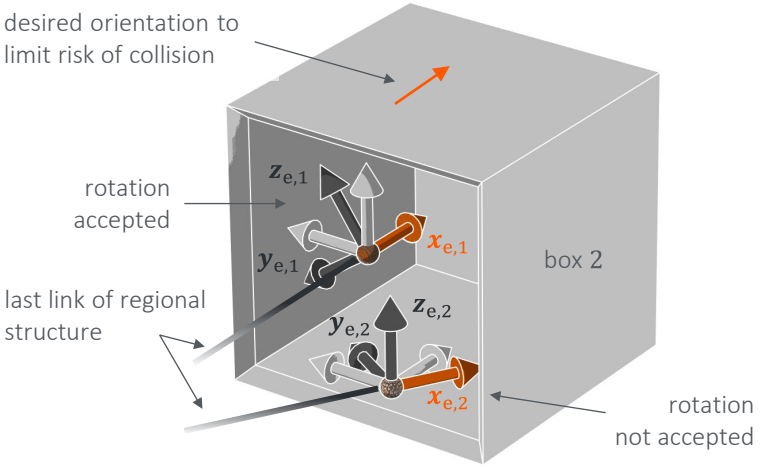


Figure 5.5 Permitted rotation in case of orientation specification

Rotations around other vectors of the end-effector frame are not allowed in this case. Depending on the application, the user can determine around which axis the end-effector should be able to be freely rotated. In this case and in many others, free rotation around the x axis of the end-effector and its orientation specification regarding the environment is useful. The information whether an orientation condition is given or not, and if so, with respect to which axis, is given in the vector $\boldsymbol{g}$. There are the options 'position', where no boundary condition exists for the orientation of the end-effector, and options to select the x, y, or z axis as constraint for a desired orientation with free rotation, which refers to rotation 'around x vector', 'around y vector', and 'around z vector' of the end-effector frame, respectively. It can be concluded that based on the environmental conditions, additional boundary conditions can be created regarding the orientation of the last robot link to avoid collisions. It is important that the user roughly considers what is relevant when reaching the default positions. Figure 5.6 shows a pick and place application in which the objects are to be moved from one box to another.

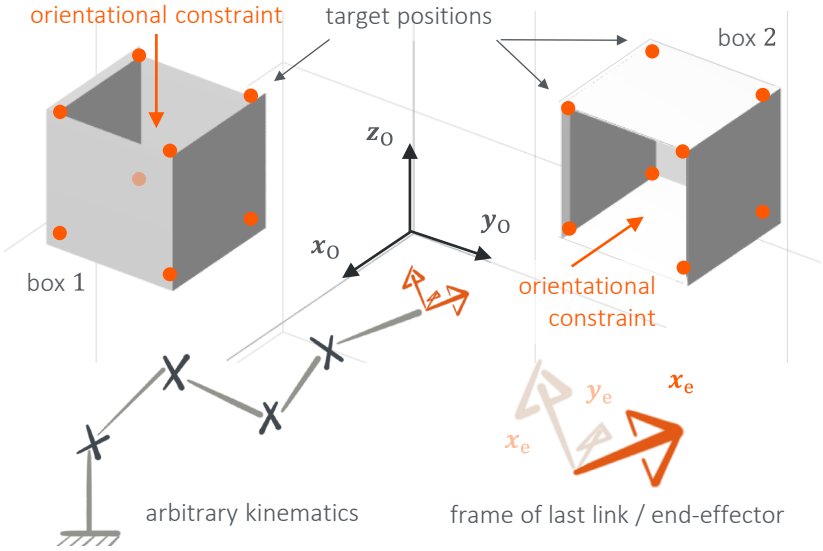


Figure 5.6 Exemplary task to show how to define orientational constraints

It is obvious that the items in the boxes can be reached in different ways. In box 1 the items are accessible from above and in box 2 they can be reached from the front. The orientation boundary conditions must be specified accordingly. In box 1 the robot kinematics to be generated can enter from above, which can be defined by specifying that the $\boldsymbol{x}$ unit vector of the end-effector must point in the negative z direction of the world coordinate system. Input data to define the desired orientation of the end-effector is given by $\boldsymbol{R}_{\mathrm{des},1,:} = \boldsymbol{x}_{\mathrm{e},1} = [0\ 0\ -1]$ with respect to the world coordinate system. The orientation specification for box 2 is identical to the example in Figure 5.4, namely that the $\boldsymbol{x}$ unit vector of the end-effector must point in the negative $\boldsymbol{x}$ direction of the world coordinate frame which results in $\boldsymbol{R}_{\mathrm{des},2,:} = \boldsymbol{x}_{\mathrm{e},2} = [-1\ 0\ 0]$. This is a kind of constant orientation workspace definition, but the constant orientation refers only to one axis of the end-effector frame. In the later course of the synthesis, it must be ensured that the constant orientation workspace defined by the task description is part of the constant orientation workspace of the generated kinematics.

In previous work as described in [ZHH19], the position definition and point generation is done via interactive specification based on the environmental data and is shown in Figure 5.7. Points can be selected and placed based on STL models in a MATLAB environment and individual work volumes can be defined. To ensure that the kinematics can reach several work zones, a transition volume is automatically generated. This volume is free of collisions with the environment and the predefined robot position is considered in the calculation, so that the transition is as direct as possible.

Since the desired positions with optional orientation specifications must be provided as input data for the solver, the information from the defined volumes must be processed further. For this purpose, points can be generated within these different volumes. This can be done automatically by point sampling, so that working points are generated in the working zones and transition points in the transition volumes. These points are then used as positions to be reached by the robot's regional structure. In addition, the STL objects are identified as interference contours and if required, obstacle points can be sampled within these volumes.

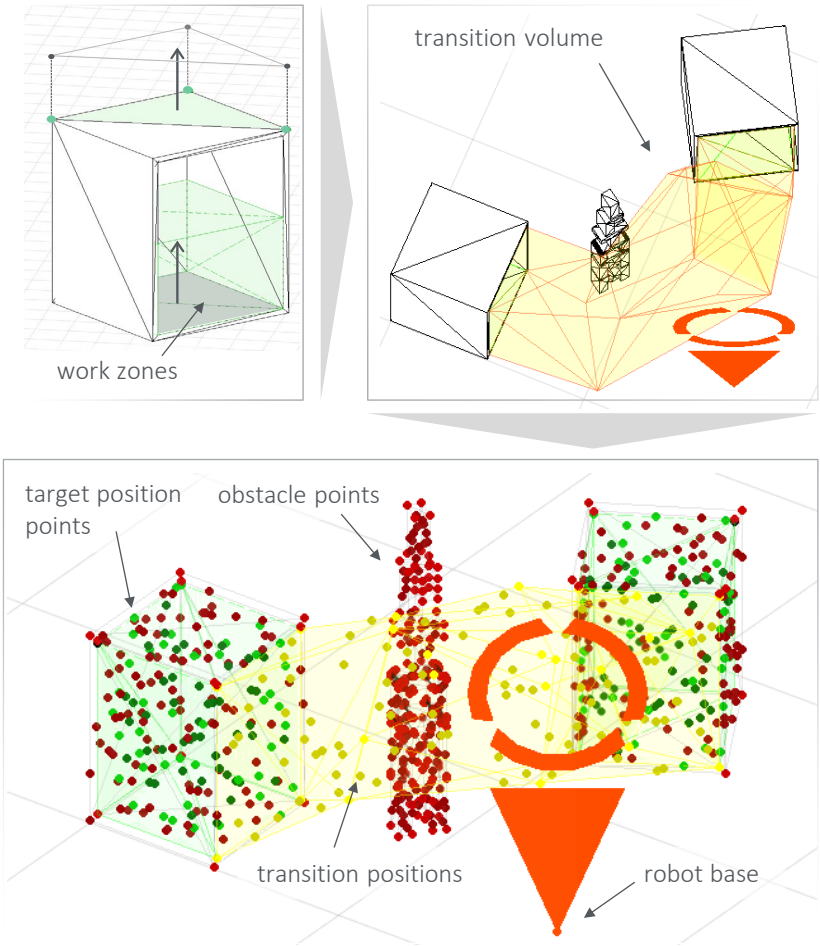


Figure 5.7 Pre-processing for task-based synthesis following [ZHH19]

The number of points to be sampled can be specified by the user. However, tests have shown that it is not necessary to also generate many points within the work zones and use them as input for the synthesis. If the work volume is spanned by suitably selected corner points, it is sufficient to use these points as input for the synthesis and the entire area can still be reached with suitable kinematics. Reducing the number of points decreases the computational effort and thus speeds up the process while retaining the

same functionality. By additionally specifying so-called guide points and a reasonably selected sequence of the target positions, the task can be described more precisely and thus improve the calculations within the solver, as described in Chapter 6. Therefore, the point sampling approach used in [ZHH19] was not developed further, but the focus was placed on smart selection of a minimum number of target positions, as even the reduced number of default positions leads to sufficient results.

Together with the specification of the optional orientation constraints, the task can be suitably described. Once all the specifications have been made, the user interaction is completed, and the data can be prepared for the solver automatically. The desired positions are stored in matrix $\boldsymbol{P}$. If additional boundary conditions regarding the orientation of the last link are needed, the vectors describing the desired orientations are stored in the matrix $\boldsymbol{R}_{\mathrm{des}}$ for the referring positions.

In addition to defining the motion task, mechanical boundary conditions are also specified during pre-processing as described in the following Chapter 5.2 more detailed.

5.2 Defining robot constraints

Within the pre-processor, the user manually specifies additional mechanical constraints referring the robot itself, based on the given requirements. An overview of the properties to be defined is shown in Table 10 and is discussed in the following, starting with the set-up of requirements regarding the kinematic scheme followed by the set-up of the link length combinations.

Table 10 Properties to define the robot kinematic

Property	Symbol	Description
Degree of freedom	dof	Defines the number of joints
Joint types	a_{type}	Preset prismatic, revolute or both joint types
Joint orientations	a_{ori}	Define approved joint axes
Minimum link lengths	l_{min}	Minimum link length for each link
Maximum link length	l_{max}	Maximum link length for each link
Step size	s	Step size for link length discretization
TCP offset	$\boldsymbol{d}_{\text{TCP}}$	Offset of the tool center point
Number of robot priorities	nr_{prio}	Number of desired solution selections

Set-up of requirements regarding the kinematic scheme

First, the minimum degree of freedom dof of the kinematic chain to be generated must be specified. In general, the minimum degree of freedom considered in this design process is three. This is the minimum dof required to reach arbitrary points in space with the robot's regional structure. If no kinematics can be determined based on the input within the solver, the process can iteratively be repeated with one additional degree of freedom until suitable solutions can be generated, or no solutions can be found even for the maximum permissible degree of freedom and the process is cancelled.

In addition to specifying the degree of freedom, the user also has the option of determining which joints and which axis orientations are to be considered.

This is especially useful when mechanical boundary conditions are already known regarding the robot axes. If not only revolute joints R but also prismatic joints P are to be considered within the presented tool, these can be defined as first joint of the regional structure, as last joint that belongs to the robot's tool, or both. Prismatic joints at other positions are not considered in the standard procedure of kinematic pre-selection within the presented design tool but can be integrated in future work. However, this would increase the calculation time and is often not desired for a robot structure due to the mechanical and cost-specific disadvantages of prismatic joints. So, these combinations are not considered in the default setting, but can be specified by the user if required.

The user can define for each axis which joints are to be considered. The information is stored in a vector $\boldsymbol{a}_{\mathrm{type},i}$ by defining 'revolute', 'prismatic' or 'both' for each joint $i = 1, \dots, dof$. The referring orientation for each of the joints is stored in vector $\boldsymbol{a}_{\mathrm{ori},i}$ by defining either x, y, and z as joint axis, or a combination of them, or no specifications at all, so that all the three orientations will be considered. The orientation of the joint axis then corresponds to the x, y or z unit vector direction of the world coordinate system. Joint orientations are only allowed in the x, y or z axis, so that there is always a multiple of $90°$ as an angle between the joint axes, to keep the combinatorics manageable and because these axis arrangements have proven to be practicable. In this work, the specification and description of the joints and their axes alignment is intentionally chosen and not, for example, based on the Denavit-Hartenberg notation, so that easier communication between the developers and other people is possible and a general understanding is sufficient to exchange information regarding the kinematics.

After the user's input regarding the mechanical constraints of the joints, all kinematics to be investigated for the given task can be generated automatically within the solver that is described in Chapter 6.1.

Set-up of link length combinations

In addition to the joint specifications, the permitted ranges of the individual link lengths of the robot must be defined. The setup of the link lengths combinations is a preparation for the dimensional synthesis for all considered

robot types processed within the solver in Chapter 6.2. Based on the input of the minimum allowed length for each link $l_{\mathrm{min},i}$, the maximum link length $l_{\mathrm{max},i}$ and the desired step size s_i for each link, the discretization of the robot links is calculated and stored in vector $\boldsymbol{l}_{\mathrm{pr},i}$ for $i = 1, \dots, dof$, following equation (5.1). The size of $\boldsymbol{l}_{\mathrm{pr},i}$ which is $(1 \times m_i)$ can be calculated using (5.2) and (5.3), with m_i representing the number of possible link lengths that are investigated for link i. If $m_i{}'$ is not a whole number, it is rounded down to the next integer. After preparation of the link lengths for each link, all possible combinations of those among each other are generated and stored in matrix $\boldsymbol{L}_{\mathrm{pr}} = [\boldsymbol{l}_{\mathrm{set},1'}, \dots, \boldsymbol{l}_{\mathrm{set},nr_{\mathrm{pr}}'}]^T$. The total amount $l_{\mathrm{nr,pr}}$ of the generated link length sets can be derived from (5.4).

$$\boldsymbol{l}_{\mathrm{pr},i} = [l_{\mathrm{min},i} \quad l_{\mathrm{min},i} + s_i \quad \dots \quad l_{\mathrm{min},i} + \mathrm{m}_i{}' \cdot \mathrm{s}_i] \tag{5.1}$$

$$m_i{}' = floor\left(\frac{l_{\mathrm{max},i} - l_{\mathrm{min},i}}{s_i}\right) \tag{5.2}$$

$$m_i = m_i' + 1 \tag{5.3}$$

$$l_{\mathrm{nr,pr}} = \prod_{i=1}^{dof} m_i \tag{5.4}$$

To reduce the computational effort within the solver, all link length combinations that obviously cannot reach the desired goal positions, not even in the stretching position of the robot, are sorted out as illustrated in Figure 5.8. For this pre-selection the sum of link lengths for each set $i = 1, \dots, l_{\mathrm{nr,pr}}$ of the generated link length combinations are compared with the maximum distance between robot base position $\boldsymbol{p}_{\mathrm{b}}$ and goal position $\boldsymbol{p}_g$ with $g = 1, \dots, n$ and only accepted as valid set, if the required inequality (5.5) is fulfilled. All valid link length sets are stored in matrix $\boldsymbol{L}$ as given in (5.6) with dimension $(l_{\mathrm{nr}} \times dof)$ and l_{nr} representing the number of considered sets that fulfill the given requirements based on the user's input.

$$\sum_{j=1}^{dof} \boldsymbol{l}_{\mathrm{set},i}(j) > max\left(\{\|\boldsymbol{p}_1 - \boldsymbol{p}_\mathrm{b}\|_2, \ldots, \|\boldsymbol{p}_n - \boldsymbol{p}_\mathrm{b}\|_2\}\right) \tag{5.5}$$

$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{l}_{\mathrm{set},1} \\ \vdots \\ \boldsymbol{l}_{\mathrm{set},l_\mathrm{nr}} \end{bmatrix} = \begin{bmatrix} l_{1,1} & \cdots & l_{1,dof} \\ \vdots & \vdots & \vdots \\ l_{l_\mathrm{nr},1} & \cdots & l_{l_\mathrm{nr},dof} \end{bmatrix} \tag{5.6}$$

Finally, the user can specify how many solutions per kinematics type are to be stored temporarily as priority solutions within the solver to be able to process them afterwards. This property is set as $nr_\mathrm{prio} = 5$ as the default setting if the user does not specify it individually.

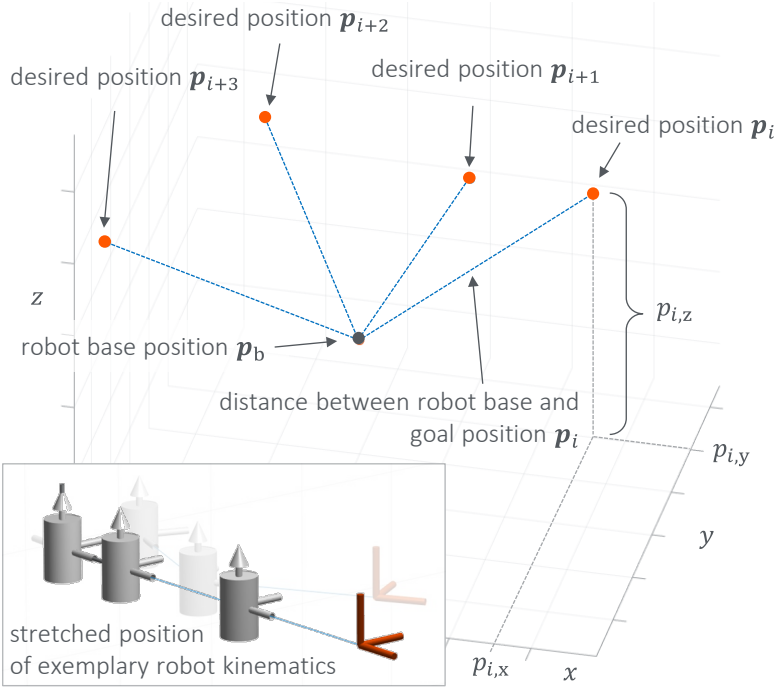
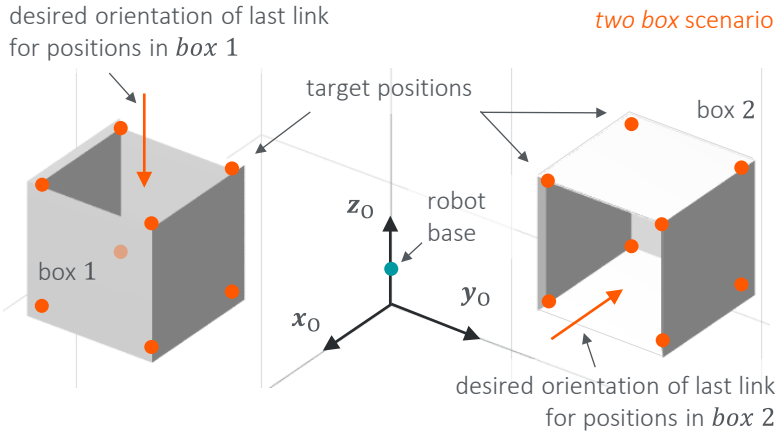


Figure 5.8 Preselection of suitable link length sets

Based on the procedure described in this chapter, the user input can be processed within the pre-processor so that the parameters are assigned

according to the robotlist and the appropriate link length sets to be considered are generated automatically.

To illustrate the presented work and to have a cross-chapter example, the *two box* task from Figure 5.6 will be incorporated in the following work and an overview of the output of the pre-processor with respect to that handling task is visualized in Figure 5.9.



property	value
position + orientation	see figure above
robot base	$(0\ 0\ 0.3)^T$
dof	5
joint constraints	consider only R joints first joint axis along z: $a_{\mathrm{ori},1} = z$
link length bounds	$l_{\min} = [0.1\ 0.1\ 0.1\ 0.1\ 0.4]$ $l_{max} = [0.5\ 0.5\ 0.5\ 0.5\ 0.8]$ $s = 0.2$ $l_{\mathrm{nr,pr}} = 243 > l_{\mathrm{nr}} = 237$

Figure 5.9 Properties resulting for the *two box* scenario after pre-processing

The target positions and orientation requirements are used as described earlier and as visualized in Figure 5.9. The coordinate origin of the world frame with $0.3\ m$ offset for the z axis is selected as the basis for the robot kinematics and the degree of freedom of the kinematic scheme to be considered is five. Structures with only revolute joints are to be investigated and the lower and upper bounds for the link lengths are given in Figure 5.9 and result in 237 automatically generated valid link length sets.

After the pre-processing is completed, the necessary input data is ready for the solver, in which the type and dimensional synthesis is fully automated as described in detail in the following Chapter 6.

6 Solver – fully automated conceptual design

The second process step within the presented robot design procedure is the conceptual design within the solver. An overview of input, steps within the procedure and the output as well as the integration into the overall process are shown in Figure 6.1.

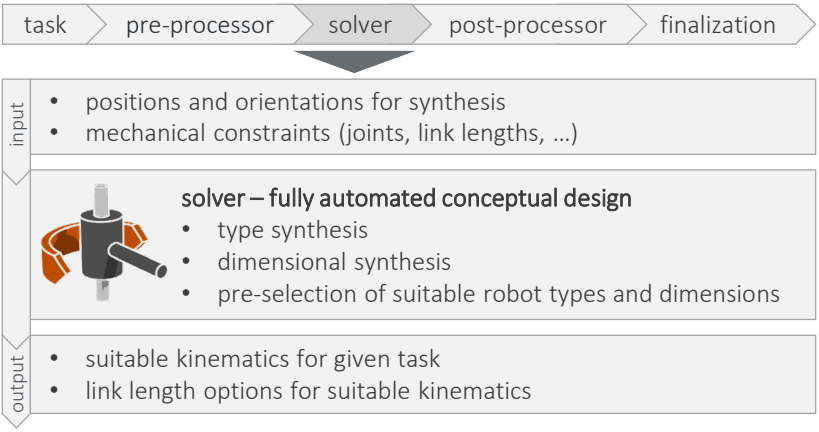


Figure 6.1 Solver within the development process

For this procedure, the output data from the pre-processor can be used as input data for the solver. Based on the specifications regarding the kinematics boundary conditions, all kinematics to be considered are generated within the type synthesis as described in Chapter 6.1. Then, for each of this kinematics the dimensional synthesis is processed, considering the link length sets and task description from the pre-processing, as described in Chapter 6.2. The output of the solver are suitable robot objects being defined by their kinematic scheme and corresponding link lengths, so that these solution drafts can be used as a basis for the further processing within the post-processor. All steps within the solver are fully automated, so that no user intervention is necessary.

After the chapter's contents have been discussed, the next Chapter 6.1 describes the type synthesis, in which the kinematic schemes to be considered are generated.

6.1 Type synthesis

Within the scope of type synthesis, kinematic schemes or also called topologies that are suitable for the given task are generated so that qualitative mechanical structures of the serial robot are obtained. All kinematic structures that are to be considered in further processing of the development tool are generated automatically within the type synthesis of the solver. This is done based on the user's input data that happened within the pre-processing as described in Chapter 5.2. In this chapter, the number of structures and the need to limit this as much as possible is discussed first, followed by an explanation of the procedure applied to generate the schemes and an overview of the resulting workspaces. The first part of the chapter deals with structures consisting of revolute joints only and subsequently discusses the integration of prismatic joints at the first or last position of the kinematic chain.

The number of possible kinematic schemes r_{nr} can be calculated by combining all possible joint types and all possible orientations of the referring axes. For the general case the number of possible architectures can be calculated using equation (6.1), with n_{joints} representing the number of different joint types that are considered for all $i = 1, \dots, dof$ joints, and n_{ori} describing the number of their possible axis orientations.

$$r_{\mathrm{nr}} = \prod_{i=1}^{dof} n_{\mathrm{joints},i}\, n_{\mathrm{ori},i} \tag{6.1}$$

As the number of possible combinations grows considerably by increasing the degree of freedom, number of different joint types and their possible axis orientations, it becomes evident that depending on the application it may be advisable to keep the number of combinations as low as possible if the computational effort must remain manageable. For this reason, the user can define existing boundary conditions or limitations during pre-processing so that undesired kinematics are no longer considered, which reduces the computation time. Within this work only revolute and prismatic joints are

considered and axis orientations in the direction of the orthogonal x, y and z axes of the world coordinate frame with respect to its home configuration which is when all joint parameters equal zero, so that $q_i = 0$ and the resulting crossing angles are either $0°$ or $90°$. Thus, expression (6.1) becomes equation (6.2) under the given limitations.

Considering three degrees of freedom of the kinematics, a total amount of 216 kinematic types can be generated based on these specifications. For $4\ dof$ there exist 1296 structures and for $5\ dof$ there are 7776 kinematic schemes following equation (6.2) and indicated in right column in Table 13. Since, despite the restriction to two joint types and three possible axis orientations, the number of possible combinations grows significantly as the degree of freedom increases, it is reasonable to take other boundary conditions into account during type synthesis. These additional specifications can be given by the user during pre-processing as described in Chapter 5.2 and their effect on the synthesis of possible kinematic structures is explained in the following.

$$r_{\mathrm{nr}} = 2^{dof} \cdot 3^{dof} \tag{6.2}$$

As described in Chapter 5.2, there may already be requirements for the joints of the robot kinematics. Thus, in some cases it makes sense to not include prismatic joints or at least to restrict them to certain positions. In addition, some axis arrangements do not make sense, such as two successive prismatic joints with the same axis orientation. Structures that do not span a three-dimensional workspace but can only reach positions on a line or a volume surface should also not be considered. Further, some kinematic schemes are equivalent, also called isomorphisms, and should be sorted out accordingly. By considering only one of the equivalent kinematics at a time and eliminating ineffectual kinematic structures, the computational effort is reduced while maintaining the same result in terms of content.

In summary, the possible kinematic chains are reduced based on the following aspects:

- Consider only prismatic P and revolute R joints
- Consider only axes orientations in x, y or z direction
- Respect task- or design-based restrictions from pre-processing
- Eliminate structures that do not span a spatial workspace
- Do not consider isomorphisms
- The last joint axis must not point in x direction of the world frame

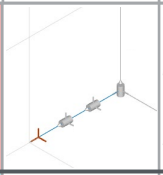
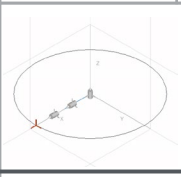
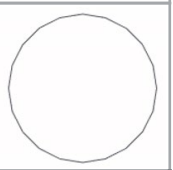
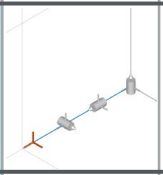
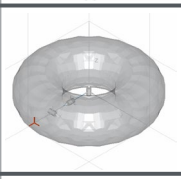
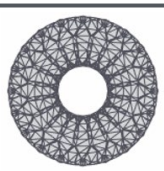
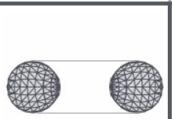
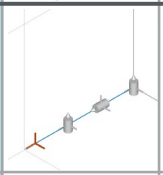
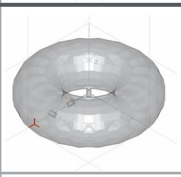
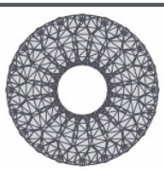
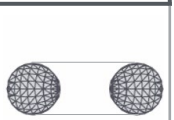
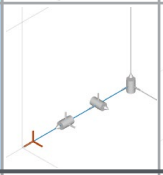
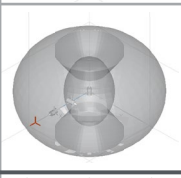
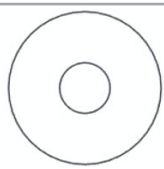
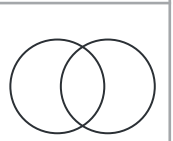
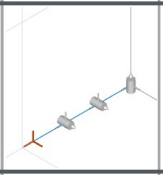
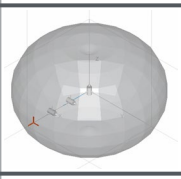
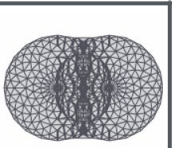
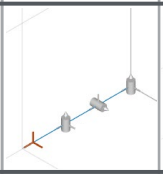
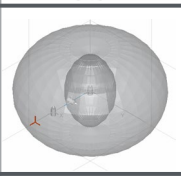
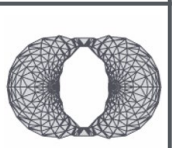
First, the generation of the kinematic schemes consisting exclusively of revolute joints is discussed in the following.

Kinematic schemes consisting of revolute joints

To determine which kinematic chains can be generally suitable, $3R$ chains are investigated first as basic structures, consisting of three revolute joints in serial arrangement. If the axis of the first revolute joint is firmly specified as z axis, nine kinematic structures can be generated as shown in Table 11 and Table 12 and they are set-up as described in the following.

A general kinematic structure in this work is created in such a way that the first joint can be placed at any position and the axis orientation is applied according to the desired x, y or z direction. Then the link length is added in the direction of the x axis of the world frame and the subsequent joint is placed to match the desired axis orientation. Again, the corresponding link length is added in the direction of the x axis of the world frame and this procedure is repeated until the desired degree of freedom is reached. In this way, the qualitative kinematics shown in Table 11 and Table 12 can be created.

Table 11 Workspace volumes and cut views dependent on kinematic structures – I

	structure	workspace	$x-y$ plane	$y-z$ plane
1 $R_z\,R_x\,R_x$				
2 $R_z\,R_x\,R_y$				
3 $R_z\,R_x\,R_z$				
4 $R_z\,R_y\,R_x$				
5 $R_z\,R_y\,R_y$				
6 $R_z\,R_y\,R_z$				

suitable kinematics

not suitable kinematics

isomorphisms; one is considered for each basic chain

kinematics considered for $dof + 1$

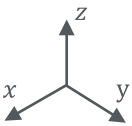
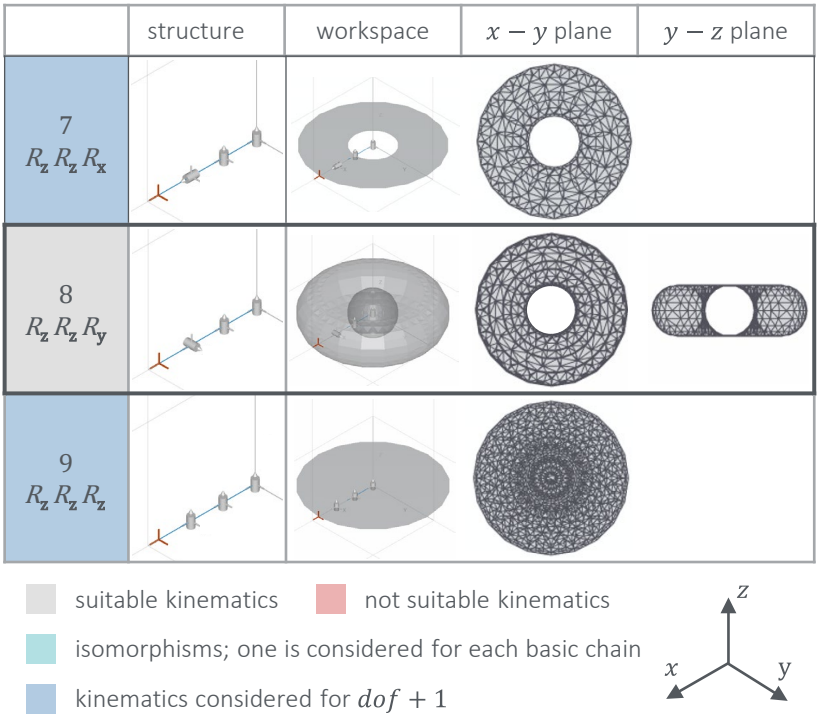


Table 12 Workspace volumes and cut views dependent on kinematic structures – II



This kind of description and representation was chosen so that developers can discuss the kinematic structures intuitively, without having to transfer them into the Denavit-Hartenberg notation or the other way around. This kind of description is more intuitive and does not require a profound knowledge but only a basic understanding in the field of kinematics. In addition, the frames of the individual structural parts are located at the joint axis where the mechanical joints are to be provided later, so that here the position can be varied and adjusted via the coordinate transformation but is clearly defined for the later design procedure. However, it is also possible to obtain the description of the kinematic schemes in the Denavit-Hartenberg notation, if required, within the provided tool.

The nine generated kinematic schemes are shown in the second row of Table 11 and Table 12 and consist of three revolute joints with their axis

pointing either in x, y or z direction as indicated by the referring index and are described as R_x, R_y or R_z joint, but the first joint of each chain shown here is set as R_z. From the visualization of the workspaces, it is obvious that depending on the joint arrangement there will be different motion possibilities for the robot, which can be built based on these kinematic schemes. The resulting workspaces and the reachable area on the $x - y$ plane and $y - z$ plane in the cut views are shown in columns three to five of Table 11 and Table 12.

Based on the joint arrangement and axis alignment, different workspaces and motion capabilities result for the kinematic chain. The workspaces shown here are the results when no axis limits are given, which is the case if all revolute joints can rotate without upper or lower bounds and for the joint parameters follows $q_i \in [-\pi, \pi]$. The workspaces would be reduced accordingly if axis limits were specified as it is described in Chapter 7.1. In addition, the workspaces shown are a qualitative representation that are dependent on the robot's link lengths. Depending on the axis arrangement and the ratio of the link lengths to each other, voids can be formed within the workspace [Gup86]. Positions inside these voids cannot be reached by the regional structure of the robot. Therefore, it is desirable that the workspace has no voids, or that they are at least as small as possible or in the area of no interest. However, to give a qualitative impression of the different workspaces based on the axis arrangement, the same lengths are chosen for all links in the given illustrations in Table 11 and Table 12. Following the rules for pre-selection of kinematic schemes mentioned above, structure 1 should not be considered, neither for $3\ dof$, nor for chains with more dof, since it consists of two successive R_x joints, which does not improve its reachability, and in addition, the structure is only capable to reach positions on the circle with its end-effector. Further, if the second axis of structure 2 is rotated by $90°$, structure 3 is obtained and therefore, is an isomorphism and it is sufficient to consider just one of the two kinematic schemes. The structures $4, 7$ and 9 have an R_x joint as the last joint, so that they are not considered for $3\ dof$ structures because the last axis only allows a reorientation of the end-effector. However, for kinematics with more degrees of freedom this arrangement can be useful and thus is not eliminated completely. Structure 9 has a planar workspace and is therefore not suitable for structures with

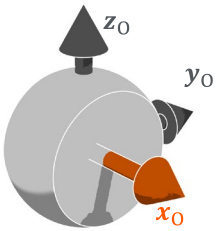
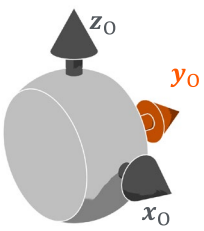
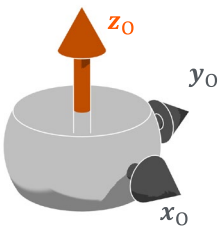
3 dof neither. Nevertheless, this is a sub-chain that is suitable for structures with more degrees of freedom and can result for instance in the well-known SCARA kinematics. In summary, the bold framed structures $2, 5, 6$ and 8 from Table 11 and Table 12 are appropriate $3R$ chains, so that it is a total of four combinations with their first axes pointing in the z direction as given in the second column of Table 13.

Table 13 Comparison of pre-selection vs. all possible combinations – R joints

	considering just R joints: nR chains with $n = dof$ preselection vs. all possible combinations						all possible P and R joint combinations
dof	first joint axis aligned with z direction of world frame: $a_{ori,1} = z$			arbitrary first joint axis: $a_{ori,1} = x, a_{ori,1} = y$ or $a_{ori,1} = z$			arbitrary first joint axis: $a_{ori,1} = x, a_{ori,1} = y$ or $a_{ori,1} = z$
3	4	44.4 %	9	12	44.4 %	27	216
4	11	40.7 %	27	33	40.7 %	81	1296
5	28	34.6 %	81	84	34.6 %	243	7776

In [Sch87] for example, a total of three structures is selected as useful $3R$ chains for the regional structure, namely the structures $5, 6$ and 8 from Table 11 and Table 12. Nevertheless, in the description and generation of the structures used in this work, it is reasonable to also consider structure 2 from Table 11 and to not eliminate this kinematic scheme from the solution field. The workspaces shown in Table 11 and Table 12 result from the kinematic schemes with their first axis of the revolute joint being aligned with the z axis of the world frame. To generate all possible combinations out of the three revolute joints, each considered kinematic scheme with the first axis pointing in z direction must be re-oriented in such a way that the first joint axis points in the x or, for the third possible orientation, along the y axis, respectively. This reorientates the qualitative workspaces as shown in Table 14 and thus, the total amount of $9 \cdot 3 = 27$ possible kinematic structures can be generated based on all of the $3R$ kinematic chains from in Table 11 and Table 12 and the number of variations is given on the right in the second column of Table 13.

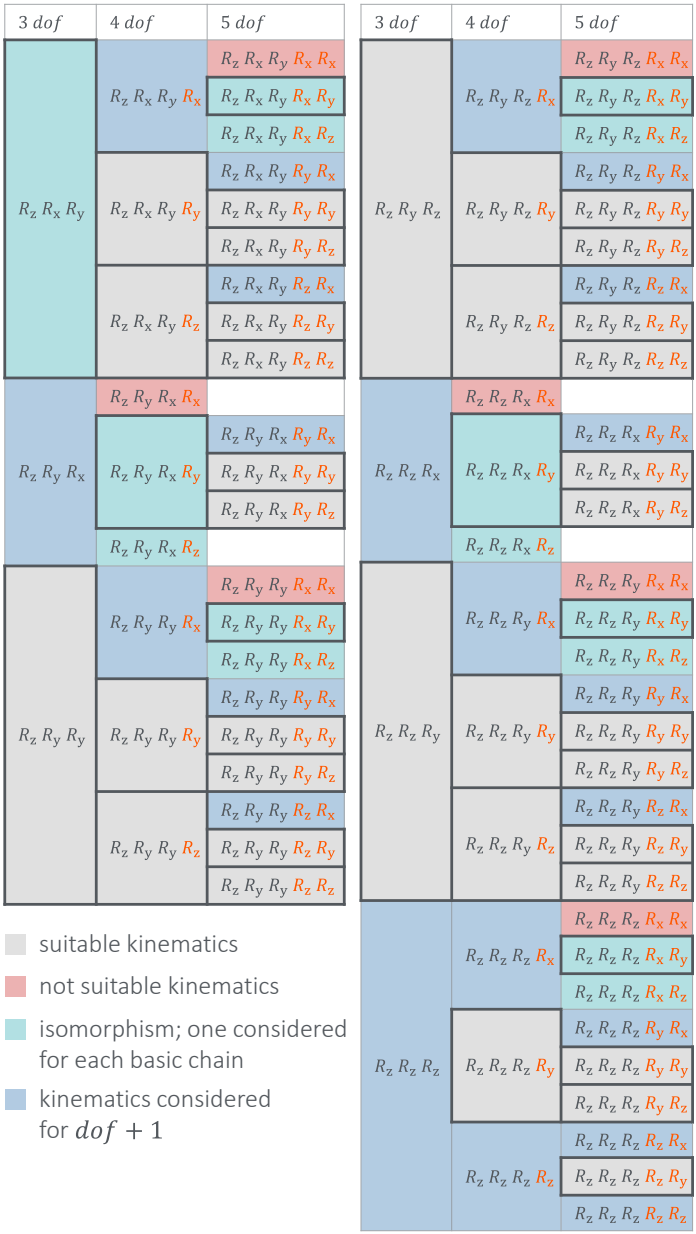
Table 14 Workspace shaping in dependence of the first joint axis orientation

$a_{\mathrm{axi},1} = x$	$a_{\mathrm{axi},1} = y$	$a_{\mathrm{axi},1} = z$
		

The four bold framed preselected $3\,dof$ kinematic schemes result accordingly in a total of $4 \cdot 3 = 12$ structures as given on the left of the second column of Table 13. Caused by the reorientation of the four basic kinematic schemes, triple sets of the twelve generated kinematics are always isomorphisms and they just differ in their first axis' orientation, but they will be considered independently if desired by the user, since the orientation of the first axis can strongly influence the reachability of the specified positions. Therefore, depending on the application, the orientation of the joint axes can be predefined if necessary. If no special requirements have been made, all possibilities will be considered successively.

To generate kinematics with 4 dof, the seven usable $3\,dof$ basic schemes from Table 11 and Table 12 are used as given in Table 15. They are composed of one of the structures marked in green (2) and all the structures marked in gray $(5, 6, 8)$ and blue $(4, 7, 9)$. The colors are indicating the suitability of the scheme with respect to the previously mentioned selection guideline. Using these pre-selected structures, the other kinematics with four degrees of freedom can be generated. To do so, another R joint with corresponding orientations can be attached to the respective basic chains. For example, based on the $R_\mathrm{z}R_\mathrm{x}R_\mathrm{y}$ chain, three new $4\,dof$ structures can be generated by adding an R_x, R_y or R_z joint, respectively. Out of those, only the two marked in gray are useful as $4\,dof$ chains, and thus count to the $4\,dof$ useful chains. The $R_\mathrm{z}R_\mathrm{x}R_\mathrm{y}R_\mathrm{x}$ chain marked in blue has an R_x joint as last joint, and thus is only to be considered if the $5\,dof$ chains are generated based on the $4\,dof$ chains, as it is given in the third column of Table 15.

Table 15 Overview of generated structures and their classification



In summary, the suitable structures are selected according to the previously mentioned criteria. Based on the seven $3\,dof$ basic structures, there are a total of eleven structures with the first axis pointing in the z direction of the world frame. From these suitable $4\,dof$ schemes a total of 33 possible arrangements can be generated respecting variations in the first axis orientation along the x and y axis, respectively.
The green marks in Table 15 indicate isomorphisms, of which only one structure at a time will be further considered. Gray indicates the structures that are suitable for the desired degree of freedom. These structures suitable for the corresponding degree of freedom, together with the structures marked in blue, form the new basic chains for structures with an additional degree of freedom, namely $dof = 5$ in this case. The red marked structures are neither useful chains for the corresponding degree of freedom, nor will they be used as basic chain to generate structures with an additional degree of freedom and are thus not considered further.
If this method is continued, 28 solutions for structures with $dof = 5$ are obtained based on the eleven $4\,dof$ basic chains as highlighted with bold frames in Table 15. Multiplied by three for the corresponding orientations of the first joint axis in x, y and z direction of the world frame, this results in a total of $28 \cdot 3 = 84$ kinematic chains to be considered for $5\,dof$ structures.
The pre-selection significantly reduces the number of necessary kinematics to be considered. The comparison of the number of structures is given in Table 13 for $dof = 3$ to $dof = 5$. If for example $5\,dof$ structures are investigated, only $34.6\,\%$ of all possible combinations must be considered as indicated by the grey bar. This reduction considerably improves the computation time by pre-eliminating unsuitable kinematics even before the dimensional synthesis. Using the presented approach, kinematic regional structures from $dof = 3$ to $dof = 5$ can be efficiently generated and used for the further synthesis procedure.

Since all the structures presented so far consist of revolute joints, but chains with prismatic joints are also to be considered, the following sections discusses the integration of prismatic joints as first or last joint within the kinematic scheme.

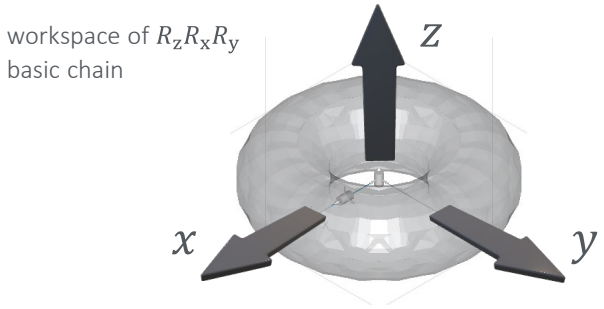
Kinematic schemes including prismatic joints

For some handling tasks it might be useful or even necessary to have one or more prismatic joints within the kinematic chain. In this work, three possible arrangements with prismatic joints are primarily considered, with $n = dof$:

- First joint is a P joint, rest consist of R joints: $P, (n-1)R$
- Last joint is P a joint, rest consists of R joints: $(n-1)R, P$
- First and last joint are P joints, rest are R joints: $P, (n-2)R, P$

If a prismatic joint is integrated in front or after structures consisting of revolute joints, the new workspace is expanded in the direction of the joint axis. The new workspace is thus obtained by extruding the workspace or work surface spanned by the revolute joints along the prismatic joint axis. Depending on the position of the prismatic joint placement within the kinematic chain, its range of motion can differ. For example, it is possible to make the stroke significantly larger from a mechanical perspective if the prismatic joint is the first joint, compared to the placement as last joint. Figure 6.2 gives a qualitative overview of how the workspace of the $3\ dof$ $R_\mathrm{z}R_\mathrm{x}R_\mathrm{y}$ basic chain expands when a prismatic joint is added. Depending on the orientation of the joint axis, the workspace is extruded in varying directions. Adding a P_x or R_y joint results in an identical workspace apart from the orientation in space if the workspace of the base chain is rotationally symmetrical about the z axis as it is in this example.

Depending on the placement of the prismatic joint, the useful structures for the desired degree of freedom can be generated in different ways. To begin with, the generation of structures that have the prismatic joint in first place are discussed.



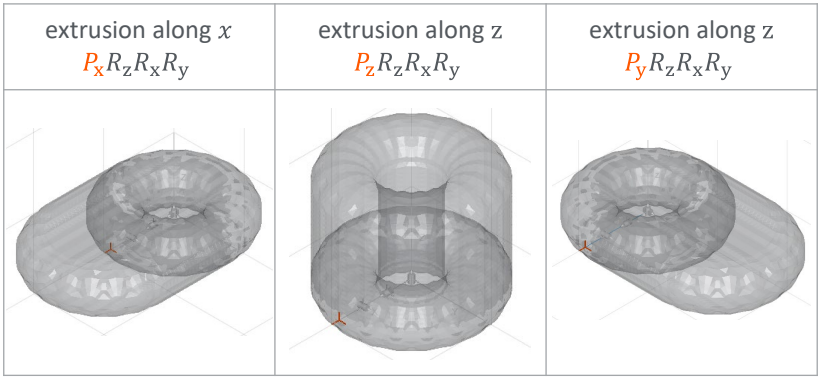


Figure 6.2 Workspace generation including a P joint in the kinematic scheme

First joint is a P joint, rest consist of R joints: $P,(n-1)R$ schemes

For $3\,dof$ chains with the first joint being a prismatic joint, four useful structures can be generated. The axis of the prismatic joint is aligned along the z axis in all structures. In analogy to the nR chains, the workspace can be reoriented in the three main orthogonal directions x, y and z of the world frame, resulting in a total of $4 \cdot 3 = 12$ structures as given in the row for $3\,dof$ structures in Table 16.

To create the $4\,dof$ schemes, the total of the four bold framed useful $3\,dof$ structures from Table 15 are used and a prismatic joint P_x, P_y or P_z is placed ahead of them. If the P_z joint is used, the $R_zR_zR_z$ structure is also considered, since the planar workspace normal to the z axis being extruded along the z axis creates a workspace suitable for many applications. The resulting $P_zR_zR_zR_z$ kinematic chain is a special form of the known SCARA kinematics

and must also be considered for $4\ dof$ schemes. This results in a total of five structures when using P_z as first joint. Adding a P_x or P_y as first joint, a total of four usable chains can be generated each. The last two workspaces are identical and only oriented differently in the world coordinate system. Alternatively, one of the workspaces can also be created by rotating the other workspace around the z axis. Even if the kinematics are identical, it can still be useful to consider them individually, so that the user can simply specify the main direction of the workspace in the pre-processor without worrying about how the kinematics must be transformed about which axis and is therefore more intuitive. The resulting $5 + 4 + 4 = 13$ structures with $4\ dof$ and P_z, P_x and P_y as first joint can be reoriented again along the world frame axes in analogy to Table 14, resulting in a total of $3 \cdot 13 = 39$ possibilities as given in Table 16 in the reorientation column for $4\ dof$ schemes.

Table 16 Comparison of pre-selection vs. all possible combinations – $P, (n-1)R$

	considering one P joint: $P, (n-1)R$ preselection vs. all possible combinations						P and R joint combinations
dof	$a_{ori,1} = z$			$a_{ori,1} = x, y$ or z			$a_{ori,1} = x, y$ or z
3	4	44.4 %	9			12	44.4 %	27	216
4	5	18.5 %	27	13	16 %	81	39	48.1 %	81	1296
5	12	14.8 %	81	34	14 %	243	102	42%	243	7776

The same procedure can also be used for structures with an additional degree of freedom resulting in $5\ dof$ schemes. The new generated structures are based on the $4R$ chains from Table 15 and the resulting number of possible structures is shown in Table 16. A comparison between the preselection related to all possible combinations shows that in the 5 dof example only $42\ \%$ of all kinematic chains must be considered, which also enables an improvement in the computational effort and a better overview. Another possibility of using a prismatic joint is not using them a first joint as discussed above, but to provide this joint as the last joint of the kinematic chain and this is explained in the following.

Last joint is a P joint: $(n-1)R,P$ or $P,(n-2)R,P$ schemes

The possibility of using a prismatic joint as the last joint of the kinematic chain should be applicable both to schemes with only revolute joints, and to structures where the first joint is a prismatic joint. In general, such a last joint would be assigned to the robot's tool and not to the regional or local structure. Accordingly, it is not represented in the scope of synthesis for the regional structure that is presented here. Nevertheless, to give the user a possibility to generate reasonable solutions for these requirements as well if needed, this case can be handled via the task description within pre-processing and the proposed approach is explained in the following.

To also include P joints as last joint of the robot kinematic, the user must define the target positions in such a way that they cover the desired points that are to be reached with the regional structure. In this context also the optional orientation requirements can be specified. However, the points that are to be reached with the tool are not explicitly specified. Instead, the user must ensure that the preset positions are selected in such a way that they can be used as a starting point and that the stroke can be realized from there by using the last prismatic joint. Optional changes to the orientation of the tool can be implemented using the robot wrist if one is provided within the complete kinematic scheme.

If, for example, the kinematics for the box scenario from Figure 5.9 might be equipped with a prismatic joint within the robot's tool, it is no longer necessary to specify all the points which span the volume inside the box for the synthesis, but only those points which span the opening area for entering the box. If the last link is oriented in such a way that it points parallel to the sides of the box, a prismatic joint can simply be attached to that structural part. For the later design procedure, it is important to ensure that the stroke of the prismatic joint is large enough for the robot to reach the entire desired volume. To achieve this, the depth of the box can be specified as a reference value for the stroke length in the case of a normal stroke system, or in case of a telescopic stroke it can be half the depth of the box. These design advices are given here because this kind of kinematics is not mentioned separately in the dimensional synthesis.

Based on the user's input and the processing of the data within the presented part of the solver, all kinematic structures to be considered can be generated. The number of structures depends on the selected degree of freedom and additional boundary conditions regarding the joints that was done by the user within pre-processing as explained in Chapter 5. The resulting numbers of different kinematic schemes to be considered for the different options are given in Table 13 and Table 16. Through the pre-processing also the corresponding link lengths are already known and stored in a total of l_{nr} link length sets. As a preparation for the dimensional synthesis, all robot objects to be investigated are automatically generated within the solver. As mentioned earlier in Chapter 5, the MATLAB tool developed mainly consists of the GUI, the main program that uses the two classes *c_robotlist* and *c_robot*. The properties of *c_robotlist* were defined within pre-processing, and the class *c_robot* is used for the following procedure. For each possible kinematic scheme an object of the class *c_robotlist* is generated and within for each link length set an object of the class *c_robot* is created and the properties for kinematic type and link lengths are set automatically. An overview is given in Figure 6.3 and the white robot object boxes are the objects to be considered for further processing. The number of all robot objects r_{all} to be investigated within the dimensional synthesis is given by equation (6.3), since for each of the r_{nr} investigated kinematic scheme one robot object is created per link length set.

$$r_{\mathrm{all}} = l_{\mathrm{nr}} r_{\mathrm{nr}} \tag{6.3}$$

To summarize, based on the user's specifications regarding the task (Chapter 5.1) and boundary conditions (Chapter 5.2) within pre-processing, all objects required for the further investigation during the dimensional synthesis in the following Chapter 6.2 are automatically generated within the MATLAB environment of the solver.

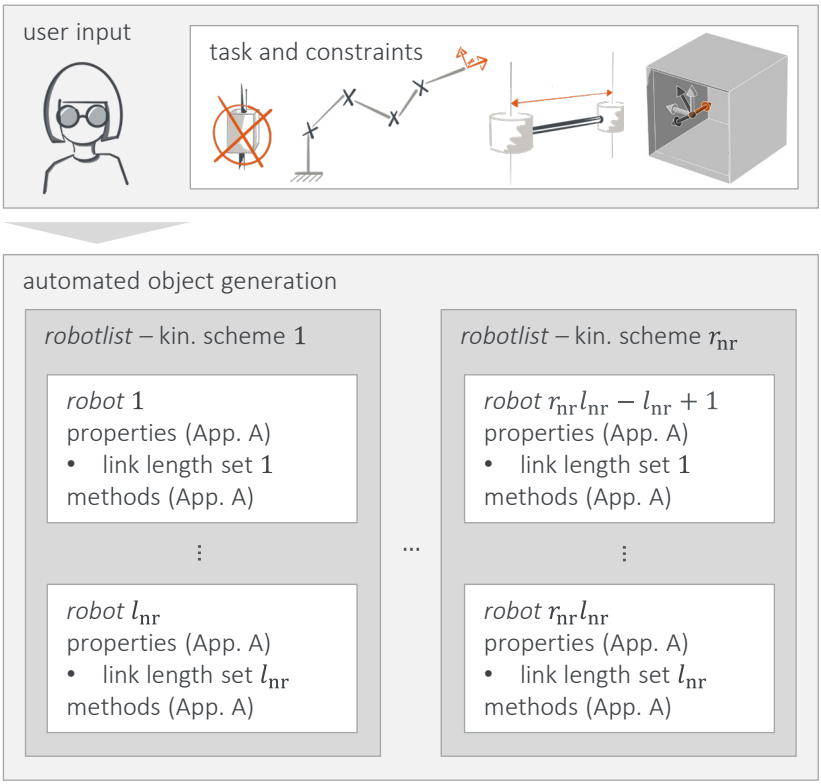
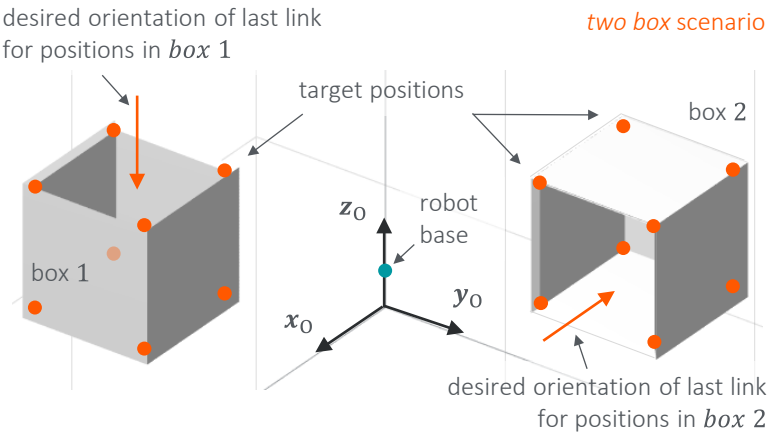


Figure 6.3 Automatic object generation based on user input

For the *two box* scenario already introduced in Chapter 5.2, the number of robot objects r_{all} can be specified as given in Figure 6.4 and they are generated with the desired properties automatically. The kinematic schemes to be considered result from the specification of the $5\ dof$ as well as the restriction that only revolute joints are allowed, and the first axis must be oriented in z direction. The $r_{nr} = 28$ schemes to be considered result from these requirements and the approach for generation of the kinematic types presented in this chapter. Multiplying the number of link length sets $l_{nr} = 237$ results in the number of robot objects. These objects will be investigated in more detail in the following chapter on dimensional synthesis.



property	value
position + orientation	see figure above
robot base	$(0\ 0\ 0.3)^T$
dof	5
joint constraints	consider only R joints first joint axis along z: $a_{\mathrm{ori},1} = z$
link length bounds	$l_{\min} = [0.1\ 0.1\ 0.1\ 0.1\ 0.4]$ $l_{max} = [0.5\ 0.5\ 0.5\ 0.5\ 0.8]$ $s = 0.2$ $l_{\mathrm{nr,pr}} = 243 > l_{\mathrm{nr}} = 237$
number of robot objects	$r_{\mathrm{all}} = 237 \cdot 28 = 6636$

Figure 6.4 Properties resulting for the *two box* scenario after type synthesis

6.2 Dimensional Synthesis

During the dimensional synthesis the design parameters for the referring kinematic schemes are identified so that the resulting robots can fulfill the given task and reach all target positions under the given constraints.
In this chapter, the general procedure of the automated dimensional synthesis is discussed first. Afterwards, the algorithm used to solve the inverse kinematics is addressed. Various parameters such as joint limits and gain factors, which influence the performance of the algorithm are investigated. Furthermore, the error calculation is explained and how it fits with the specifications of the pre-processor. Finally, it is discussed how the suitable kinematics are determined and pre-sorted for further investigation that can be done by the user within post-processing.

General procedure

Within the type synthesis described in Chapter 6.1 all robot objects to be further evaluated were generated. For each kinematic scheme there is one object for each link length set to be considered as visualized in Figure 6.3. For all these objects of the class *c_robot*, it must now be investigated whether they can reach all the p_{nr} specified desired positions that describe the task, while also considering the orientation constraints if these are given. In general, it is not important here to know with which joint parameters this can be achieved, because it is sufficient to know whether there is a solution of the inverse kinematics for a target position and therefore it can be reached by the robot, or not and the object therefore does not represent a suitable solution. However, this problem can be defined as an inverse kinematic problem. Even if the joint parameters are irrelevant, the necessary information can be generated by solving the inverse kinematic problem, namely, to identify whether the robot object can reach the target without significant error between actual and desired position. One advantage of knowing the joint parameters is that the solutions for all subsequent targets can be generated more efficiently if the kinematics is already in the right configuration and close to the area of the target position since the utilized algorithm is highly dependent on the initial values. In addition, it can also be ensured that not only individual positions can be reached, but that they can

be approached one after the other in the sequence of target positions defined by the user. In total, the maximum number of necessary solution attempts of the inverse kinematic problem results according to equation (6.4). It is dependent on the number of link length sets l_{nr}, the number of kinematic schemes to be considered r_{nr} and the number of target positions p_{nr}.

$$k_{\max} = r_{\mathrm{all}} p_{\mathrm{nr}} = l_{\mathrm{nr}} r_{\mathrm{nr}} p_{\mathrm{nr}} \tag{6.4}$$

To reduce the computational effort by decreasing the number of attempts to solve the inverse kinematic problem $k_{\max}$, the procedure given in Figure 6.5 and an example for one robot scheme depicted in Figure 6.6 is applied. After all robot objects within the robot list objects have been created automatically as described in Chapter 6.1 and visualized in Figure 6.3, it can be determined whether robot objects from the given robot lists representing the kinematic schemes to be considered are suitable by solving the inverse kinematic problem.

For this purpose, the first kinematic scheme under consideration $(k = 1)$ is consulted and all l_{nr} robot object numbers are stored in the first row of the matrix with successful objects $successKin$ since all of them must be evaluated. Then the IK is calculated for the first target position $(n = 1)$ for the first robot object $(i = 1)$, thus the object number $successKin(n, i)$ is evaluated. If the error between target and actual position is smaller than the maximum tolerated $error_{\mathrm{tol}}$, the robot object is considered successful for the position n under evaluation and the referring object number is stored accordingly in $successKin(n + 1, count)$. Then, the counter is then increased by one, so that the subsequent successful robot can be saved accordingly. If the error is greater than the permissible error, the robot object under consideration is no longer included for any further calculations. Instead, it is directly defined as unsuitable, since it is a necessary criterion that a suitable robot can reach all specified positions within the given tolerances. Then the counter i is increased by one and accordingly the next robot object already successful for all previous positions evaluated so far is evaluated for the currently considered position n by means of IK.

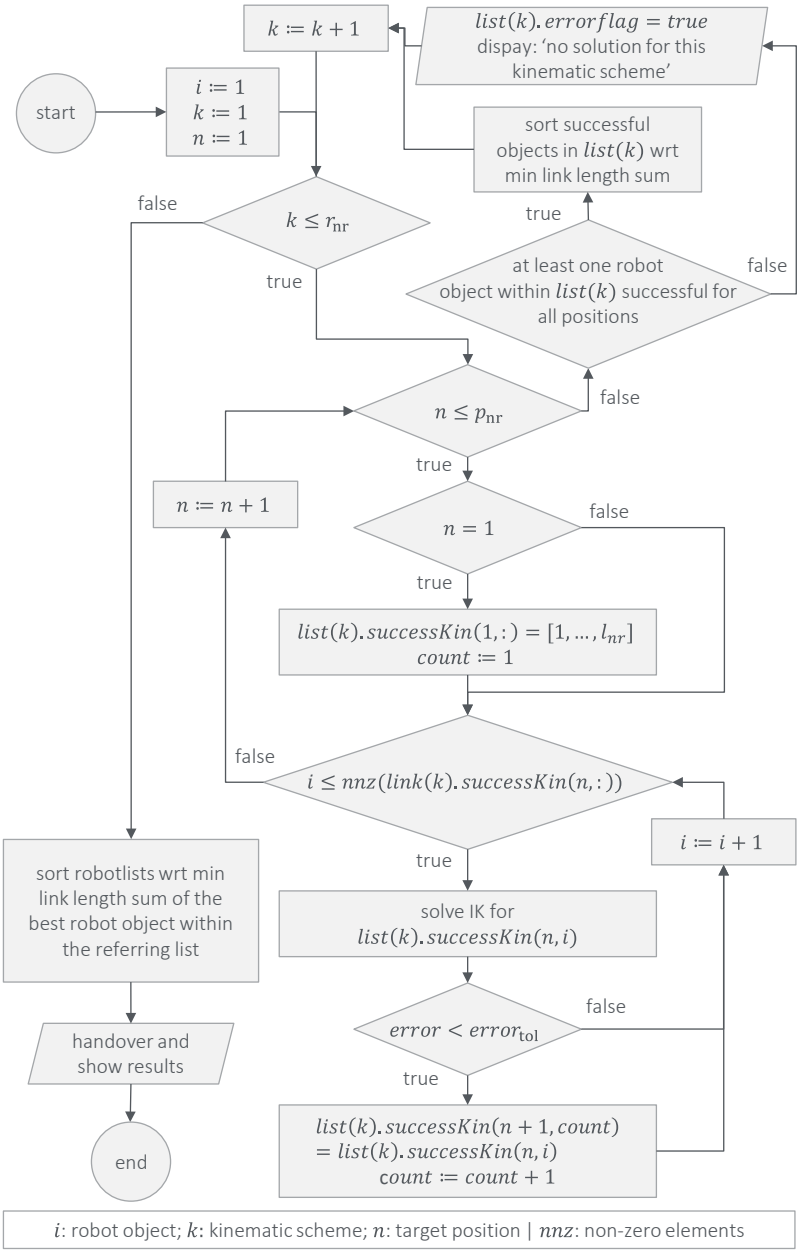


Figure 6.5 Synthesis procedure flow chart

Depending on whether there is a solution of the inverse kinematics within the given tolerance range or not, the robot object number is stored as successful or discarded as described before. If the current position is evaluated for all objects of interest, the counter n is increased by one and the same procedure is performed accordingly for the position $n+1$. This is continued until all p_{nr} positions are evaluated for all robot objects that have been successful so far and then the counter k is increased by one and thus the next kinematic scheme is considered. This procedure is carried out until the already successful robot objects for the respective positions have been evaluated by means of IK for all robot lists and have been saved or discarded. The successful objects are sorted within the corresponding list by their minimum link length and finally the lists are pre-sorted among each other with respect to the minimal link length sum.

Figure 6.6 visualizes an example of the selection of suitable objects for the scheme $R_{\mathrm{z}}R_{\mathrm{y}}R_{\mathrm{y}}$ with the referring link length sets and illustrates how just the already successful objects are used for the further calculations. It is noticeable how the successful objects decrease with each further position and only the suitable configurations are considered in the further process, and thus the computational efficiency is increased since the number of attempts to solve the IK is reduced.

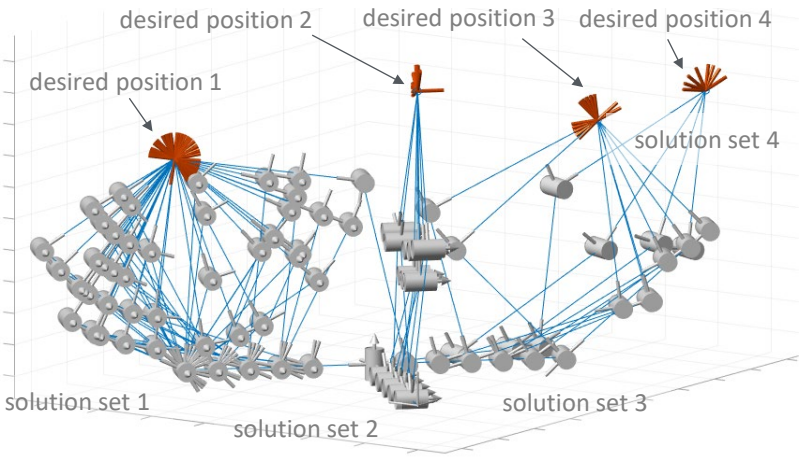


Figure 6.6 Reduction of object in solution set for each further position

To conclude, the result of the dimension synthesis are all robot objects that can reach the desired target positions. The robot objects are stored within the robot lists regarding their kinematic scheme and presorted automatically with respect to the minimum link length sum. An overview of all suitable schemes and the corresponding appropriate link length sets that enable the referring kinematics to reach all target positions are provided as the output of the solver. Also, there is a visualization and animation of the robot objects that can be used by the developer for further evaluation and selection within the framework of the pre-processor as described in Chapter 7.1. For calculations and visualization, functions of the robotics system toolbox [Mat20] were used.
After the general procedure is described, the approach for solving the inverse kinematics is outlined in the following.

Solving the inverse kinematic problem via saturation in the null space

In the context of this work, it is necessary to solve inverse kinematics for a variety of different kinematic chains with different link length sets. Often redundant structures can be involved and in general it can be useful if joint limits can also be considered directly within the IK. Furthermore, the method used to solve the IK must allow that additional boundary conditions can be defined, such as the orientation specification of the last link of the structure without over constraining the task as described in Chapter 5.1. In addition, it is useful to apply methods and algorithms that are already successfully applied and investigated.
These requirements lead to the numerical method of saturation in the null space (SNS) presented in [FLK12] as chosen approach for solving the IK and this algorithm is integrated to the solver of the development tool.
The SNS algorithm offers an efficient way to locally solve the inverse differential kinematics problem in robotics on joint space level. This method is also applicable if the investigated kinematic is redundant or task redundant. Additional boundary conditions such as joint limits as well as velocity and acceleration bounds can be respected simultaneously. Successively, the algorithm restricts the use of joints that would exceed their limits and performs the motion with the remaining joints, that are not saturated and are free to move. [FLK12]

As visualized in the procedure overview in Figure 6.5, the IK is calculated for each robot object for each target position if the corresponding object was able to reach all previous positions. As an additional constraint joint limits can be defined but for the first dimensioning joint angles are not restricted for revolute joints, so that for the joint parameters it follows $q_{i,\mathrm{r}} = [-Inf; Inf]$ to ensure a more robust process, and for prismatic joints, the limits are defined depending on the corresponding link length l_i either to $q_{i,\mathrm{p}} = [0, l_i]$ or $q_{i,\mathrm{p}} = [-l_i/2\,, l_i/2]$ as desired by the user. If velocity or acceleration limits are known, they can be defined and will be considered in further calculations. To be able to perform the calculations, a starting configuration must be specified for the algorithm and can be used as values for the joint parameters from which the calculation is to be started. Whether and how fast the algorithm successfully finds a solution strongly depends on the selected start configuration. In the first iteration step for the first position to be calculated, the home configuration is used for this purpose. If no home configuration was explicitly defined by the user within the pre-processing, it is set to $\boldsymbol{q}_{\mathrm{home}} = \mathbf{0}^T$ by default. For all further calculations the joint values of the previous calculation are used as the new starting values for the following iteration.

Based on the initial joint values, the pose of the end-effector is calculated using forward kinematics. Then the error between target and actual position can be computed. The way how this is done depends on whether an orientation condition was additionally defined for the target position within the pre-processor or not as given in equation (6.5). If this is not the case, the position error is calculated by means of vector subtraction of the desired and the current position by equation (6.6).

$$error = \begin{cases} [\boldsymbol{p}_{\mathrm{des}} - \boldsymbol{p}_{\mathrm{cur}}], & no\ orientation\ constr. \\ \begin{bmatrix} \boldsymbol{p}_{\mathrm{des}} - \boldsymbol{p}_{\mathrm{cur}} \\ \boldsymbol{\phi}_{\mathrm{error}} \end{bmatrix}, & orientation\ constr. \end{cases} \tag{6.5}$$

$$[\boldsymbol{p}_{\mathrm{des}} - \boldsymbol{p}_{\mathrm{cur}}] = \begin{bmatrix} p_{\mathrm{des,x}} - p_{\mathrm{cur,x}} \\ \mathrm{p}_{\mathrm{des,y}} - p_{\mathrm{cur,y}} \\ p_{\mathrm{des,z}} - p_{\mathrm{cur,z}} \end{bmatrix} \tag{6.6}$$

If an orientation requirement is also given, the orientation error must also be determined and considered in addition to the position error. For this purpose, the angle-axis representation is chosen to calculate the error $\boldsymbol{o}_{\mathrm{error}}$ between the vectors describing the desired orientation and the current orientation (6.7). The desired orientation $\boldsymbol{o}_{\mathrm{des}}$ was given by the user within the pre-processing by means of a vector definition and the actual orientation $\boldsymbol{o}_{\mathrm{cur}}$ results from the current orientation of the x unit vector of the end-effector frame with respect to the world coordinate system. Alternatively, the desired specification can also be set within the pre-processor based on the y or z unit vector of the end-effector if this is required for the task description. The axis around which the current vector $\boldsymbol{o}_{\mathrm{cur}}$ must be rotated to be mapped onto the desired vector $\boldsymbol{o}_{\mathrm{des}}$ results from the cross product of these two vectors and is therefore the normal vector to the plane that is spanned by those. The angle θ around which the corresponding rotation must be performed can be calculated using (6.7) and the resulting orientation error $\boldsymbol{\phi}_{\mathrm{error}}$ is given by equation (6.8) based on the angle θ and the normalized vector describing the axis of rotation $\boldsymbol{o}_{\mathrm{a}}$.

$$\theta = \operatorname{atan2}(\|\boldsymbol{o}_{\mathrm{cur}} \times \boldsymbol{o}_{\mathrm{des}}\|, \boldsymbol{o}_{\mathrm{cur}} \cdot \boldsymbol{o}_{\mathrm{des}}) \tag{6.7}$$

$$\boldsymbol{\phi}_{\mathrm{error}} = \theta \frac{\boldsymbol{o}_{\mathrm{cur}} \times \boldsymbol{o}_{\mathrm{des}}}{\|\boldsymbol{o}_{\mathrm{cur}} \times \boldsymbol{o}_{\mathrm{des}}\|} = \theta \boldsymbol{o}_{\mathrm{a}} = \theta \begin{bmatrix} o_{\mathrm{a,x}} \\ o_{\mathrm{a,y}} \\ o_{\mathrm{a,z}} \end{bmatrix} = \begin{bmatrix} \phi_{\mathrm{x}} \\ \phi_{\mathrm{y}} \\ \phi_{\mathrm{z}} \end{bmatrix} \tag{6.8}$$

The error can thus be calculated depending on whether an orientation specification is to be considered or not and is used to calculate the new end-effector velocities to approach the target position following (6.9). The gain factor amplifies the error so that the algorithm converges faster.

$$\dot{\boldsymbol{x}} = gain \cdot \boldsymbol{error} \tag{6.9}$$

If the norm of the error is greater than the permissible error as visualized exemplarily in Figure 6.7 and while the maximum number of iterations has

not been exceeded, the program continues to transfer the current position $\boldsymbol{p}_{\mathbf{cur}}$ to the desired position $\boldsymbol{p}_{\mathbf{des}}$ via the SNS algorithm.

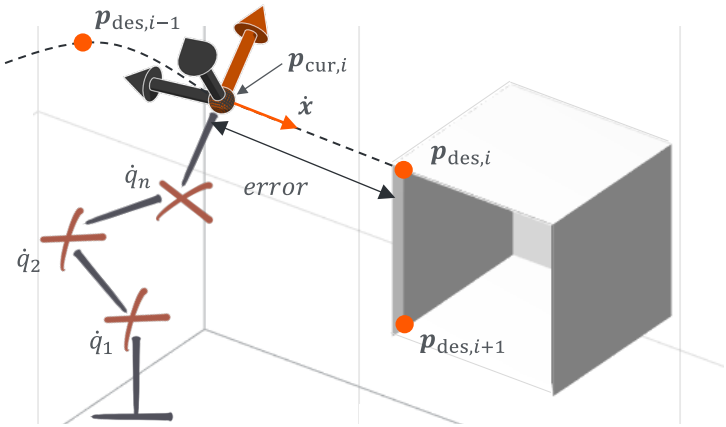


Figure 6.7 Kinematic approaching desired positions by minimizing the error

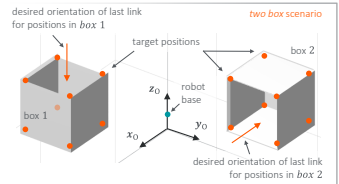
For this purpose, the Jacobian $\boldsymbol{J}$ is needed and automatically generated for the corresponding kinematics and evaluated for the current configuration based on the fundamentals described in Chapter 2.2.6 and using MATLAB's robotics system toolbox [Mat20]. Then the limits for the SNS are calculated based on the joint limits $\boldsymbol{q}_{\mathbf{lim}}$, speed and acceleration limits, the current configuration $\boldsymbol{q}_i$ and the given time step $\boldsymbol{t}_{\mathbf{s}}$ for iteration. The calculated limits, in addition to the current end-effector velocity $\dot{\boldsymbol{x}}$, the Jacobian and the maximum number of allowed iterations, provide the input data for the SNS algorithm. As output results among others the joint velocity $\dot{\boldsymbol{q}}_{\mathbf{SNS}}$ (6.10) corresponding to the end-effector velocity $\dot{\boldsymbol{x}}$, the matrix $\boldsymbol{W}$ that specifies which joints are saturated, the task scaling factor $\boldsymbol{s}$ and the null-space vector $\boldsymbol{q}_{\mathbf{N}}$. Based on this the new end-effector position of the next iteration step can be derived via integration as given in equation (6.11). A detailed description of the algorithm and the procedure is given in the corresponding literature [FLK12].

$$\dot{\boldsymbol{q}}_{\mathbf{SNS}} = \dot{\boldsymbol{q}}_{\mathbf{N}} + (\boldsymbol{JW})^{\#}(s\dot{\boldsymbol{x}} - \boldsymbol{J}\dot{\boldsymbol{q}}_{\mathbf{N}}) \tag{6.10}$$

$$\boldsymbol{q}_{i+1} = \boldsymbol{q}_i + \dot{\boldsymbol{q}}_i t_s \tag{6.11}$$

Based on the new joint variables the current end-effector position and the error between this position and the desired target position are calculated again using the forward kinematics and the process is repeated.
If the error is smaller than the permitted tolerance, the process is terminated successfully because a suitable solution has been found. In case that the maximum number of allowed iterations is reached, the process is also terminated, but the kinematics under consideration is identified as unsuccessful and is no longer considered in the further procedure as shown in the process overview in Figure 6.5. After successful calculation, the next position can be evaluated. The new start values for the first iteration for the new position are those joint variables that were used to reach the previous target position. This procedure does not only ensure that all desired positions can be reached in general but also that the kinematics can approach any position from the corresponding previous position, since the IK is solved at velocity level.
When the calculations for all robot object and positions are completed, the structures with corresponding link length sets are determined, which can successfully reach all desired target positions as shown for the *two box* scenario in Figure 6.8 that was already introduced to illustrate the method via an exemplary application.

The output of the solver which is then the input for the post-processing is thus a first overview of the suitable kinematic schemes as well as the corresponding dimensions with which these structures can fulfil the desired task that was defined within the pre-processor. Depending on the task and the given boundary conditions, the number of solutions may vary. If there are many possibilities, it is helpful if the user is supported in the selection and evaluation process. This is done within post-processing as described in the following Chapter 7.



property	value
position + orientation	see figure above
robot base	$(0\ 0\ 0.3)^T$
dof	5
joint constraints	consider only R joints first joint axis along z: $a_{\mathrm{ori},1} = z$
link length bounds	$l_{\min} = [0.1\ 0.1\ 0.1\ 0.1\ 0.4]$ $l_{max} = [0.5\ 0.5\ 0.5\ 0.5\ 0.8]$ $s = 0.2$ $l_{\mathrm{nr,pr}} = 243 > l_{\mathrm{nr}} = 237$
number of robot objects	$r_{\mathrm{all}} = 237 \cdot 28 = 6636$

evaluation via solution of IK for all positions for 6636 robot objects from 28 kinematic schemes

list with successful objects

kin. scheme	link length set nr
$R_zR_xR_yR_xR_y$	74,75,127,154,181,182,208
$R_zR_xR_yR_zR_y$	75,155,181,209
$R_zR_yR_xR_yR_y$	47,65,76,100,118,136
$R_zR_yR_yR_xR_y$	154
$R_zR_yR_yR_zR_y$	152
$R_zR_yR_zR_xR_y$	172,178,190,196,202

Figure 6.8 Identifying successful robot objects for the box scenario

7 Post-processing – evaluation and design

The last semi-automated process steps within the presented robot design procedure are evaluation, selection and design tips within the post-processing. An overview of input, content within the procedure and output as well as the integration into the overall process are shown in Figure 7.1

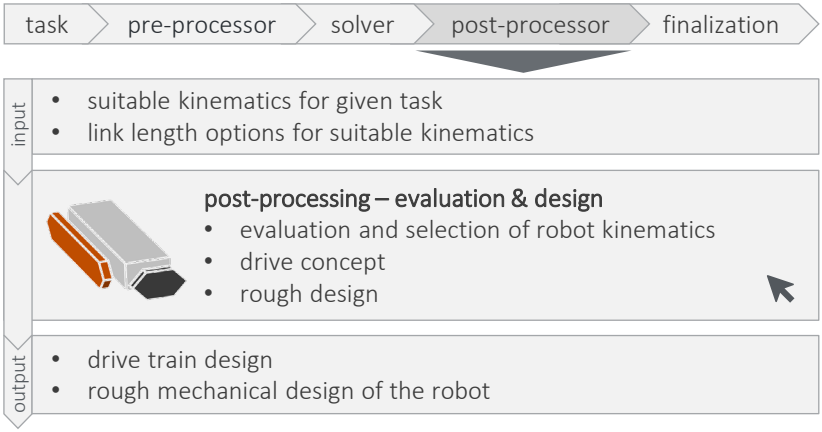


Figure 7.1 Post-processing within the development process

The output data of the solver is used as input for post-processing. These are the robot objects which can fulfill the required task and for each suitable kinematic structure there is one robot object for each link length set. These objects must be evaluated and selected in the next step, which is the first phase within post-processing that is explained in Chapter 7.1. To select a suitable robot object, the user is supported in the selection and targeted modification of the dimensioning so that one of the successful kinematics can be finalized. Afterwards, the rough design of the robot can then be carried out based on the chosen robot objects. Talking about the design of the robot, the aspect of safety can play an essential role depending on the application. Especially in the field of service robotics this can play a significant role and therefore Chapter 7.2 describes in more detail how the inherent safety of the system can be improved and how topics like drive train design and the structural design can be incorporated. The output of the post-processing is

the selected and optionally modified kinematic structure as well as a first arrangement of the robot's components for the referring robot object.

7.1 Evaluation and selection of robot kinematics

As described before, the input data of the post-processing is the output data of the solver and that is processed in the next step. This involves the evaluation of the solver results and the selection of a suitable kinematics or several of them to be considered for further processing. Depending on the task defined within the pre-processor and the additional boundary conditions, there can be a variety of feasible solutions. To enable the users to evaluate and compare kinematics more efficiently, the kinematic schemes and robot objects were automatically pre-sorted within the solver according to the minimum link length sums. Figure 7.2 shows an exemplary overview of the suitable kinematic schemes with the numbers of suitable link length sets as well as the prioritization according to the minimum link lengths, ranging from $1.8\ m$ as the best solution up to $2.2\ m$.

list with successful objects

scheme	link length set nr
$R_zR_xR_yR_xR_y$	74,75,127,154,181,182
$R_zR_xR_yR_zR_y$	75,155,181,209
$R_zR_yR_xR_yR_y$	47,65,76,100,118,136
$R_zR_yR_yR_xR_y$	154
$R_zR_yR_yR_zR_y$	152
$R_zR_yR_zR_xR_y$	172,178,190,196,202

pre-sorted list wrt link length

scheme	nr	link length sum
$R_zR_yR_xR_yR_y$	47	$1.8\ m$
$R_zR_yR_zR_xR_y$	172	$1.8\ m$
$R_zR_xR_yR_xR_y$	74	$2.0\ m$
$R_zR_xR_yR_zR_y$	118	$2.0\ m$
$R_zR_yR_yR_xR_y$	154	$2.2\ m$
$R_zR_yR_yR_zR_y$	152	$2.2\ m$

Figure 7.2 Output of the solver for further processing by the user

The overview of possible solutions in a table is not helpful to select the suitable kinematics and optimizing the robot only considering the link lengths might not be the universal approach for optimization. That's why all kinematics can be evaluated visually by the user with the help of the tool

developed within this work. Starting with the first kinematic scheme from the pre-sorted list from Figure 7.2, the referring robot objects can be visualized. To get a better understanding of how the respective robot object reaches the given target positions, all corresponding configurations with which the structures fulfill the task within the given environment are displayed in a first step. This can also be done for all other successful link length sets of the corresponding kinematic scheme. An example is shown in Figure 7.3 for the four best link length sets with respect to the minimum length for the $R_z R_y R_x R_y R_y$ scheme, which was the first structure from the pre-sorted list in Figure 7.2.

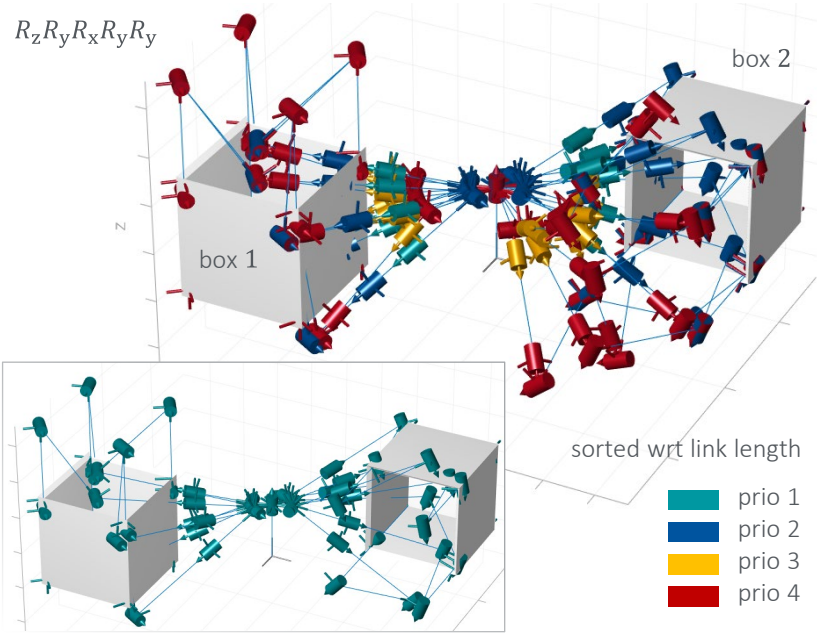


Figure 7.3 Prioritized solutions for one robot structure

This visualization gives a first impression of the effect of the different link lengths and it can already be recognized by the user whether one of the robot objects is particularly well suited. The different joint colors show the prioritized robot objects, that differ in their link length sets. To further evaluate the objects, they can be animated, and the robot object will move

to all desired target positions one after another, considering the optional orientation conditions. Figure 7.4 shows that the robot, which is visualized here only via the joints and connecting lines, can collide with the environment. This can often be avoided by changing its configuration. Depending on the degree of freedom as well as the boundary conditions, there may be one, several or an infinite number of ways in which the robot can reach a target position. The configurations mentioned here differ in the way how the kinematics reach the position by means of different joint parameters. The number of possibilities corresponds to the number of possible solutions of the inverse kinematics. During the simulation, the user has the possibility to reconfigure the kinematics so that the target positions can be reached without collision and to get an idea on how the different solutions may look like. Exemplary reconfigurations to avoid collisions for the first position $\boldsymbol{p_1}$ and position $\boldsymbol{p_9}$ are visualized in Figure 7.4.

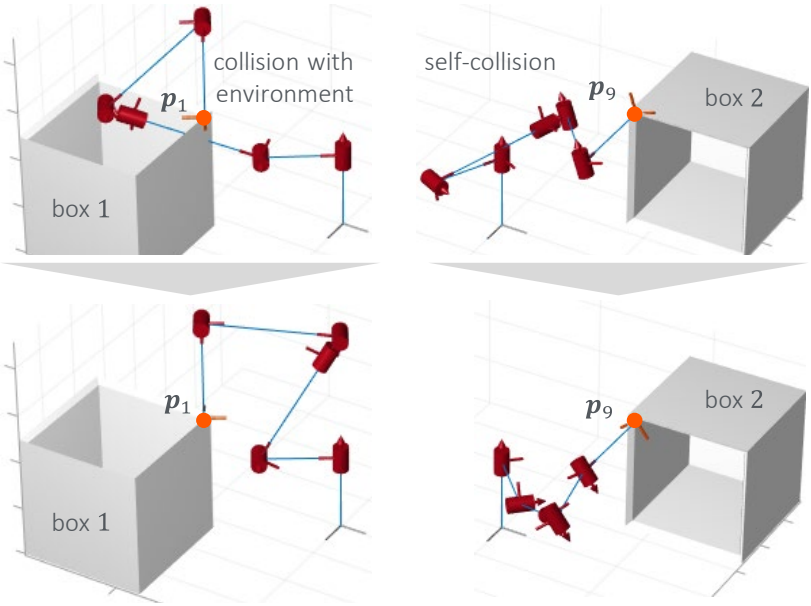


Figure 7.4 Different configurations to reach the target positions

Mathematically, this is achieved by solving the inverse kinematics again for the desired position. Since the solution obtained by using the numerical SNS

algorithm significantly depends on the initial joint values, random initial joint parameters are chosen for a reconfiguration. Tests have shown that after two to three iterations, an appropriate configuration is generated, if there is any. The possibility to reconfigure the kinematics provides the user a simple and intuitive way to view the different robot objects and evaluate criteria such as how the desired positions can be reached without collision with the environment or self-collisions and which ratio of the link lengths seems suitable. The procedure for evaluation described above can be applied to all desired kinematic schemes and the corresponding robotic objects. For additional evaluation and interactively moving of the robot object, a simple graphical user interface visualized in Figure 7.5 is provided to the user. With this tool, any kinematics can be generated and moved interactively within the desired environment. For his purpose some functions of MATLAB's robotic system toolbox [Mat20] are used, for instance to set up the rigid body trees of the robot objects. The link lengths can be adjusted interactively, and the robot base position can be changed. Animating how the robot moves to desired target positions respecting the optional orientational constraints shows the user their influence on the robot configuration. Also, other criteria such as the workspace can be evaluated by visualization of various views and the influence of the joint limits is also considered. For workspace analysis, the user can specify upper and lower bounds for each joint and evaluate whether the desired movability and reachability is ensured even with these restrictions. Furthermore, the joint limits are also respected for the workspace visualization, so that the influence is directly shown and can be interpreted by the user. This allows the desired structures to be further investigated, prioritized, and modified. In addition to the manual modification of the link length, the user can also execute a script that automatically considers a displacement of the joints in the y and z directions when setting up the robot object. In this way, a translational part in the y and z direction is also incorporated in the transformation between the different links that are represented as rigid bodies in a rigid body tree. In analogy to the discretization of the link length in the x direction, the user can specify the lower and upper limit as well as the step size for the discretization of the translational part in the y and z direction. The evaluation of the solutions can then also be carried out using the support and tools described in this chapter.

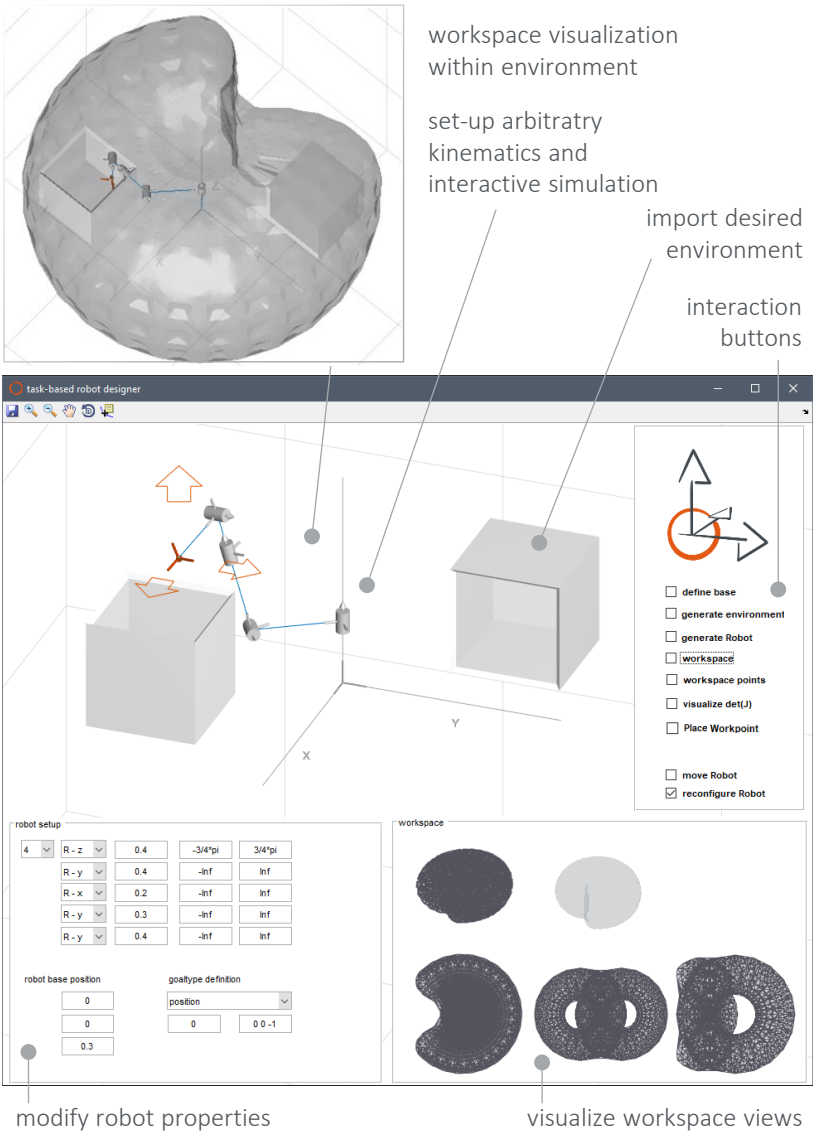


Figure 7.5 GUI to modify and evaluate robot objects for the desired task

Since the final transformation between the robot links can still change depending on the design of the hardware components, the presented GUI is

also to be used for the further development process and changes of the kinematics can be reviewed directly, so that the individual steps can be iterated to generate a suitable solution at the end.
In this way, a robot object or several of them can be selected and modified if necessary, and on this basis the further design of the robot can be realized. Of course, the physical effects must be considered by the user when selecting the robot object. For example, it should be noted that depending on the axis arrangement the necessary torque for a required process force can vary depending on whether the own weight of the robot components is carried by the bearings or must also be held by the motor. An exemplary explanation of the required actuation torques for motor selection is given in Chapter 8.5 and this impact should be in the user's mind during the selection process. After a suitable robot object has been selected based on the support provided by the tool and the user's estimation, the further design can be carried out as described in the following chapter.
Of course, there is a variety of other characteristics for comparison such as the various condition numbers or diverse indices such as those described in Chapter 2.1.2 and Chapter 4 and listed for instance in the literature [PS15b]. These characteristics can be integrated into the tool in future work. However, this is not necessary at this point, as the pragmatic approach already presented provides profitable support for the user and the evaluation at this level is sufficient for the given use cases.

7.2 Possibilities to improve the inherent safety of the system

For the initial shaping and design of the robotic system, it is necessary to consider the requirements resulting from the application. These include for instance the maximum dimensions of the interference contour as well as requirements on the inherent safety of the system, which is especially important when humans and robots are working in cooperation. This chapter describes how the robotic system can be designed to be safe not only by using sensors, but also by enhancing the mechanical design for inherent safety. Inherent safety is the elimination of a hazard by the design of the system or the reduction of the risk resulting from the hazards.

It is an important element of risk reduction as the risk of injury is decreased in advance, unlike when safety is provided by means of protection measures. The inherent mechanical safety can be positively influenced to a large extent by geometric factors and physical aspects. [DIN12100]
For the development of robotic systems, the following aspects mentioned in [DIN12100] are adapted to a robotic application scenario since they can have a significant impact on the inherent safety design and thus should be considered. First, the geometric factors are discussed, followed by an overview of the physical aspects.

Workspace visibility and arrangement

If the user can see and estimate the robot's movements properly, the system's behavior is predictable, and the risk of the user being surprised by the robot's motion is reduced. In addition, a risk probability can be influenced by the arrangement of the robot's workspace and the working area of the human. Depending on whether the human and the robot are working at the same time in the same place, with or without physical contact, the necessary safety can be achieved by means of various protective measures and must be adapted according to the form of co-work between human and robot accordingly. If the main working areas of humans and robots do not overlap or overlap only slightly, this has a positive effect on safety and the efficiency of the process as the robot is then allowed to move faster.

Design and arrangement of mechanical components

Minimum or maximum distances between moving components can reduce or even avoid hazards due to crushing or shearing, as compliance with the guideline values can prevent body parts such as the hand or an arm from entering the corresponding clearance and being injured. [ISO13857] gives an overview of safety distances to prevent hazard zones being reached by upper and lower limbs and [ISO13854] defines minimum gaps to avoid crushing of parts of the human body. These aspects are essential in the field of robot design, especially regarding the arrangement and distances of the individual robot links to one another, as these can be particularly dangerous in terms of crushing. Figure 7.6 shows an extract of the minimum or corresponding maximum distances to avoid the risks of crushing a hand and finger.

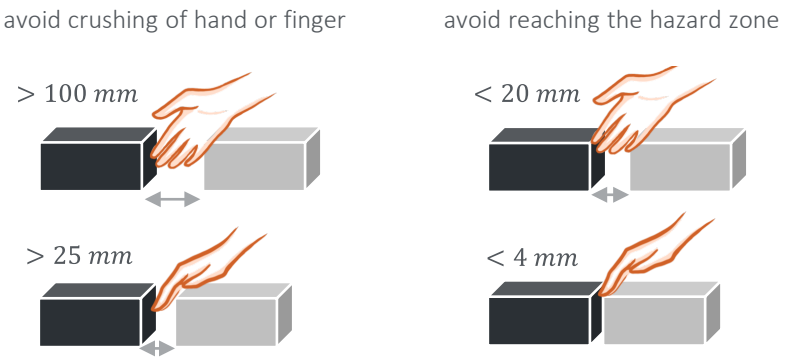


Figure 7.6 Minimum and maximum gap sizes for an inherent safe design

The left part of the figure shows how the minimum distances must be designed so that the body part is not crushed, and the right part shows an example of the maximum gap dimensions that are permissible so that the area behind which might be hazardous for instance due to moving components, cannot be reached with the corresponding body part. The designer can use these values as a guideline when designing gaps between two moving components or covers, such as the gap in the cover for linear guides, or when adjusting the distance between robot links. This enables a reduction of the injury risk for the respective body parts just by the arrangement of the components and the distance between them. The shaping of the structural design itself, as well as the consideration of the gap dimensions, may cause the kinematic dimensions of the system to be changed. When this is the case, the user can refer to the tools described in Chapter 7.1 to evaluate whether the modified kinematics are still suitable to perform the required task, considering the environmental conditions.

Avoidance of sharp edges and corners:

Generally, sharp edges and angles, rough surfaces and protruding parts should be avoided, as this can significantly reduce the risk of injury in the event of a collision with such geometries. As a reference, the designer can orientate on a minimum radius of $5\ mm$ [BGI11]. If the designer takes these

requirements directly into account in the mechanical design, the risk of injury can be reduced with little effort.

In addition to the geometric means of improving inherent safety, physical aspects can also make a significant positive contribution to hazard reduction and are thus addressed in the following.

Limitation of the actuating power

To reduce the potential hazard, it is advisable to define the maximum process forces and torques so that they meet the requirements of the given task but are not significantly over-dimensioned. A suitable description of the requirements for the forces and torques to be applied not only reduces the risk of injury, but also results in other positive properties for the design, for example that the drive components are not bigger than actually required, which leads to an improvement in the design space and the effective masses.

Targeted use of compliant components

Another way of reducing risks is the targeted use of compliant components. This can be done within the drive components, as for instance described in [HSV09; Ros16], so that in the event of a collision, a certain amount of the collision energy is absorbed or dissipated by these parts. Another measure that is easy to implement is the use of flexible covers to absorb the impact energy.

Limitation of the kinetic energy

One of the most effective measures to increase inherent safety is to limit the kinetic energy of the system. By principle, this can be achieved by limiting two parameters, namely the velocity or the moving masses of the system. According to [TS15066], the maximum permissible kinetic energy $E_{\mathrm{kin,max}}$ can be determined depending on which part of the body can be injured, with the most critical part of the body being the human head. With this value and either the specification of the velocity or the masses, the allowable moving mass or the allowable velocity respectively, can be determined via reformulation of equation (7.1), where m_{h} is the effective mass of the human body region and can be taken from [TS15066]. The effective mass of the

robot system is composed of the total mass of the moving parts of the robot m_r and the effective payload m_l, including tool and workpiece.

$$E_\mathrm{kin} = 0.5 \left(\frac{1}{m_\mathrm{h}} + \frac{1}{0.5 m_\mathrm{r} + m_\mathrm{l}} \right)^{-1} {v_\mathrm{rel}}^2 \tag{7.1}$$

In general, the permissible velocity should not be limited excessively, as this reduces the efficiency and cycle time of the system. The aim is therefore to keep the moving masses as low as possible so that a safe process can be guaranteed that meets the requirements in terms of performance.
To reduce the effective moving robot mass, lightweight design plays a crucial role, and it is worth looking at those components that contribute the most to the weight. These components are the structural parts as well as the drive train, which implies that the weight of the robot can be significantly reduced by lightweight design of the structural parts or modification of the drive train. One or two joints are attached to each of the structural parts of a serial robot, and they accordingly represent the supporting structure between the axes. The design of these components and their connection to the joint units therefore have a major influence on the mechanical properties of the system, such as stiffness and weight. In general, the structural parts can be produced using various manufacturing processes. They can for instance be milled from solid material and accordingly designed as castings in series production, or semi-finished products of various shapes can be used, such as rectangular tubes or circular tubes. In this work, the selection is reduced to the use of semi-finished products, since they are suitable for initial validation of the calculated kinematics and for rapid assembly and low manufacturing costs. In this context, circular hollow sections (CHS) and rectangular hollow sections (RHS) are considered and, according to the requirements, one or the other profile may be more suitable.
CHS have good mechanical properties due to the uniform distribution of the cross-section around the center of gravity and offer high-performance behavior, especially in torsional or combined load cases. RHS, on the other hand, are particularly resistant to bending and the planar surfaces offer better integration and mounting of other components. Since the robot structural parts are usually exposed to varying load cases, the use of CHS can

reduce the weight by an equivalent mechanical performance compared to RHS and is therefore recommended, if the design is suitable. Depending on the requirements, the user can therefore select the cross-section and dimension it according to the strength theory so that the component will have sufficient stiffness.

Another option for reduction of the effective masses is to relocate the drive components to the base of the robot or at least closer to the base. Drive components include motor, encoder, joint torque sensor, bearing, gearbox and optionally additional transmission elements as visualized in Figure 7.7, and they can contribute a significant portion on the robot mass, depending on the design.

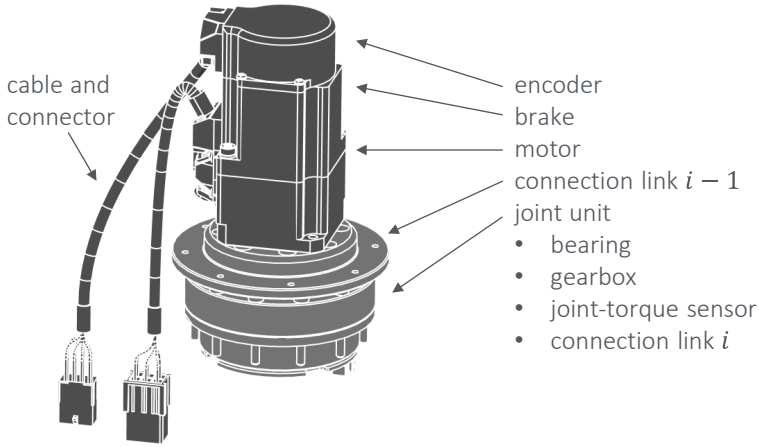


Figure 7.7 Exemplary drive components

Normally, the drive components of the robot's regional structure are mounted in link $i-1$ which is located ahead of the axis i to be driven. As a result, the weight of the drive components is distributed over the entire regional structure and the masses are moved accordingly. To reduce the effective mass of the system, the drive components can be relocated to the robot base and the necessary torque and angular motion can be transmitted to the corresponding axis by means of transmission elements. The

transmission can for instance take place via traction drives such as the timing belt drive as investigated for instance in [AZE16; YSK13; ZRK04].
If the drive components are relocated, this affects not only the effective mass, but also other mechanical properties and Table 17 shows a selection of them as well as the influence on the properties when the drive components are relocated to the base.

Table 17 Influence of relocating the drive components to the base

Property / characteristic	Influence of relocating	Impact on inherent safety
Effective mass	⬇	Significantly positive
Required torque	⇩	Positive
End-effector velocity	⇩	No direct impact
Design space close to the end-effector	⇩	Dependent on application
Drive train stiffness	⇩	No direct impact
Complexity	⬆	No direct impact

The direction of the arrow indicates whether it is an increase or decrease of the value of the parameter under consideration, and the colors indicate whether this has a positive (green), negative (red) or neutral (gray) effect. The filled arrows represent that this influence is an essential aspect with respect to an application in the field of service robotic. Further, the right column of the table describes the impact of the changes in terms of inherent safety. If for example the motor and gearbox are moved to the base and thus belt drives are used as transmission elements, the stiffness of the drive train is reduced. This is generally a negative mechanical effect on the robot's performance, but a loss of stiffness could be accepted in the field of service robotics since the advantage of the reduction of the effective mass

predominates. However, the increase in complexity resulting from the additional components and their integration plays a decisive role for the general design, so that depending on the application and requirements, it must be evaluated whether it makes sense to relocate the drive components or not, but the user should consider this option to increase the inherent safety of the system. In Chapter 8.5 these considerations and the resulting influence on the robot design are carried out by means of a practical example within the research project REFILLS.

With the robot structure and the appropriate dimensions as described in Chapter 7.1 selected, as well as the approaches suggested within this chapter regarding link structure and component arrangement to increase the inherent safety, the user can proceed with the rough design and continue with a standardized development process. Once the rough design has been completed, the detailed design and concretization can take place. After the components have been ordered and manufactured, the system can be assembled, integrated, and evaluated. If necessary, the development steps must be repeated iteratively to optimize the system until all the necessary requirements are fulfilled as desired.
In summary, the last chapters presented the methods as well as the implementation on how to support a user in the development of a novel serial robot in a semi-automated way and a collection of relevant steps within the presented approach is shown in Figure 7.8. Based on the given boundary conditions, the task to be automated as well as already known requirements can be defined within the pre-processing (Chapter 5). With usage of this data, the type and dimensional synthesis is carried out within the solver (Chapter 6) so that suitable kinematics are determined, and the corresponding link lengths are calculated. The generated and pre-selected robot objects are pre-sorted based on the total link lengths and can be evaluated by the user with the support of an interactive GUI during post-processing (Chapter 7), so that one or more suitable robot objects can be modified if necessary and are identified for the further development process. In addition, some advice has been given which may improve the inherent safety of the system and these aspects should be considered by the user right from the beginning of the design phase. In this way, a first rough arrangement

and pre-dimensioning can be achieved. From this stage, the normally used development process can be continued and the robot system can be further designed and finalized.

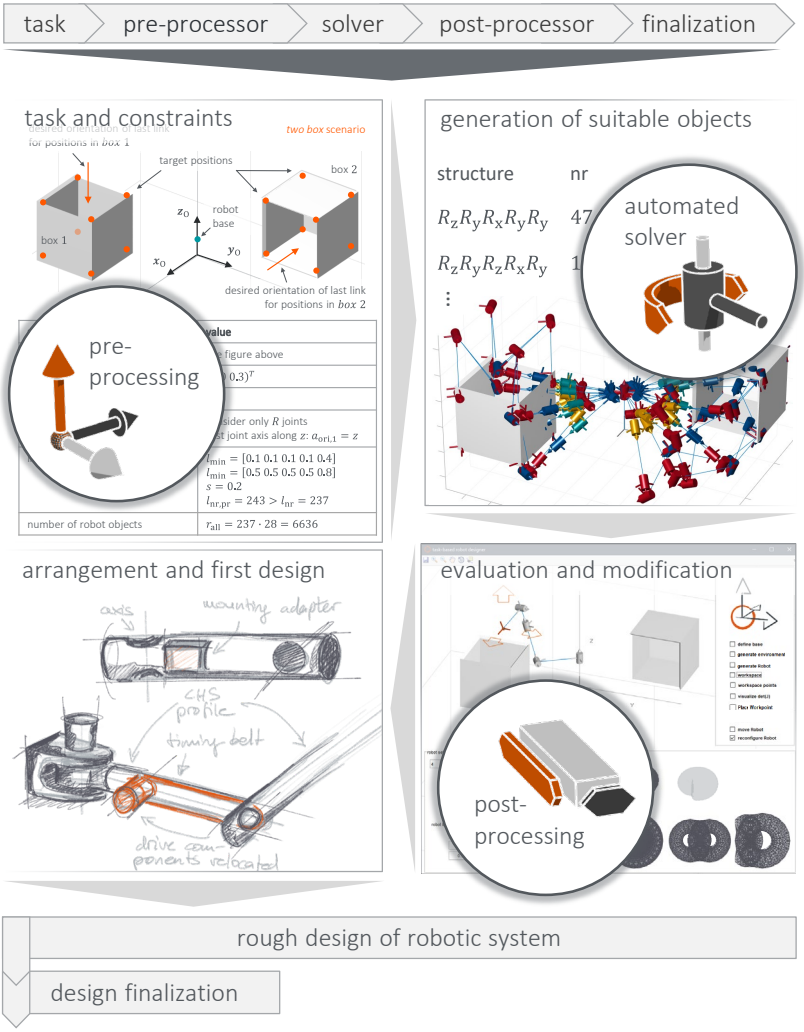


Figure 7.8 Exemplary process steps of the presented approach

With the support of the presented approach, the developer can drastically save time and is also visually supported by the tool during the design, so that a suitable kinematic structure can be found more easily, and the robotic system can be created based on these results. The automated computation within the solver can take up to several hours, depending on the given boundary conditions, but usually can be done overnight in less than twelve hours, so that after starting the computation process the results are ready for further elaboration in the next morning.

To evaluate the presented approach, the next Chapter 8 shows the exemplary development of a robotic system within the REFILLS project in the field of professional service robotics using the provided methods and tools, followed by a more detailed look on the design.

8 Utilization of the proposed design method for the REFILLS project

This chapter deals with how the proposed method for robot design supports the user with the development of robotics enabling fully integrated logistics lines for supermarkets (REFILLS) in the field of service robotics [Sic21]. The project was funded by the European Union under horizon 2020 with the aim of improving intralogistics at supermarkets and drugstores in the future and is outlined in the next Chapter 8.1. In Chapter 8.2, the boundary conditions to be considered in the development of the presented robot are discussed. The robotic system will be developed with the support by the proposed approach for semi-automated robot design. In this way, the pre-processor, the solver, and the post-processor are run through based on a real development project.
To do so, in Chapter 8.3 the previously identified requirements are used as input for the pre-processor to define the handling task and constraints. This data is prepared so that within the solver in Chapter 8.4 the appropriate kinematics are identified, and the associated link length sets are determined. Within the framework of post-processing the selection of the robot object to be further elaborated is supported, as described in Chapter 8.5. This concludes the last partially automated step of the methodology and the results are then usable for the following design process. For further development, the suggestions regarding the increase of inherent safety are considered, with focus on the drive concept. The mechatronic rough and detailed design based on the interim results are described in Chapter 8.6 and the set-up of the system as well as a brief evaluation of the proposed robot are explained in Chapter 8.7. In summary, the complete development process from initial concepts to the construction of a robot prototype is discussed in the following chapters.

8.1 REFILLS project introduction

As described above, the focus of the funded research project REFILLS is to improve intralogistics of retail stores in the future. To do so, it is essential to work on the relevant topics to identify and tackle challenges to provide suitable technologies and possible solutions for future development. With

Consortio CREATE, dm-drogerie markt GmbH + Co. KG, intel corporation, KUKA Deutschland GmbH, University of Campania “Luigi Vanvitelli”, Swisslog Holding AG 2021, and University Bremen being partners within this project, a great variety of different expertise is covered. Within the scope of the project, solutions were developed for the automation or partially automation of the key process steps in the retail store for replenishing the goods. These automation processes are divided into different scenarios, which represent the contents of the process steps according to the current status. The aim is to use modular robotic systems to solve the various tasks and to be able to adapt the entire system depending on the size of the store and the requirements. [EC16]
The process steps to be optimized are listed in the following and visualized in Figure 8.1:

1. Inventory of goods so that products can be reordered as needed
2. Pre-sorting of delivered goods from pallets onto trolleys
3. Transporting the goods to be stored to the appropriate shelves
4. Replenishing the products and those that no more fit into the shelf are returned to the backroom

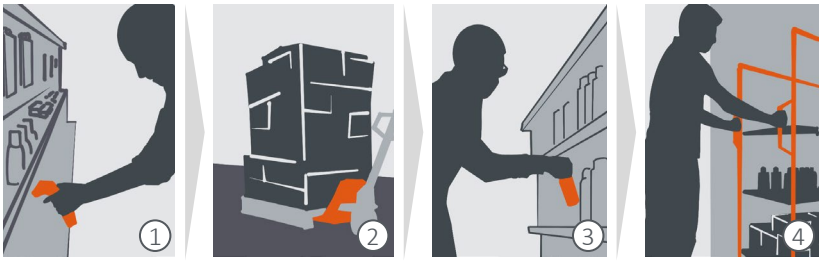


Figure 8.1 Intralogistics steps according to current process

If these process steps are designed more efficiently, the transportation of goods as well as the knowledge of products within the store can be significantly improved, since in the current set-up, it is often not known how many goods are in which location, and the transportation of remaining

product to the backroom also restricts an efficient flow of materials. To improve the process the robot modules outlined in Figure 8.2 can be used:

1. A scanning module can take over the inventory automatically
2. The delivered goods are autonomously sorted on trolleys according to predefined categories in the backroom of the store
3. With a carrier module, which provides mobility by coupling to the passive modules, the trolleys are autonomously driven to the required position in front of the shelves
4. The handling module refills the shelves with the products stored on the trolleys in collaboration with a clerk or autonomously

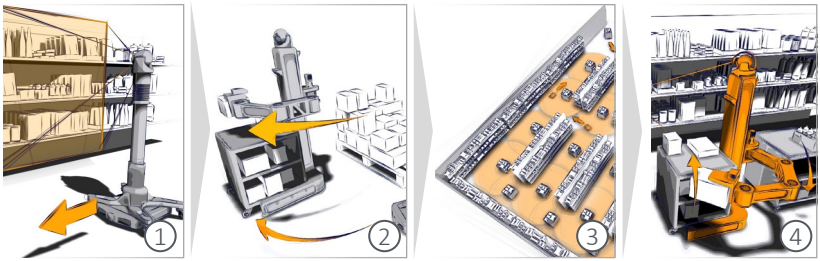


Figure 8.2 Concepts for robotic modules presented in [EC16]

Improved tracking in the store should eliminate the need to return products and in combination with a smart pre-sorting algorithm the intralogistics' efficiency could be increased drastically.
The following chapters deal with the development of the handling module, which is designed to enable the autonomous and collaborative replenishment of the goods. A conceptual drawing from the project proposal is depicted in step four of Figure 8.2. The investigated boundary conditions and requirements thus relate to the scenario of product placement and the other modules and scenarios are not elaborated further.

8.2 REFILLS – requirements for the handling module design

To develop the robotic system in a targeted manner, the already known requirements are first collected. These relate to the task to be automated

and the environment in which this is done. Table 18 shows a small excerpt of the requirements to be met by the system. They refer to different categories such as geometry, mechanics, ergonomics, and electrics to show the variety of requirements that must be considered.

Table 18 Extract of requirements for the handling module development

Requirements	Collaborative scenario	Autonomous scenario
Payload $[kg]$	10	2
Object dimensions $[mm]$	• Handling boxes $400 \times 400 \times 350$	• Variety of selected items
Design space $[mm]$	• Max. module width: 1200 • Max. module height:2000 • Robot arm dimensions: be able to handle within shelves and trolley	
Process	• Increase efficiency • Ergonomic process	• Dexterous manipulation
Hardware interfaces		• Gripper WSG50 [SCH20]
	• Same basic kinematics for both scenarios • Trolley integration • Interface to carrier module • Integration of existing units	
Voltage supply $[V]$	• $5, 12, 24, 48$	

The selection refers primarily to requirements that are important for successfully realizing the scenarios of both, the autonomous and collaborative shelf refilling. In some categories the requirements are identical and for some requirements they differ according to the two use-cases.

Collaborative shelf refilling

In the collaborative refilling process, clerks and robots work together to place products on the shelves from boxes weighing up to $10\ kg$ that are prepared on the trolley. The aim of using a robotic system is to make the process more

ergonomic for the workers because handling these heavy boxes, especially from the lower trolley levels is uncomfortable and not a task that should be performed permanently. In addition to ergonomics, efficiency plays a decisive role. Therefore, the process must not take longer than the current state of the process. Especially in this scenario, where humans and robots cooperate, the safety of the system and the safety of the cooperation is mandatory.

Autonomous shelf refilling

In the autonomous filling process, the robot works alone and replenishes the shelves with individual products, which are separated and stored within the trolley. The individual products weigh up to $2\ kg$ and the packaging can be rigid or flexible. Since the products are to be grasped by the robot, handled, and placed in different ways, there is a requirement for dexterous manipulability in this scenario and thus a minimum of $6\ dof$ of the robotic arm including a robot wrist is required by the partners. In addition, the gripper to be used is already defined as WSG50 from Schunk GmbH & Co. KG [SCH20].

Common requirements for both scenarios

In addition to the requirements specific to the scenarios, there are also requirements that apply independently to the overall system. These include for instance the maximum permissible dimensions of the module. The height results from the requirement that the system must fit through doors, and the maximum permissible width results from the corridor width of the stores in which the system is to be used. In addition, the robot arm should be designed as flat as possible especially in the area close to the end-effector, to be able to handle within the shelves and trolley levels as flexibly as possible. In summary, the handling module must be able to navigate in the surroundings of the store and handle components without colliding with the interfering contours of the environmental objects.

By combining several modules that are developed in the scope of refills, the required functionalities can be provided. For this reason, the handling module must allow integration of other modules. For example, it should be possible to integrate the trolley into the handling module in such a way that

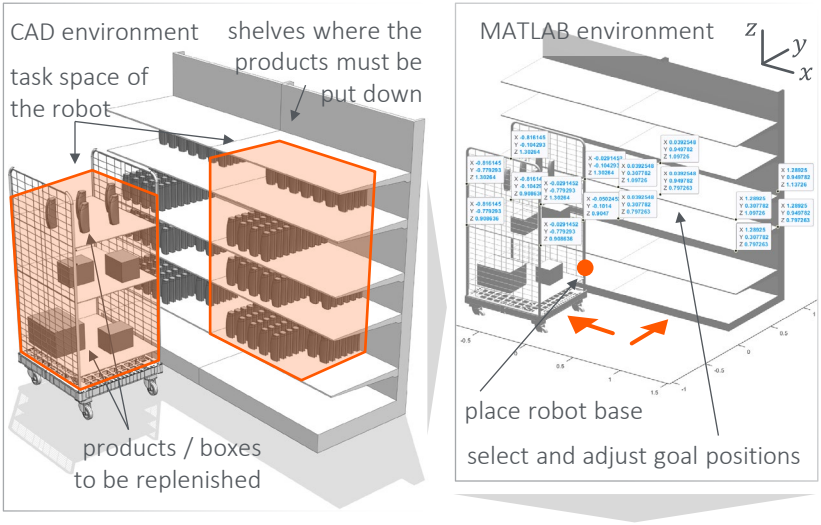
a reasonable system combination results and the robot arm can easily reach and handle the goods within the trolley. In addition, the handling module is passive in terms of mobility, but should be able to be moved by humans and can be navigated and transported autonomously through the store in combination with the carrier module. An additional requirement for the robot arm is that existing joint units should be used, if possible.
As a requirement in the field of electronics, it is given that $48\ V$ must not be exceeded as the voltage supply, since it is a system for human-robot collaboration. In addition, the voltages specified in Table 18 must be provided for the various electrical consumers of the system.

This excerpt of requirements represents the basis for deriving the specifications to be met by the system to be developed. The list of requirements is adapted and expanded throughout the entire development process and must always be observed. Having identified and explained certain requirements, pre-processing can be started as explained in the following Chapter 8.3.

8.3 REFILLS – pre-processing

The first of the partially automated process steps for the development of new robotic systems is pre-processing. As described in Chapter 5, the handling task to be automated and additional boundary conditions are defined at this stage. The working areas to be reached with the robot's regional structure are the trolley interior to manipulate the boxes or grasp the separated objects, as well as the spaces between the shelves to place the single products as visualized in Figure 8.3.
To ensure that the robot kinematics to be developed reaches the entire required working area, the corner points of the two cuboids that span the desired volume are selected as position specifications. This is done interactively by selecting the corner points of the model and the right corner points of the shelf are adjusted manually in such a way that not the entire shelf is covered, but the area as shown by the orange volumes in Figure 8.3, so that the robot arm does not become excessively large. If products should be stored in other shelf areas during the storage process, the handling

module can be moved autonomously by the carrier. Therefore, it is useful not to expand the volume within the shelf too much. The selected target positions are then stored as positions and can be passed as parameters to the robot objects and are defined by their referring coordinates $\boldsymbol{p}_i = [p_{\mathrm{x},i}, p_{\mathrm{y},i}, p_{\mathrm{z},i}]$.



task and constraints

```
dof = 5;

positions =
[-0.0291,-0.7793, 0.9086;
 -0.8161,-0.7793, 0.9086;
…
   1.0000, 0.9498, 0.7972;
   1.0000, 0.3078, 0.7972];

goalR(1,:) =  [-1 0 0];
goalR(2,:) =  [-1 0 0];
…
goalR(15,:) = [ 0 1 0];
goalR(16,:) = [ 0 1 0];

base = [0 0 0.3];
dtcp = [0 0];
```

```
%just revolute joints - 5R
CRx = t_prepareKinType(dof,'Rx');
CRy = t_prepareKinType(dof,'Ry');
CRz = t_prepareKinType(dof,'Rz');
jntLim(i,:) = [-Inf Inf];
lmin = [0.1 0.1 0.1 0.1 0.4];
lmax = [0.5 0.5 0.5 0.5 0.6];
step = 0.2;

%one prismatic joint - 1P4R
CPz = t_prepareKinType(dof,'Ry');
jntLim(i,:) = [0 l(i)];
lmin = [1.5 0.2 0.2 0.2 0.4];
lmax = [1.5 0.6 0.6 0.6 0.6];
step = 0.2;
```

Figure 8.3 REFILLS scenario pre-processing – task definition and constraints

In addition, orientational constraints are defined for all target positions to reduce the risk of collisions already during synthesis. The positions defining the volume within the trolley should be approached so that the last link of the regional structure is aligned parallel to both the sides and the trolley planes, thus pointing in the negative direction of the x unit vector of the world coordinate frame. To reduce the risk of collision when placing the products on the shelf, it is specified for the points spanning the corresponding volume, that the last link must point in the direction of the y unit vector of the world frame. The orientational constraints are visualized in Figure 8.3 by the orange arrows and indicate the alignment of the last robot's link. In addition, the robot base and the offset of the end-effector frame or TCP in y and z direction are defined. The structures to be considered are the kinematic schemes defined in Chapter 6.1, each with $5\,dof$ and the orientation of the first axis in x, y and z direction is permitted. The possible combinations are stored in the matrices $\boldsymbol{C}_{\mathrm{Rx}}, \boldsymbol{C}_{\mathrm{Ry}}$ and $\boldsymbol{C}_{\mathrm{Rz}}$, respectively. The upper and lower bounds for the link lengths of the individual links and the step size for discretization are given in Figure 8.3. For the last link, the interval is selected in a higher range, since this link is to be moved into the working areas without collision. In addition to kinematic schemes consisting only of revolute joints, structures are also considered in which the first joint is a prismatic joint with axis in z direction, so that P_{z} is also considered as first joint and the resulting kinematics are stored in $\boldsymbol{C}_{\mathrm{Pz}}$. For the first design, only the length of $1.5\,m$ is considered here as the link length range for the first link to reduce the number of calculations. The link length referring to the prismatic joint is only relevant for the stroke length and can be selected here according to the environmental conditions and adjusted if necessary.

Based on the values in Table 13 and Table 16, there are a total of $r_{\mathrm{nr}} = 28 + 28 + 28 + 12 + 11 + 11 = 96$ kinematic schemes to be considered. For the $5R$ chains the $l_{\mathrm{nr,5R,pr}} = 162$ prepared link length sets are reduced to $l_{\mathrm{nr,5R}} = 85$ by eliminating the obviously unsuitable sets as described in Chapter 5.2. For the $1P4R$ scheme there are $l_{\mathrm{nr,1P4R}} = 54$ link length sets to be considered. Thus, based on the specifications of the boundary conditions and following equation (6.3), in sum a total of $r_{\mathrm{all}} = 3 \cdot 28 \cdot 85 + 34 \cdot 54 = 8976$ robot objects are obtained, which are

further considered and automatically processed within the scope of the solver as presented in the following Chapter 8.4.

8.4 REFILLS – solver

Input data for the solver is the data prepared by the pre-processor. Based on the mechanical boundary conditions it is known which kinematics are to be considered and the permissible link length ranges are known. From these data all robot objects to be considered are generated automatically and are investigated in the following for their suitability for the required handling task. The entire process within the solver is automated, so no user input is required, and after calculation the robot objects suitable for taking the products out of the trolley and putting them into the shelves are provided.
For this purpose, the inverse kinematics for each of the 8976 robot objects and all of the 16 target positions are calculated as described in Chapter 6.2. The maximum number of superior calls to solve the inverse kinematics according to equation (6.4) is $k_{\max} = 8976 \cdot 16 = 143616$, but can be reduced to a total of 11962 for all kinematic types together, following the procedure presented in Figure 6.5, so that only previously successful kinematics will be considered for further calculations to reduce computation time. This strategy reduces the total number of calculations by $91.67\ \%$ and the total calculation time is $163.59\ min$ for all kinematic structures combined, on a notebook with Intel® Core™ i5-6300HQ CPU @ 2.30 GHz and 16 GB RAM. The corresponding overview with the data for the separate kinematic schemes which differ by their first joint is given in Table 19.
A total of 15 kinematic structures are identified as potentially suitable for the required task under the given boundary conditions, which represents $12.71\ \%$ of all kinematic chains under consideration. The robot objects with minimum link lengths but still able to reach all preset positions have a total length of $2\ m$ when adding all the links together. Since the stroke of the prismatic joint with $1.5\ m$ is included in the total length of the P_z4R schemes, the total length is naturally much greater, but this should not be viewed negatively here. For each kinematic structure, those robot objects that can reach all target positions with their link length sets under the given boundary conditions are stored for further processing.

Table 19 Parameters and results generated within the solver

	R_x4R	R_y4R	R_z4R	P_z4R
no. kinematic structures	28	28	28	34
no. robot objects r_{all}	2380	2380	2380	1836
IK calls / total number	2906/38080 8 %	2845/38080 7 %	3780/38080 10 %	2431/29376 8 %
computation time $[min]$	34.76	50.92	47.79	30.12
no. successful structures	2	3	4	6
min. total link length $[m]$	2.2	2.0	2.0	3.1
one of the successful robot objects visualized	$R_xR_zR_xR_xR_z$	$R_yR_xR_yR_xR_y$	$R_zR_yR_zR_zR_z$	$P_zR_zR_zR_zR_z$

The objects are pre-sorted with respect to their link length sum since the robot arm should only be as large as necessary to reach all specified positions. The identified pre-sorted robot objects represent the output of the solver and can be evaluated, modified, and selected for further design by the user in the scope of post-processing as described in the following Chapter 8.5.

8.5 REFILLS – post-processing

During post-processing, the output data of the solver, which are the pre-sorted potentially suitable robot objects, is further processed. An over dimensioned design should be avoided, especially in consideration of collision avoidance as well as robot mass and torque to be applied by the motors to drive the robot's axes. Although this pre-sorting can help the user to select the robot objects, it is only a first criterion and the final selection is made by the user under consideration of further aspects.

The 15 structures identified as potentially suitable within the solver in Chapter 8.4 are therefore visualized to the user. By moving to the target positions and the possibility to let the robot objects move to arbitrary positions within the given environment respecting optional orientational constraints, a first impression of the structures can be formed. This includes for example whether the specified positions can be reached without a collision with the environment. Table 20 gives an overview of the potentially suitable kinematic chains and results of a first observation. The first part of the number column is the sequential numbering, and the second part indicates the robot object number of the corresponding robot scheme as in Table 19, which differ by the specification of the first joint. Some kinematics can be excluded directly by visual inspection within the provided user interface. In the category with a prismatic joint, only the variant in which all joint axes are parallel and pointing in the z direction is useful since the R_{x} or R_{y} joint in the other structures offer no additional value for the kinematics and this redundant degree of freedom cannot be used for collision avoidance within the given environment with the parallel trolley layers and shelves. Other potential problems may arise due to the risk of collision with the ground or self-collision due to the robot's link length ratios.

Table 20 First evaluation of robot objects based on visualization and animation

no	structure	useful	comment / first observation
01/11	$R_\mathrm{x}R_\mathrm{z}R_\mathrm{z}R_\mathrm{x}R_\mathrm{z}$	✓	• min. link length of $l_5 = d_\mathrm{T}$ • danger of collision with floor cause of l_2
02/14	$R_\mathrm{x}R_\mathrm{z}R_\mathrm{x}R_\mathrm{z}R_\mathrm{z}$	✓	• min. link length of $l_5 = d_\mathrm{T}$ • reaches shelf barely without self collision
03/05	$R_\mathrm{y}R_\mathrm{z}R_\mathrm{x}R_\mathrm{y}R_\mathrm{y}$	(✓)	• min. link length of $l_5 = d_\mathrm{T}$ • danger of self collision when entering shelf
04/07	$R_\mathrm{y}R_\mathrm{x}R_\mathrm{z}R_\mathrm{x}R_\mathrm{y}$	✓	• min. link length of $l_5 = d_\mathrm{T}$ • danger of collision with floor cause of l_2
05/15	$R_\mathrm{y}R_\mathrm{x}R_\mathrm{y}R_\mathrm{x}R_\mathrm{y}$	✓	• min. link length of $l_5 = d_\mathrm{T}$ • danger of collision with floor cause of l_2
06/06	$R_\mathrm{z}R_\mathrm{y}R_\mathrm{x}R_\mathrm{y}R_\mathrm{y}$	✓	• min. link length of $l_5 = d_\mathrm{T}$ • danger of collision with floor cause of l_2
07/07	$R_\mathrm{z}R_\mathrm{y}R_\mathrm{x}R_\mathrm{y}R_\mathrm{z}$	(✓)	• min. link length of $l_5 = d_\mathrm{T}$ • danger of collision with trolley
08/08	$R_\mathrm{z}R_\mathrm{y}R_\mathrm{y}R_\mathrm{x}R_\mathrm{y}$	(✓)	• min. link length of $l_5 = d_\mathrm{T}$ • danger of collisions when entering shelf
09/17	$R_\mathrm{z}R_\mathrm{y}R_\mathrm{z}R_\mathrm{z}R_\mathrm{z}$	✓	• min. link length of $l_5 = d_\mathrm{T}$ • danger of collisions when entering shelf
10/07	$P_\mathrm{z}R_\mathrm{z}R_\mathrm{y}R_\mathrm{z}R_\mathrm{z}$	X	• R_y joint has no useful impact
11/10	$P_\mathrm{z}R_\mathrm{z}R_\mathrm{z}R_\mathrm{y}R_\mathrm{z}$	X	• R_y joint has no useful impact
12/11	$P_\mathrm{z}R_\mathrm{z}R_\mathrm{z}R_\mathrm{z}R_\mathrm{y}$	X	• R_y joint has no useful impact
13/12	$P_\mathrm{z}R_\mathrm{z}R_\mathrm{z}R_\mathrm{z}R_\mathrm{z}$	✓	• danger of self collision when entering shelf
14/27	$P_\mathrm{z}R_\mathrm{x}R_\mathrm{z}R_\mathrm{z}R_\mathrm{z}$	X	• R_x joint has no useful impact
15/32	$P_\mathrm{z}R_\mathrm{x}R_\mathrm{z}R_\mathrm{z}R_\mathrm{z}$	X	• R_x joint has no useful impact

For this reason, it will be further examined whether there are well-suited link length sets with which the desired task space can be achieved, respecting the orientational boundary conditions and without a collision by utilization of the tool. In this way, the user can successively examine the different structures, move the end-effector to the desired regions, reconfigure the robot, and change the parameters such as robot base position or link lengths.

The exemplary result of further evaluation of the successive object numbers $02, 04, 09$ and 13 is shown in Figure 8.4 and their schemes are depicted in the appendix (Table 24 and Table 25). To compare the different structures, four evaluation criteria are selected.

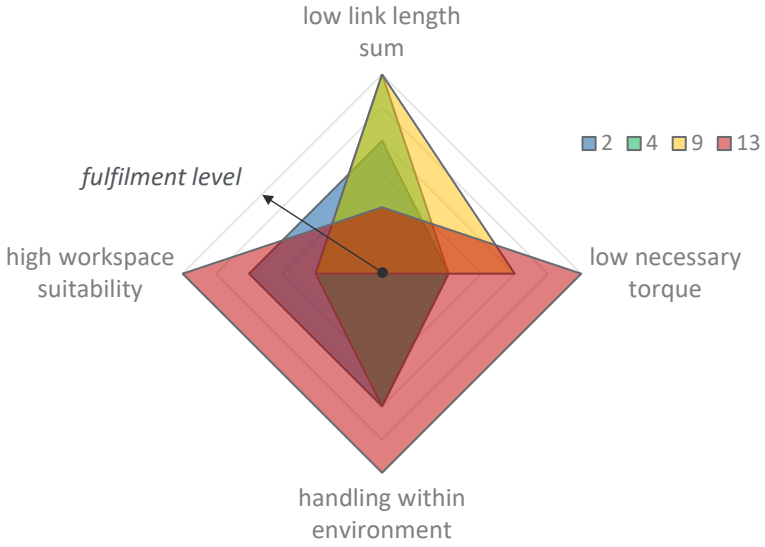


Figure 8.4 Evaluation and comparison of four pre-selected structures

The larger the area covered by the rating of each criterion, the more suitable is the robot object. It is generally desirable to keep the sum of link lengths as low as possible, as this makes the entire system lighter and reduces the risk of collisions caused by unnecessarily large interfering contours. The alignment of the joint axes determines the necessary torque that must be applied to be able to hold the robot's own weight as well as the payload. With axes permanently pointing in the z direction, this weight can be carried by the bearings, which reduces the necessary torque, and this in turn influences the motor size and thus the motor weight positively. The handling within the environment evaluation is a measure of how well and flexibly the end-effector of the kinematics can be moved to the target positions without colliding with the environment and can be investigated by the developer with the help of the presented user interface. Since in the REFILLS handling task

the objects are handled between the trolley levels and shelves, there is a need for proper horizontal movability. The last criterion depicted in Figure 8.4, exploitation of the workspace, indicates how good the ratio of the task space to be reached is in relation to the actual workspace of the robot.
The evaluation of the different kinematics represented by the colored areas in Figure 8.4 shows that based on the four selected criteria, structure 13, which is a $P_z R_z R_z R_z R_z$ scheme, is the most suitable. Therefore, the robot is designed based on this kinematic structure as described in the following Chapter 8.6.

8.6 REFILLS – mechanical design finalization

Since the presented tool was developed after the design freeze of the kinematic structure and the rough dimensioning within the REFILLS project, a first design of the handling module already has been done based on experiences and know how in the field of robot kinematics. The chosen kinematics also was a $P_z R_z R_z R_z R_z$ scheme, thus being the same result as presented in the last Chapter 8.5. The concept development took a lot of time, effort, and required a profound technical knowledge. The fact that the developed tool also generates exactly this solution as a suitable solution and that it is also identified as particularly suitable by support of the GUI and evaluation based on the selected criteria shows that the results generated by this tool can certainly keep pace with the conventional development of kinematics, and however, requires significantly less time. The required time results from the time for the partially automated, application-dependent pre-processing, the calculation time of the solver, which in this example is $2.73\ h$ as mentioned in Chapter 8.4, and the time required for post-processing with the help of the developed GUI, which is also application-specific. However, with the help of the tool an overview is provided for the user so that the selection process is supported. All in all, the time required to develop suitable kinematics including the identification of suitable link lengths using the presented tool totals a few days, compared to several weeks for purely manual development, using the standard CAD tools.
Due to the preceding milestone of the design freeze within the project, the link lengths of the robot structure were already fixed to the values shown in

Table 21. A particularly well-suited link length set based on the calculations within the tool and evaluation of the user being supported by the GUI is also given in Table 21. Based on the determined kinematics and the corresponding link-length set, the design can be done in the next step. Thus, as visualized in Figure 3.1, the kinematics development can be integrated into an already used development process like for instance [VDI2221-1], so that the developer is supported by the partial automation of the kinematics development and can then resume the implemented standard process, so that the design of the system can be carried out in a structured and systematic manner. Some essential aspects during the further development of the REFILLS robot arm are described in the following.

Table 21 Pre-determined link length and generated length by using the tool

link no.	link length generated manually $[mm]$	link length generated with help of the presented tool $[mm]$
1	430	600
2	430	400
3	280	600
4	425	400

Following the requirements from Table 18, in the collaborative scenario boxes with $10\,kg$ must be handled, and in the autonomous scenario individual products with $2\,kg$ are to be manipulated, both with the same basic robot kinematics. To be able to manipulate the $10\,kg$ boxes in space and still not over-dimension the robot for the autonomous storage, a special handling strategy was developed. The handling strategy has been patented in [ZR20] and is outlined in the following. To handle the $10\,kg$ boxes, the main task of the collaborative scenario is divided into two sub-tasks:

1. Planar manipulation by pushing of the handling objects by the robot arm kinematics
2. Vertical manipulation of the handling objects via actuated trays that can move along the z axis

In the collaborative scenario it is not necessary that the robot can orientate the objects, thus the principal axes are sufficient to position the boxes and no robot wrist is needed. The robot arm pushes the appropriate boxes out of the trolley on one of the actuated trays as visualized in Figure 8.5.

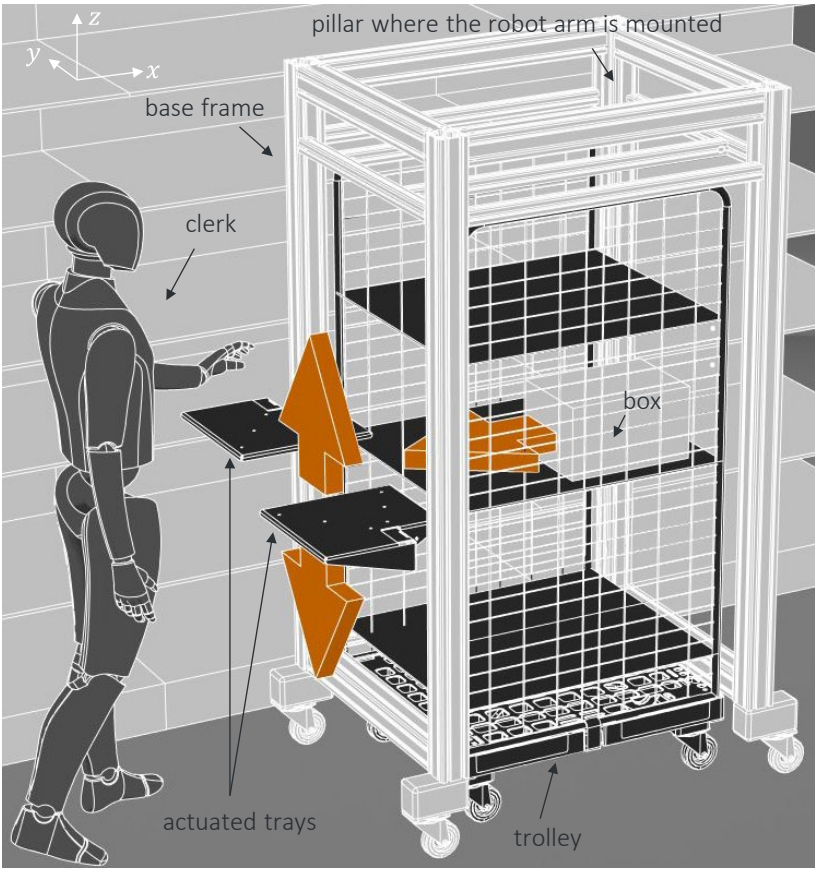


Figure 8.5 Arrangement of handling module and handling strategy

When the box is arranged on the tray, it is autonomously manipulated to an ergonomic height by one of the trays so that the staff can easily take the products out of the box and place them into the shelves. During this process the robot can prepare the next box by pushing it on the second tray. Although

the mass of the objects in this scenario ranges up to $10\ kg$, the robot does not need to be designed to lift that payload due to the proposed handling strategy.

Since the products to be placed on the shelves are already positioned on the trolley, it should be possible to integrate this trolley into the overall robotic system as effectively as possible, with the option that the mobile platform can exchange the individual trolleys. This requirement leads to the developed base frame of the handling module in which the trolley can be integrated as illustrated in Figure 8.5. The robot arm is attached to one column of the base frame and the actuated trays are provided on the opposite columns. An advantage of this handling concept and set-up becomes apparent in the workspace arrangements of human and robot. The zones do intersect merely in the area of the trays as shown in Figure 8.6.

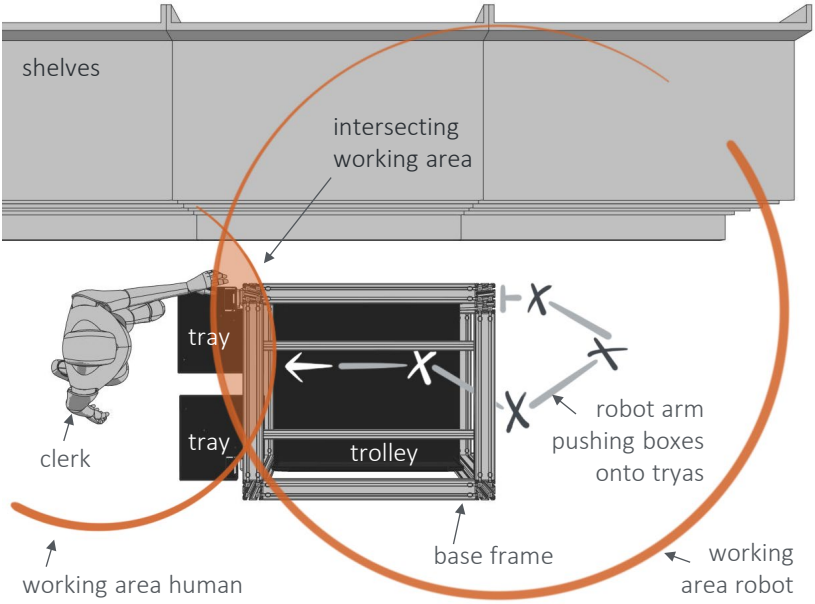


Figure 8.6 Arrangement of working areas

Up to this point, the handling concept presented is independent of the robot arm kinematics. However, during kinematics synthesis, it must be ensured

that the robot can reach all areas of the trolley as well as the desired shelf area. This has already been implemented in Chapter 8.3 through the task description in the context of pre-processing. The applied orientation conditions ensure that the last link of the robot arm moves parallel to the levels and sides of the trolley and shelf while reaching the desired working area, so that the risk of collisions is significantly reduced by the task description itself. Since the rough arrangement of the handling module has been defined, the more detailed design can be carried out. For this purpose, the robot kinematics is first integrated into the handling module. The generated solution of the $P_z R_z R_z R_z R_z$ scheme as a favorized structure is particularly well suited for this arrangement, since a complete pillar of the handling module is available as a mounting option for the robot arm, and thus the prismatic joint can be integrated ideally. Due to the height of the handling module, which is predetermined by the trolley, the stroke of the first axis is sufficiently large without creating additional interfering contours. As already mentioned in the requirements, the same basic kinematics should be used for both scenarios of refilling the products. Therefore, the robot arm is designed to be used in two configurations as visualized in Figure 8.7.

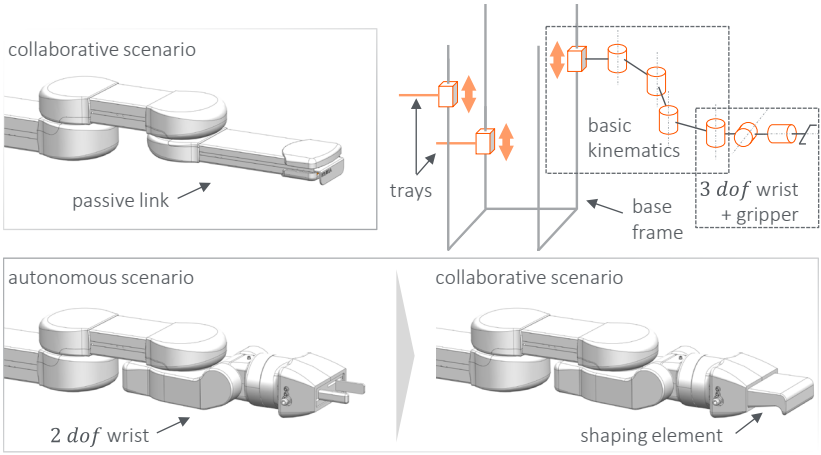


Figure 8.7 Robot arm configuration for the different scenarios

In the collaborative scenario, link four is designed as a passive structural part, as no additional degrees of freedom of a robot wrist are required here. This link can be replaced for the autonomous scenario, in which dexterous manipulability is required, and instead the robot arm is equipped with a wrist with two additional degrees of freedom and a two-jar gripper. In this configuration, the last axis, which is actually part of the regional structure, becomes the first axis of the local structure that allows orientation around the z axis, so that with the extension of the two axes wrist, a $3\ dof$ wrist kinematics is created. If the robot is generally to be equipped with a robot wrist, this configuration can also be used for the collaborative scenario by adding a shaping element to the gripper so that the boxes can be pushed more easily as visualized in the lower right corner of Figure 8.7.
To summarize, in the collaborative scenario the handling module has a total of seven axes, five of which are used for the robot's basic arm kinematics and the remaining two for the additional trays. In the autonomous scenario, a $R_{\mathrm{y}}R_{\mathrm{x}}$ wrist is added to the $P_{\mathrm{z}}R_{\mathrm{z}}R_{\mathrm{z}}R_{\mathrm{z}}R_{\mathrm{z}}$ scheme, giving the robot a total of seven degrees of freedom. The redundant dof is essential for collision avoidance in this narrow environment.
Now that the configurations and the overall arrangement of the system have been described, the mechatronic design for the principal axes will be discussed in the following, respecting the approaches to increase the inherent safety of the system that were presented in Chapter 7.2.

Sensor devices

To increase the safety of the system, sensors are used in addition to the approaches for improving the inherent safety. For the detection of a collision in the vertical direction, each link is equipped with pressure sensitive bumpers at the top and bottom. In combination with a newly developed cover concept published in patent [ZW20], collisions can thus be detected easily and by means of low-cost sensor technology and cover design, and also offer a slight compliance in the vertical direction to absorb impact energy. The link covers are made up of $3d$ printed parts and simple bent sheet metal as visualized in Figure 8.8.

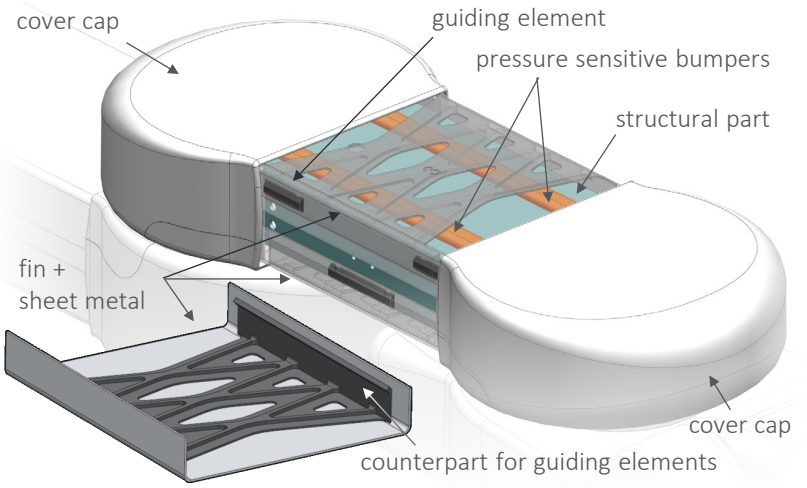


Figure 8.8 Novel cover concept with integrated sensor devices

The $3d$ printed fin elements are pressed into the sheet metals and they transmit the force in case of a collision to the pressure sensitive bumpers and are guided by $3d$ printed elements that are clicked into the structural parts of the link. The cover caps are also $3d$ printed and can be slid onto the sheet metals to cover the axis units.

To detect collisions in the horizontal movement, each drivetrain is equipped with a joint torque sensor. With these sensors, it is also possible to set the maximum permissible torque of the respective axis as a constraint, so that safe human-robot collaboration can take place by limiting and measuring the process forces.

Drivetrain design

In the REFILLS application, safe human-robot collaboration plays a crucial role, especially in the collaborative scenario. As described in Chapter 7.2, inherent safety can be increased by reducing the effective robot mass, and in addition to the structural parts, the drive components are particularly influential in this regard and are thus further investigated in the following.

Instead of integrating the drive components on the axis to be driven as in the standard setup for principal axes, they are relocated to the base of the robot

arm and this concept is further referred to as drive concept *base*. The developed drive concept as well as its comparison to the standard drivetrain is published in [ZHC20] and addressed in the following.

The drive components of all axes of the $P_zR_zR_zR_zR_z$ kinematics are integrated in the robot arm base, with the required torque and rotation being transmitted to the corresponding axis by using synchronous belt drives as visualized in Figure 8.9.

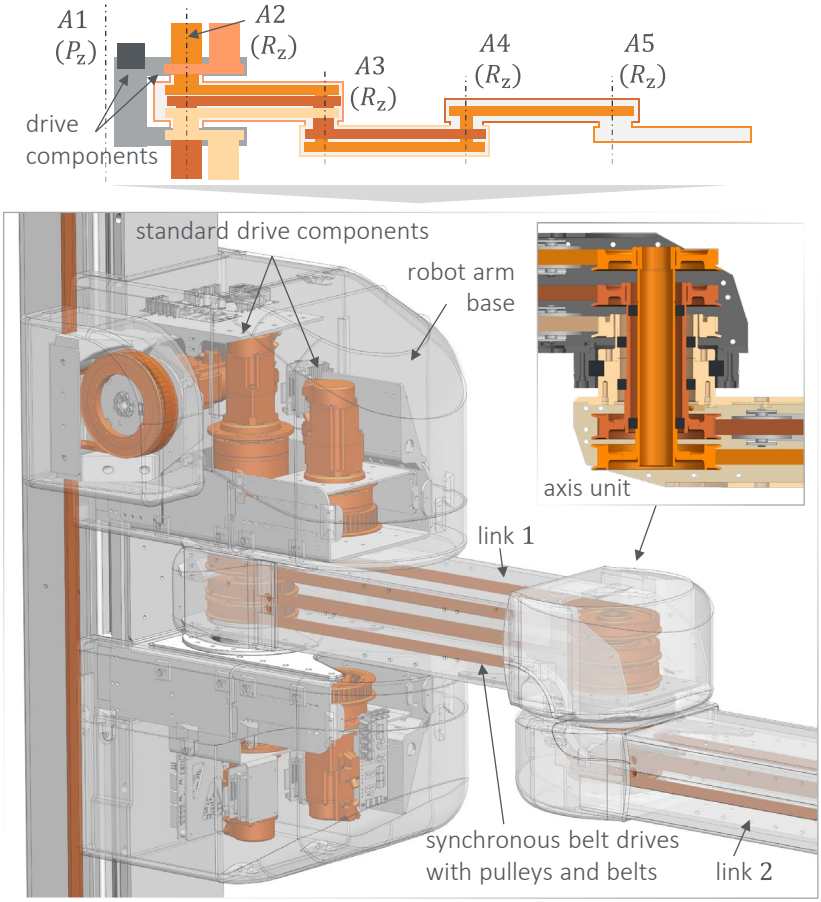


Figure 8.9 Drive concept for the REFILLS robot arm following [ZHC20]

The evaluation criteria listed in Table 17 are analyzed in the following regarding the selected drive concept to illustrate how the choice of the drive concept can influence the overall system and to show the advantages of this concept compared to the standard set-up for the REFILLS scenario. An overview of the comparison is shown in Figure 8.10 and a more detailed description is given in [ZHC20]. In the following the six evaluation criteria are further explained one after another, starting with the effective robot mass and continuing clockwise.

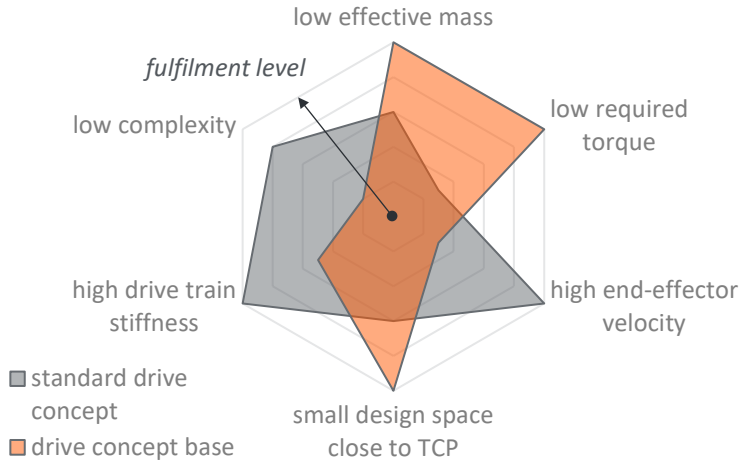


Figure 8.10 Evaluation of standard and base drive concept

Effective robot mass

Relocating the drive components to the robot arm base reduces the effective robot masses moved in the horizontal plane by about $40\ \%$ compared to a conventional design using similar components and considering the same requirements for the handling task [ZHC20]. This offers an enormous advantage especially for applications with human-robot collaboration, since the risk of injury is reduced. Furthermore, depending on equation (7.1), the robot speed can be increased, so that a more efficient process is possible. These two aspects are essential for the REFILLS application, so that a safe but also time efficient process can be ensured.

Required torque and end-effector velocity

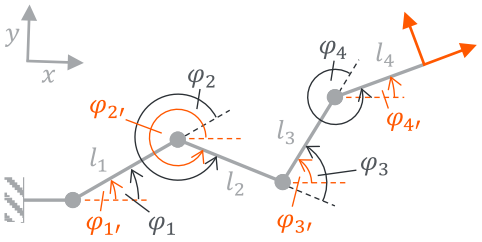
The modified arrangement of the drive components used here alters the kinematic relationship between the joint angles and the end-effector pose compared to the conventional set-up. The Jacobian $\boldsymbol{J}_{\mathrm{s}}$ of the standard set-up of the drive components, which maps the joint velocities to the end-effector velocities of the robot kinematics is given in equation (8.1) with $\mathrm{s}_{1\ldots n} = \sin(\varphi_1 + \cdots + \varphi_n)$ and $\mathrm{c}_{1\ldots n} = \cos(\varphi_1 + \cdots + \varphi_n)$.

$$\boldsymbol{J}_{\mathrm{s}} = \begin{bmatrix} 0 & -l_1\mathrm{s}_1 - l_2\mathrm{s}_{12} - l_3\mathrm{s}_{123} - l_4\mathrm{s}_{1234} \\ 0 & l_1\mathrm{c}_1 + l_2\mathrm{c}_{12} + l_3\mathrm{c}_{123} + l_4\mathrm{c}_{1234} \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{8.1}$$

$$\begin{matrix} -l_2\mathrm{s}_{12} - l_3\mathrm{s}_{123} - l_4\mathrm{s}_{1234} & -l_3\mathrm{s}_{123} - l_4\mathrm{s}_{1234} & -l_4\mathrm{s}_{1234} \\ l_2\mathrm{c}_{12} + l_3\mathrm{c}_{123} + l_4\mathrm{c}_{1234} & l_3\mathrm{c}_{123} + l_4\mathrm{c}_{1234} & l_4\mathrm{c}_{1234} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{matrix} \Bigg]$$

In the chosen drive concept *base*, the motor of an axis i, which drives link i, is not mounted in the previous link $i-1$, but in the robot's arm base. This results in a different relationship between the drive variables and the end-effector motion since the drive angles are independent of each other due to the mounting in the arm base. If only one axis is moved and the other motors remain in the zero position, only the orientation of the member with the driven axis changes and the subsequent links remain parallel to the x-axis of the world frame. The difference in axis angles based on the two concepts is shown in Figure 8.11. The altered kinematic relationship caused by the drive concept *base* can be expressed by redefining the axis angles. Instead of having the joint angle of axis i referenced to link $i-1$, the joint's zero position in this example is always parallel to the world's x-axis, so the joint angles $\boldsymbol{q}_{\mathrm{s}} = [s, \varphi_1, \ldots, \varphi_4]$ from the standard drive concept can be expressed using the joint angles $\boldsymbol{q}_{\mathrm{b}} = [s, \varphi_{1,}, \ldots, \varphi_{4,}]$ of the new drive concept *basis*. If

the joint variables are substituted according Figure 8.11 into equation (8.1), the Jacobian J_b is obtained in (8.2), which maps the joint velocities to the end-effector velocities taking the special drive concept into account.



standard drive concept
drive concept *base*

$$\varphi_{1'} = \varphi_1$$
$$\varphi_{2'} = \varphi_1 + \varphi_2$$
$$\varphi_{3'} = \varphi_1 + \varphi_2 + \varphi_3$$
$$\varphi_{4'} = \varphi_1 + \varphi_2 + \varphi_3 + \varphi_4$$

Figure 8.11 Kinematic relationship between the drive concepts

The changed kinematic relationship results in different end-effector velocities and required torques for the same process forces, compared to the standard drive concept, which can be seen from the Jacobians $\boldsymbol{J}_s$ and $\boldsymbol{J}_b$. The necessary torques $\boldsymbol{\tau}$ for a required process force $\boldsymbol{\gamma}$ can be determined directly from the transposed Jacobian, as given in equation (8.3) [SVS09].

$$\boldsymbol{J}_b = \begin{bmatrix} 0 & -l_1 \sin\varphi_{1'} & -l_2 \sin\varphi_{2'} & -l_3 \sin\varphi_{3'} & -l_4 \sin\varphi_{4'} \\ 0 & l_1 \cos\varphi_{1'} & l_2 \cos\varphi_{2'} & l_3 \cos\varphi_{3'} & l_4 \cos\varphi_{4'} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8.2}$$

In the standard drive concept, the required torque at axis i that must be applied for a certain process force $\boldsymbol{\gamma}$, is a function of the corresponding link length l_i in addition to the following link lengths $l_{i+1}, \dots, l_{dof}$.
In contrast, for the same process force only the link length l_i associated with the axis i is used as the lever arm in the calculation of the torque in the *base* drive concept. This behavior can be understood with equation (8.3) and the Jacobians J_s for the standard drive concept, and J_b for the *base* concept as given in (8.1) and (8.2), respectively. Due to this different axis couplings, smaller motors can be used within the drive concept *base* while maintaining

the same torques that must be applied, which has a positive influence on the robot weight.

$$\boldsymbol{\tau} = \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{\gamma} \tag{8.3}$$

This influence discussed above, which has a positive effect on the torque transmission of the *base* concept, has consequently a negative effect on the speed transmission since the mechanical power of the system remains the same. At the same motor speed, the maximum end-effector velocity is greater with the standard drive concept than with the drive concept *base*. Analogous to torque transmission, the magnitude of the influence depends on the link lengths. Since in human-robot collaboration applications the end-effector velocity is often limited to reduce the risk of injury, the negative impact is less severe and, depending on the application, can be accepted considering the advantages gained by the modified drive concept.

Design space

If the motors are relocated towards the base, two positive effects on the design space of the robot occur. Since the drive components are no longer located on the corresponding axis, the design space in the area of this axis is significantly reduced. With the developed system, about $40\ \%$ of the design space height is reduced in comparison to the standard drive concept when considering the last two robot links and, and if the requirements remain identical and similar components are assumed. [ZHC20]

In addition, as described above, motors with a lower maximum torque can be used compared to the standard drive concept, which also reduces the motor sizes as such. Nevertheless, the design space is significantly increased by $40\ \%$ in the areas where the drive components are relocated to, which in this case is the robot's arm base [ZHC20]. Depending on the application, it is therefore necessary to evaluate at which positions more design space is acceptable and where a large interfering contour must be avoided. In the REFILLS application it is important that the contour close to the end-effector is as flat as possible so that the system can handle within the narrow spaces

of the trolley levels and the shelves, making the *base* drive concept clearly advantageous regarding this criterion.

Drive train stiffness

By using the synchronous belts as transmission elements, the stiffness of the *base* drive train decreases compared to the standard concept. However, the extent of the effect can be limited by using a suitable belt tensioner and proper belt material. The REFILLS application and generally applications in the field of service robotics often require lower accuracies than in industrial robotics, thus this disadvantage might be accepted depending on the requirements. As the demanded accuracy of the REFILLS robot is in the millimeter range, which is a considerably low accuracy in comparison to industrial robotics, this drawback can be accepted here.

Complexity

The last point to be mentioned from Figure 8.10 as an evaluation criterion is the complexity of the overall system. For the *base* drive concept, the additional belt drive stages result in more components being used in the overall system than with a much more direct standard drive concept. Since a simple structure with as few components as possible is one of the most important principles for every designer, this aspect is probably the biggest disadvantage of the drive concept *base*. Depending on the requirements for the application, it must therefore be evaluated whether this disadvantage, which is not negligible, can be accepted as the other aspects such as effective mass or design space are more important, or if they are not. However, there is one simplification of the set-up caused by the drive concept *base* to be mentioned here, namely that this drive concept enables simple cable routing of the system. By using hollow shafts, the cables can be guided through the robot axes without the need for complex cable routing, which is often a challenge with the standard drive train design.

In summary, there is no such concept as the perfect solution, but rather, depending on the requirements, the more important characteristics must be identified, and a suitable drive concept must be developed accordingly.

In the context of the REFILLS application requirements, the two factors effective mass and design space close to the base play a much more important role than the other criteria mentioned and should therefore be prioritized. Regarding the evaluation of the criteria as shown in Figure 8.10, the *base* drive concept is therefore more suitable for this application than the standard drive concept and was therefore implemented in the robot arm as depicted in Figure 8.9. In addition to the regional structure, the robotic wrist plays a decisive role in the autonomous scenario, therefore the design of the wrist is briefly discussed below.

Wrist design

With the robot wrist, the regional structure of the robot arm is extended by two secondary axes. Together with the last R_z joint of the actual regional structure, a $3\,dof$ wrist is thus provided for dexterous manipulation as explained earlier and visualized in Figure 8.7. In contrast to the principal axes of the regional structure, the two additional axes are actuated according to a standard drive concept and the assembly of the wrist without cover is visualized in Figure 8.12.

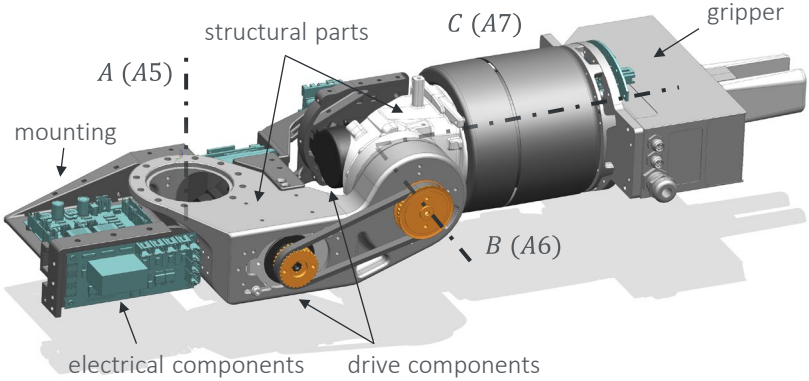


Figure 8.12 Robot wrist design

The reason for choosing the standard drive concept is on the one hand the otherwise exorbitantly increasing complexity to transfer the torque and motion and on the other hand, that the wrist should form a closed module so that a design as in the *base* drive concept would not be advantageous at

this point. In addition, a part of the wrist can be used from an already developed KUKA robot, so that previously evaluated industrial components can be used here and thus a reusability for another kinematics is demonstrated, which was an internal requirement for the mechanical design as mentioned in Table 18.

As already explained in the cover concept, $3d$ printed components are also used for the robot arm. This is also the case for the wrist, and lightweight design and economical production of the prototype components play an important role. Thus, the mounting for electrical components is manufactured via $3d$ printing with plastics to easily integrate the electrical parts into the assembly. The structural part between axes A and B that is colored grey in Figure 8.12 and depicted in Figure 8.13, is produced via selective laser melting (SLM) and the material used is a nickel titanium alloy.

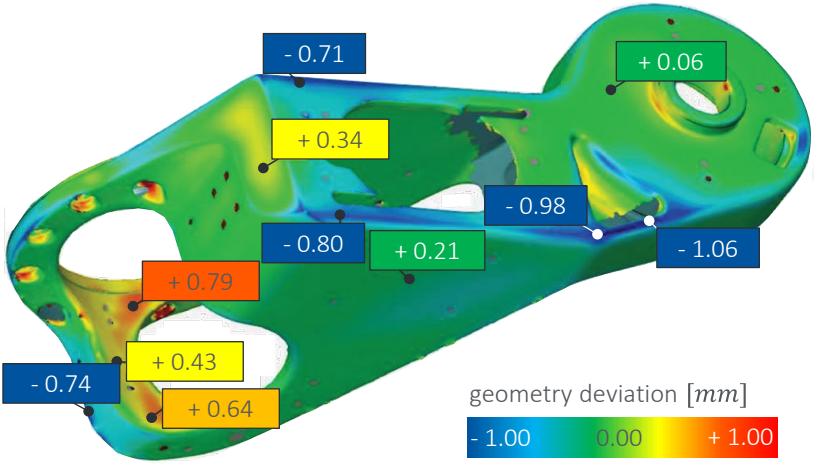


Figure 8.13 Geometry deviation between scan and CAD data

Although this process offers many advantages such as function integration into the component, lightweight design, and the possibility of undercuts that are difficult or impossible to produce with conventional manufacturing processes, it also has significant disadvantages. The component must be machined on all functional faces, for which extra fixtures or clamping areas on the component itself must be provided, and the dimensional accuracy is

also poor in comparison to generative manufacturing processes with plastics or traditional processes for metals. As visualized in Figure 8.13, the deviation of the structural part ranges from $-1.06\ mm$ up to $+0.79\ mm$ between the real model that was scanned by the manufacturer and the CAD data of the component, which reflects a poor dimensional accuracy. Overall, this is acceptable for the REFILLS wrist requirements, but it is evident that depending on the application, it must be evaluated whether this manufacturing process is suitable or if the disadvantages outweigh the benefits. Additive manufacturing processes can provide new opportunities, especially for the service robotics sector and offer great potential, but also have their drawbacks.

In summary, a SCARA special kinematic system with a novel handling strategy was developed as part of the REFILLS project. It can be modularly extended with a $2\ dof$ wrist and gripper to allow dexterous manipulation. All drive components of the regional structure have been relocated to the base of the arm to enable specific advantages for the given application and environment. For instance, the effective mass in the horizontal plane and the design space close to the end-effector were significantly reduced and thus specifically developed to meet the given requirements. After the most important aspects of the mechanical design have been explained, the following chapter provides a brief overview of the robot's assembly and its weaknesses.

8.7 REFILLS – hardware set-up

This chapter describes the mechanical set-up of the system and briefly explains the challenges during assembling and start-up.
The mechanics of the robot were built up successively axis by axis, starting with the regional structure and the first revolute joint axis $A2$. Pulleys and shafts are connected by means of tolerance rings, so that as little design space as possible is required for the shaft-hub connection, as the axis units would otherwise be too large. One challenge was the pressing of the components into each other, as there are up to five components mounted inside one another, which is given by the complex design and the belt drives

as additional transmission elements. If the assembly is to be carried out several times, assembly devices are indispensably. Stages of the assembly of the first and second revolute joint are depicted in Figure 8.14.

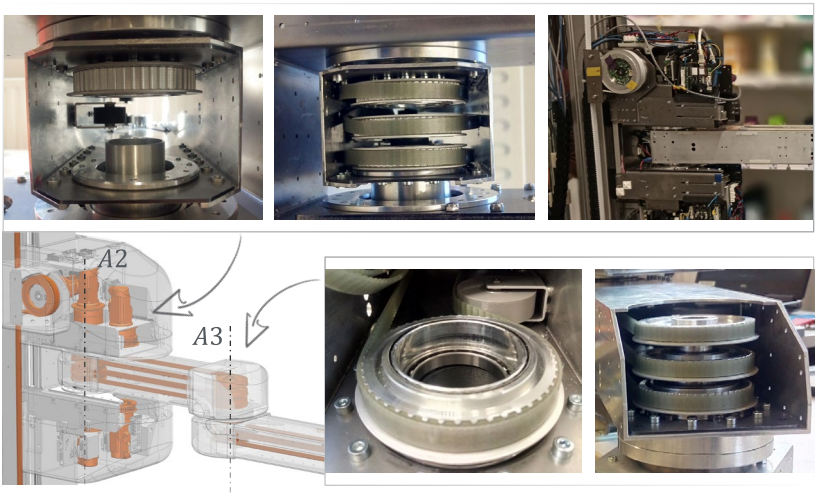
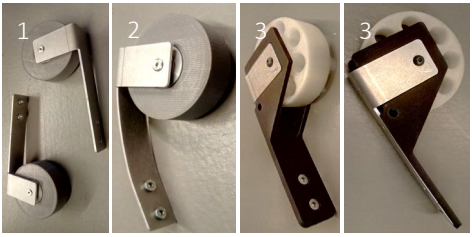


Figure 8.14 Mechanical set-up of axes $A2$ and $A3$

Another challenge was the belt tensioning. The mechanism originally intended was not stiff enough, so it bent as shown in Figure 8.15 on the left-hand side and could no longer provide the necessary pre-tensioning force.



1. Original belt tensioners
2. Deformation
3. Modified design

Figure 8.15 Modification of the belt tensioning device

To maintain the actual mechanism and avoid constructive changes, the identical components were used but extended by $3d$ printed stiffeners. This

change made it possible to apply the required pre-tension and to transfer the given loads.

The robot wrist was assembled next, with the electrical components pre-assembled on the $3d$ printed mounting and then attached to the module. The set-up is shown in Figure 8.16.

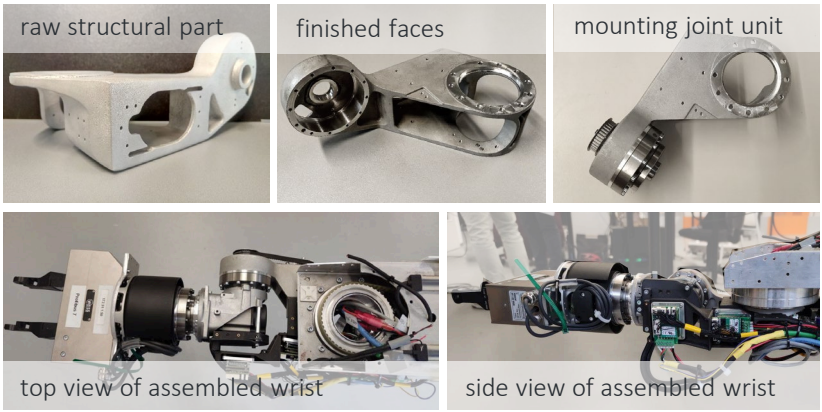


Figure 8.16 Mechanical set-up of robot wrist

After the mounting of the described assemblies, the covers can finally be attached, so that on the one hand the internal components are hidden and on the other hand the sensory monitoring for collisions in the vertical direction with the pressure sensitive bumpers is provided. The attachment of the fins to link 2, one overall construction of a link cover module and the configurable module of the passive link 4 for the collaborative scenario are visualized in Figure 8.17. The main advantages of the cover concept apart from the sensor integration, are the simple, fast, and cost-effective production. The combination of sheet metal and $3d$ printed components eliminate the need for extra molds, as required for instance in injection molding, while still enabling the realization of complex geometries. This is particularly advantageous for prototype construction, such as in the REFILLS project.

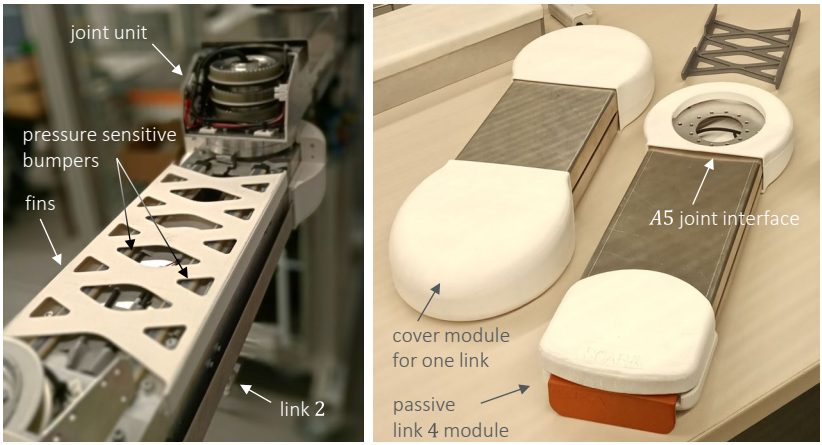


Figure 8.17 Cover assembly and link modules

In Figure 8.18 the final set-up of the REFILLS robotic system, the so-called handling module for autonomous and collaborative shelf filling is shown.

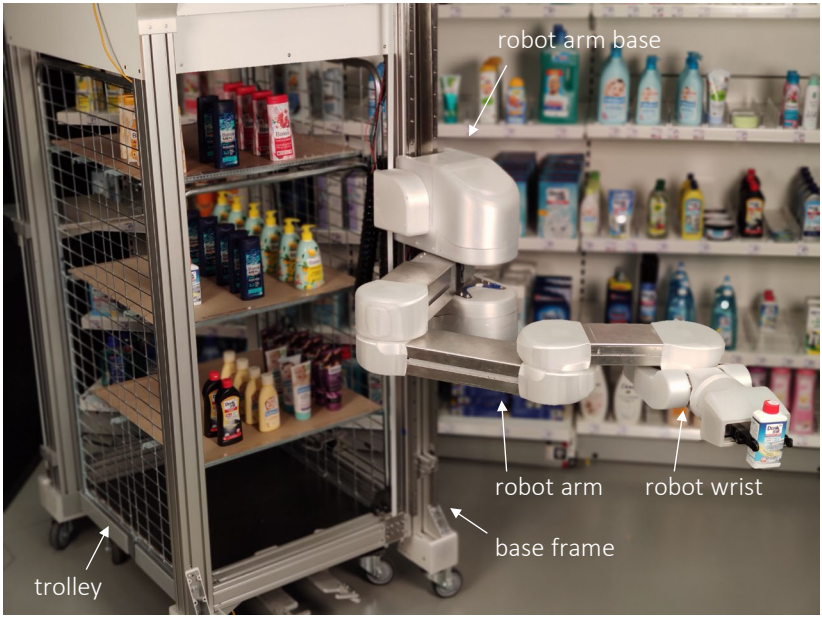


Figure 8.18 REFILLS handling module set-up in demo application environment

The robot arm is mounted with the linear axis on the base frame and the SCARA-like kinematics in combination with the drive concept *base* are particularly well suited for handling in the flat planes of the trolley and shelves. The robot wrist equips the system with additional degrees of freedom to enable dexterous manipulation of the products to be placed in the shelves for the autonomous scenario. In the collaborative scenario, the trays opposite the linear axis of the robot enable an ergonomic process for the staff as described earlier and depicted in Figure 8.5 and Figure 8.6. When all the products from the trolley have been placed on the shelves, the trolley on the base frame can be removed and exchanged, so that the process continues with more products until all the goods have been handled.
With the handling module, a system suitable for the requirements of the REFILLS application was developed, based on the problem definition utilizing the proposed method for task-based robot design and the tool provided. The detailed mechanical design of the system was then derived from the rough design and kinematic concept development to enable a tailor-made automation solution without major changes of the standard development process.
After the design process and the developed system have been described, the evaluation of the design method is discussed in the following Chapter 8.8.

8.8 REFILLS – design method evaluation

The method presented in Chapter 3.2 with the main components pre-processing, solver and post-processing as described in Chapter 5 to Chapter 7, respectively, is intended to support the developer in task-based kinematics synthesis within the development of new robotic systems. The systematic structure and the clear definition of input and output enable the method and the tool developed for this purpose to be integrated into standard development processes, which plays an essential role especially in industry. The processes covered by the methodology are partially automated, so that the concept development phase can be reduced from several weeks to a few days, which can offer decisive added value, especially in view of ever shorter development times and competition with other companies.

Using the presented methodology and the proposed tool, a suitable solution for the application within the REFILLS project could be generated within a few days, if it is assumed that the task has been clearly defined previously. The pure calculation time of the solver to determine suitable kinematic structures with suiting link length sets took $2.73\ h$ under the boundary conditions given in Chapter 8.3. The remaining time required depends on the user and took about 16 person-hours in this scenario. The developer could evaluate and adapt the proposed solutions with the help of the provided graphical user interface, so that a prioritization and identification of the favored concept was possible.

In the REFILLS project itself, 200 person-hours were planned for the hardware development of the handling module, of which approximately 40 person-hours were spent on the concept development of the kinematics and the rough arrangement of the components. In the REFILLS application example, the time for the concept development of the robot arm kinematics could be reduced by approximately $60\ \%$ when using the development tool and the proposed task-based robot design method. This significant reduction in development time demonstrates the efficiency of the method presented and not only offers considerable added value in the context of research projects but could also offer advantages in the design of robots for series development in the future.

The functionality of the presented method is proven by the fact that the tool has generated suitable kinematic structures with matching link length sets, one of which is identical to the structure developed according to the conventional development process. The robotic system developed on this basis meets the given requirements and can reach the target positions without colliding with the environment, so that the selected kinematic scheme is actually suitable in the real application, providing the proof of concept.

Nevertheless, it should be explicitly noted that the performance of automated synthesis depends largely on the task description and definition of the boundary conditions, so that a developer with a certain basic understanding is recommended to use the proposed method and development tool. In addition, the generated results must also be correctly

interpreted and evaluated by the user, so that finally a suitable overall system can be developed based on the intermediate results.

In summary, the method developed can be usefully integrated into the development process with significant resource reductions utilizing the tool provided.

9 Summary/Abstract

For decades, humans have been supported by automation technology, especially in the industrial sector. However, service robotics is also gradually gaining importance and an enormous growth in turnover is predicted for the next years. Due to ever shorter development times and the competition on the market, it is essential to develop these robotic systems efficiently. The varying environmental conditions and applications result in different requirements for the design of service robots compared to industrial robotics. The standard articulated arm kinematics, for instance, is not the ideal solution for all applications and the need for new kinematics is increasing, so that systems with less than six degrees of freedom can be more suitable or redundant robots are a valid solution. Although the field of robot kinematics is well researched, there are still gaps in the integration possibilities in development processes, ease of access and tools to support the engineers. For this reason, a methodology for semi-automated task-based kinematics development is presented in this thesis and the approach has been implemented in a tool to support the engineer in the development of new robotic systems. The approach is limited to the design of serial regional structures and is based on three core elements. In the pre-processor, the user interactively plans the motion task and defines the boundary conditions so that this data can be further processed within the solver. Suitable robot schemes with appropriate link lengths can then be identified fully automated within the framework of the type and dimensional synthesis. Finally, the results can be evaluated, prioritized, and modified by the user, so that on this basis the further development process can be continued, and the system can be designed. For the further development, important properties of the system as well as possibilities to increase the inherent safety are addressed. The procedure is illustrated and evaluated by the development of a robot within the REFILLS project for the optimization of intralogistics in supermarkets.

In summary, this work proposes a solution approach to develop new kinematics tailored to the required application in a semi-automated and efficient way.

10 Outlook

The methodology developed in the course of this work and the implementation in the design tool provide all the basic functions to support the engineer in the task-specific development of new robot kinematics to increase the efficiency of the development process. However, there is still a lot of potential to expand the scope of the tool in a meaningful way to provide even more opportunities for semi-automated robot design.

For example, an automated adjustment of the robot base would be a possibility to further optimize the kinematics and the application set-up in case that the installation position is not predetermined as input data.

In the solver, a robot wrist design could also be generated automatically based on input data, which is not necessary in the focus of the presented procedure but could certainly be a good assistance in the development process at some points. In the approach presented, the prismatic joints can only be defined in the first or last position of the kinematics, which could be extended in subsequent work to consider even more kinematic schemes.

For the post-processing, additional evaluation criteria could be integrated into the tool, such as a visualization of the robot pose-dependent performance indices within the workspace. Furthermore, it would be helpful if the results of the semi-automated robot design based on the presented tool are provided directly in the CAD environment, so that they can be further evaluated and developed there. It is also conceivable to automatically generate a simplified robot model based on the design parameters, which can then be produced quickly and inexpensively, for instance using generative manufacturing processes such as $3d$ printing via fused deposition modelling, in order to enable a haptic evaluation of the kinematics in the early development process.

Due to the input data of the minimum and maximum link lengths, the step size for discretization as well as the boundary conditions of the joint types and arrangement, the tool is ideally suited for the customized generation of robot solutions based on a modular system and could thus support the sales department or be directly available to the customer as a configuration tool. To enable a holistic approach to solutions, such a tool could be part of the ecosystem of automation. For example, in a future scenario, it would be an

enormous advance if interfaces were provided within the ecosystem and the robot controller automatically adapted to the generated robot solution, the model was available in the simulation and programming environment and components for the system could even be ordered automatically.
In summary, the work presented provides a foundation for an efficient semi-automated task-based kinematics development with much potential for further expansion.

11 Zusammenfassung

Seit Jahrzehnten wird der Mensch durch Automatisierungstechnik unterstützt, vor allem im industriellen Bereich. Doch auch die Servicerobotik gewinnt allmählich an Bedeutung und für die nächsten Jahre ist ein enormes Umsatzwachstum prognostiziert. Durch kürzere Entwicklungszeiten und die Konkurrenz am Markt ist es essenziell, Robotersysteme effizient und systematisch zu entwickeln. Durch Variation der Umgebungsbedingungen und Applikationen ergeben sich für die Entwicklung von Service Robotern andere Anforderungen verglichen zur klassischen industriellen Robotik. So ist beispielsweise nicht die Knickarm-Kinematik für alle Anwendungsfälle die ideale Lösung und der Bedarf an neuen Kinematiken steigt, sodass auch Systeme mit weniger als sechs Freiheitsgraden besser geeignet sein können oder redundante Roboter eine gute Lösung darstellen. Obwohl der Bereich der Roboterkinematik gut erforscht ist, weisen die Integrationsmöglichkeiten in Entwicklungsprozesse, Zugänglichkeit und Tools zur Unterstützung der Ingenieurinnen und Ingenieure noch immer Lücken auf. Aus diesem Grund wird im Rahmen dieser Arbeit eine Methodik zur teilautomatisierten taskbasierten Kinematik Entwicklung vorgestellt und in einem Tool implementiert, um bei der Entwicklung neuer Robotersysteme zu unterstützen. Der Ansatz beschränkt sich auf serielle Regionalstrukturen und setzt sich aus drei Kernelementen zusammen. Im Pre-Processor erfolgt die Planung der Bewegungsaufgabe und das Definieren der Randbedingungen, sodass diese Daten im Solver weiterverarbeitet werden können und geeignete Roboterstrukturen mit passenden Gliedlängen im Rahmen der Struktur- und Maßsynthese vollautomatisiert identifiziert werden. Abschließend können die Ergebnisse evaluiert, priorisiert und modifiziert werden, sodass auf dieser Basis der weitere Entwicklungsprozess fortgesetzt werden kann. Für das Ausgestalten werden wichtige Eigenschaften der Systeme sowie Möglichkeiten zur Erhöhung inhärenter Sicherheit adressiert. Das Vorgehen wird anhand der Entwicklung eines Roboters im Rahmen des REFILLS Projektes veranschaulicht und evaluiert.

Zusammenfassend wird in dieser Arbeit ein Lösungsansatz vorgeschlagen, um neue Kinematiken maßgeschneidert auf die geforderte Anwendung teilautomatisiert und effizient zu entwickeln.

12 Bibliography

[AJM13] Al-Dois, H.; Jha, A. K.; Mishra, R. B.
Task-based design optimization of serial robot manipulators
In: Engineering Optimization, 45 (2013) 6, DOI 10.1080/0305215X.2012.704027, S. 647–658.

[AL09] Aristidou, A.; Lasenby, J.
Inverse Kinematics: a review of existing techniques and introduction of a new fast iterative solver
Technical Report, University of Cambridge, Cambridge University Engineering Department, 2009, https://www.researchgate.net/publication/273166356_Inverse_Kinematics_a_review_of_existing_techniques_and_introduction_of_a_new_fast_iterative_solver (zuletzt geprüft am 06.03.2021).

[AZE16] Aziz, M. A. S.; Zhanibek, M.; Elsayed, A. S. A.; AbdulRazic, M. O. M.; Yahya, S.; Almurib, H. A.; Moghavvemi, M.
Design and analysis of a proposed light weight three DOF planar industrial manipulator
In: 2016 IEEE Industry Applications Society Annual Meeting, Portland, OR, USA, 02.10.2016 - 06.10.2016, ISBN 978-1-4799-8397-1, S. 1–7.

[BGI11] BGIA – Institut für Arbeitsschutz der Deutschen Gesetzlichen Unfallversicherung
BG/BGIA-Empfehlungen für die Gefährdungsbeurteilung nach Maschinenrichtlinie, Gestaltung von Arbeitsplätzen mit kollaborierenden Robotern, Sankt Augustin: , 2011.

[CB95] Chen, I.-M.; Burdick, J. W.
Determining task optimal modular robot assembly configurations
In: 1995 IEEE International Conference on Robotics and Automation, Nagoya, Japan, 21-27 May 1995, ISBN 0-7803-1965-6, S. 132–137.

[CB97] Chocron, O.; Bidaud, P.
Evolutionary algorithms in kinematic design of robotic systems
In: 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems. Innovative Robotics for Real-World Applications. IROS '97, Grenoble, France, 7-11 Sept. 1997, ISBN 0-7803-4119-8, S. 1111–1117.

[CCP10] Castejón, C.; Carbone, G.; Prada, G.; Ceccarelli, M.
A Multi-Objective Optimization of a Robotic Arm for Service Tasks
In: Assoc. of Mechanical Eng. and Technicians of Slovenia (Hrsg.) , Journal of Mechanical Engineering, 2010.

[CDE12] Chandrasekaran, K.; Djuric, A.; ElMaraghy, W. H.
Selection Catalogue of Kinematic Configuration for Pick and Place Application
In: ElMaraghy, H. A. (Hrsg.) , Enabling Manufacturing Competitiveness and Economic Sustainability, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, ISBN 978-3-642-23859-8, S. 41–46.

[CL04] Ceccarelli, M.; Lanni, C.
A multi-objective optimum design of general 3R manipulators for prescribed workspace limits
In: Mechanism and Machine Theory, 39 (2004) 2, DOI 10.1016/S0094-114X(03)00109-5, S. 119–132.

[COC07] Carbone, G.; Ottaviano, E.; Ceccarelli, M.
An optimum design procedure for both serial and parallel manipulators
In: Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 221 (2007) 7, DOI 10.1243/0954406JMES367, S. 829–843.

[COC08] Castelli, G.; Ottaviano, E.; Ceccarelli, M.
A Fairly General Algorithm to Evaluate Workspace Characteristics of Serial and Parallel Manipulators #
In: Mechanics Based Design of Structures and Machines, 36 (2008) 1, DOI 10.1080/15397730701729478, S. 14–33.

[Cor13] Corves, B.
Unveröffentlichter Vorlesungsumdruck Mechanismentechnik,
Vorlesungsbegleitender Umdruck zu den Lehrveranstaltungen „Elektromechanische Antriebstechnik“ und „Bewegungstechnik“, Aachen: , 2013.

[CR96] Chedmail, P.; Ramstein, E.
Robot mechanism synthesis and genetic algorithms
In: IEEE International Conference on Robotics and Automation, Minneapolis, MN, USA, 22-28 April 1996, ISBN 0-7803-2988-0, S. 3466–3471.

[DIN12100] DIN EN ISO, 12100
Sicherheit von Maschinen - Allgemeine Gestaltungsleitsätze
Norm, Verein Deutscher Ingenieure, Berlin: Beuth Verlag GmbH, 2011.

[EC16] European Commission; Consortio CREATE
GRANT AGREEMENT, NUMBER - 731590 - REFILLS, 2016.

[FG13] Feldhusen, J.; Grote, K.-H.
Pahl/Beitz Konstruktionslehre
Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, ISBN 978-3-642-29568-3.

[FLK12] Flacco, F.; Luca, A. de; Khatib, O.
Motion control of redundant robots under joint constraints: Saturation in the Null Space
In: 2012 IEEE International Conference on Robotics and Automation (ICRA), St Paul, MN, USA, 14.05.2012 - 18.05.2012, ISBN 978-1-4673-1405-3, S. 285–292.

[GR82] Gupta, K. C.; Roth, B.
Design Considerations for Manipulator Workspace
In: Journal of Mechanical Design, 104 (1982) 4, DOI 10.1115/1.3256412, S. 704–711.

[Gup86] Gupta, K. C.
On the Nature of Robot Workspace
In: The International Journal of Robotics Research, 5 (1986) 2, DOI 10.1177/027836498600500212, S. 112–121.

[Hes16] Hesse, S.
Grundlagen der Handhabungstechnik
Müchen: Hanser, 4., neu bearbeitete und erweiterte Auflage, 2016, ISBN 978-3-446-44432-4.

[Hes98] Hesse, S.
Industrieroboterpraxis, Automatisierte Handhabung in der Fertigung
Wiesbaden: Vieweg+Teubner Verlag, 1998, ISBN 978-3-322-88982-9.

[HKP21] Huczala, D.; Kot, T.; Pfurner, M.; Heczko, D.; Oščádal, P.; Mostýn, V.
Initial Estimation of Kinematic Structure of a Robotic Manipulator as an Input for Its Synthesis
In: Applied Sciences, 11 (2021) 8, DOI 10.3390/app11083548, S. 3548.

[HSV09] Ham, R.; Sugar, T.; Vanderborght, B.; Hollander, K.; Lefeber, D.
Compliant actuator designs
In: IEEE Robotics & Automation Magazine, 16 (2009) 3, DOI 10.1109/MRA.2009.933629, S. 81–94.

[HWP20] Hu, M.; Wang, H.; Pan, X.
Multi-objective global optimum design of collaborative robots
In: Structural and Multidisciplinary Optimization, (2020) , DOI 10.1007/s00158-020-02563-x.

[IA16] Icer, E.; Althoff, M.
Cost-optimal composition synthesis for modular robots
In: 2016 IEEE Conference on Control Applications (CCA), Buenos Aires, Argentina, 19.09.2016 - 22.09.2016, ISBN 978-1-5090-0755-4, S. 1408–1413.

[ISO13854] ISO, 13854
Sicherheit von Maschinen - Mindestabstände zur Vermeidung des Quetschens von Körperteilen
Norm, Deutsches Institut für Normung e. V., Berlin: Beuth Verlag GmbH.

[ISO13857] ISO, 13857
Sicherheit von Maschinen - Sicherheitsabstände gegen das Erreichen von Gefährdungsbereichen mit den oberen und unteren Gliedmaßen
Norm, Deutsches Institut für Normung e. V., Berlin: Beuth Verlag GmbH.

[ISO8373] ISO/DIS, 8373
Robotics
Norm, International Organization for Standardization, Vernier, Geneva: ISO copyright office, 2020.

[KA06] Khan, W. A.; Angeles, J.
The Kinetostatic Optimization of Robotic Manipulators: The Inverse and the Direct Problems
In: Journal of Mechanical Design, 128 (2006) 1, DOI 10.1115/1.2120808, S. 168–178.

[KB06] Kucuk, S.; Bingul, Z.
Comparative study of performance indices for fundamental robot manipulators
In: Robotics and Autonomous Systems, 54 (2006) 7, DOI 10.1016/j.robot.2006.04.002, S. 567–573.

[KCH15] Kerle, H.; Corves, B.; Hüsing, M.
Getriebetechnik, Grundlagen, Entwicklung und Anwendung ungleichmäßig übersetzender Getriebe
Wiesbaden: Springer Vieweg, 5., überarbeitete und erweiterte Auflage, 2015, ISBN 978-3-658-10057-5.

[KM20] Kutzbach, N.; Müller, C.
World Robotics 2020 – Industrial Robots, Frankfurt am Main, Germany: IFR Statistical Department, 2020.

[KW81] Kumar, A.; Waldron, K. J.
The Workspaces of a Mechanical Manipulator
In: Journal of Mechanical Design, 103 (1981) 3, DOI 10.1115/1.3254968, S. 665–672.

[LB99] Leger, C.; Bares, J.
Automated Task-Based Synthesis and Optimization of Field Robots, 1999.

[Leg99] Leger, C.
Automated Synthesis and Optimization of Robot Configurations: An Evolutionary Approach
PhD Thesis, Pittsburgh: Carnegie Mellon University, The Robotics Institute, 1999.

[MA00] McCarthy, J. M.; Ahlers, S.
Dimensional Synthesis of Robots using a Double Quaternion Formulation of the Workspace
In: Hollerbach, J. M.; Koditschek, D. E. (Hrsg.) , Robotics Research, London: Springer London, 2000, ISBN 978-1-4471-1254-9, S. 3–8.

[Mai19] Maier, H.
Grundlagen der Robotik
Berlin: VDE VERLAG GMBH, 2., überarbeitete und erweiterte Auflage, 2019, ISBN 978-3-8007-5071-9.

[Mat20] MathWorks
Robotics System Toolbox, User's Guide
Natick: , 2020.

[McC00] McCarthy, J. M.
Mechanism synthesis theory and the design of robots
In: 2000 ICRA. IEEE International Conference on Robotics and Automation, San Francisco, CA, USA, 24-28 April 2000, ISBN 0-7803-5886-4, S. 55–60.

[Mer06] Merlet, J.-P.
Parallel Robots
Dordrecht: Springer Netherlands, 2006, ISBN 978-1-4020-4133-4.

[MGP20] Müller, C.; Graf, B.; Pfeiffer, K.; Bieller, S.; Kutzbach, N.; Röhricht, K.
World Robotics 2020 – Service Robots, Frankfurt am Main, Germany: IFR Statistical Department, 2020.

[MLS94] Murray, R. M.; Li, Z.; Sastry, S.
A mathematical introduction to robotic manipulation
Boca Raton, Fla.: CRC Press, 1994, ISBN 0849379814.

[MSV16] Moulianitis, V. C.; Synodinos, A. I.; Valsamos, C. D.; Aspragathos, N. A.
Task-Based Optimal Design of Metamorphic Service Manipulators
In: Journal of Mechanisms and Robotics, 8 (2016) 6, DOI 10.1115/1.4033665.

[ODM09] Oetomo, D.; Daney, D.; Merlet, J.-P.
Design Strategy of Serial Manipulators With Certified Constraint Satisfaction
In: IEEE Transactions on Robotics, 25 (2009) 1, DOI 10.1109/TRO.2008.2006867, S. 1–11.

[OKB19] Ozakyol, H.; Karaman, C.; Bingul, Z.
Advanced robotics analysis toolbox for kinematic and dynamic design and analysis of high-DOF redundant serial manipulators
In: Computer Applications in Engineering Education, 27 (2019) 6, DOI 10.1002/cae.22160, S. 1429–1452.

[PBH21] Pastor, R.; Bobovský, Z.; Huczala, D.; Grushko, S.
Genetic Optimization of a Manipulator: Comparison between Straight, Rounded, and Curved Mechanism Links
In: Applied Sciences, 11 (2021) 6, DOI 10.3390/app11062471, S. 2471.

[PS14] Patel, S.; Sobh, T.
Goal directed design of serial robotic manipulators
In: 2014 Zone 1 Conference of the American Society for Engineering Education (ASEE Zone 1), Bridgeport, CT, USA, 03.04.2014 - 05.04.2014, ISBN 978-1-4799-5233-5, S. 1–6.

[PS15a] Patel, S.; Sobh, T.
Task based synthesis of serial manipulators
In: Journal of advanced research, 6 (2015) 3, DOI 10.1016/j.jare.2014.12.006, S. 479–492.

[PS15b] Patel, S.; Sobh, T.
Manipulator Performance Measures - A Comprehensive Literature Survey
In: Journal of Intelligent & Robotic Systems, 77 (2015) 3-4, DOI 10.1007/s10846-014-0024-y, S. 547–570.

[PSM04] Perez, A.; Su, H.-J.; McCarthy, J. M.
Synthetica 2.0: Software for the Synthesis of Constrained Serial Chains
In: ASME 2004 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Salt Lake City, Utah, USA, 2004, ISBN 0-7918-4695-4, S. 1363–1369.

[Raa86] Raab, H. H.
Handbuch Industrieroboter, Bauweise · Programmierung Anwendung · Wirtschaftlichkeit
Wiesbaden: Vieweg+Teubner Verlag, 2., neubearbeitete und erweiterte Auflage, 1986, ISBN 978-3-322-83600-7.

[Rie92] Rieseler, H.
Roboterkinematik — Grundlagen, Invertierung und Symbolische Berechnung
Wiesbaden: Vieweg+Teubner Verlag, 1992, ISBN 978-3-528-06515-7.

[RKO14] Ramirez Rodriguez, D. A.; Kotlarski, J.; Ortmaier, T.
Automatic Generation of Serial Robot Architectures from Required Motion Directions
In: Overmeyer, L.; Shkodyrev, V. P. (Hrsg.) , AST - Symposium on Automated Systems and Technologies, Garbsen: PZH-Verl. TEWISS - Technik und Wissen, 2014, ISBN 978-3-944586-84-7.

[Ros16] Rost, S.
Entwurf, Analyse und Regelung einer kinematisch redundanten Roboterstruktur mit hydraulischen Antrieben variabler Nachgiebigkeit
Dissertation, Universitätsbibliothek Braunschweig, 2016, DOI 10.24355/dbbs.084-201606290936-0.

[Ruo16] Ruokonen, A.
ROBOT BUILDER
https://robotselection.wordpress.com/robot-builder/ (zuletzt geprüft am 09.08.2021).

[Sah20] Saha, S. K.
RoboAnalyzer
http://www.roboanalyzer.com/ (zuletzt geprüft am 09.08.2021).

[SCH20] SCHUNK GmbH & Co. KG
Montage- und Betriebsanleitung WSG 50
https://schunk.com/fileadmin/pim/docs/IM0004934.PDF (zuletzt geprüft am 29.04.2021).

[Sch87] Schopen, M.
Die Auswahl von Handhabungsgeräten aufgrund der charakteristischen Merkmale ihrer kinematischen Strukturen
Düsseldorf: VDI-Verlag, 1987.

[SHV20] Spong, M. W.; Hutchinson, S.; Vidyasagar, M. (Hrsg.)
Robot modeling and control
Hoboken, NJ: Wiley, Second edition, 2020, ISBN 978-1-119-52404-5.

[Sic21] Siciliano, B.
Project ID: 731590
http://www.refills-project.eu/ (zuletzt geprüft am 22.04.2021).

[SKK02] Shiakolas, P. S.; Koladiya, D.; Kebrle, J.
Optimum Robot Design Based on Task Specifications Using Evolutionary Techniques and Kinematic, Dynamic, and Structural Constraints
In: Inverse Problems in Engineering, 10 (2002) 4, DOI 10.1080/1068276021000004706, S. 359–375.

[SO19] Shirafuji, S.; Ota, J.
Kinematic Synthesis of a Serial Robotic Manipulator by Using Generalized Differential Inverse Kinematics
In: IEEE Transactions on Robotics, 35 (2019) 4, DOI 10.1109/TRO.2019.2907810, S. 1047–1054.

[SVS09] Siciliano, B.; Villani, L.; Sciavicco, L.; Oriolo, G.
Robotics, Modeling Planning and Control
London: Springer London, 2009, ISBN 978-1-84628-641-4.

[TS15066] TS, 15066
Roboter und Robotikgeräte - Kollaborierende Roboter
Norm, Deutsches Institut für Normung e. V., Berlin: Beuth Verlag GmbH.

[TS81] Tsai, Y. C.; Soni, A. H.
Accessible Region and Synthesis of Robot Arms
In: Journal of Mechanical Design, 103 (1981) 4, DOI 10.1115/1.3254990, S. 803–811.

[TS84] Tsai, Y.; Soni, A.
The effect of link parameter on the working space of general 3R robot arms
In: Mechanism and Machine Theory, 19 (1984) 1, DOI 10.1016/0094-114X(84)90004-1, S. 9–16.

[VDI2221-1] VDI, 2221-1
Entwicklung technischer Produkte und Systeme
Norm, Verein Deutscher Ingenieure, Berlin: Beuth Verlag GmbH, 2019.

[VDI2221-2] VDI, 2221-2
Entwicklung technischer Produkte und Systeme
Norm, Verein Deutscher Ingenieure, Berlin: Beuth Verlag GmbH, 2019.

[VDI2222-2] VDI, 2222-2
Konstruktionsmethodik
Norm, Verein Deutscher Ingenieure, Berlin: Beuth Verlag GmbH, 1982.

[VDI2861-2] VDI, 2861-2
Kenngrößen für Industrieroboter
Norm, Verein Deutscher Ingenieure, Düsseldorf: VDI-Verlag, 1988.

[Vol78] Volmer, J.
Getriebetechnik
Wiesbaden: Vieweg+Teubner Verlag, 1978, ISBN 978-3-528-04096-3.

[Vol86] Volmer, J. (Hrsg.)
Industrieroboter, Entwicklung
Heidelberg: Hüthig, 2., durchges. Aufl., 1986, ISBN 3-7785-1140-8.

[WPK18] Wirtz, J.; Patterson, P. G.; Kunz, W. H.; Gruber, T.; Lu, V. N.; Paluch, S.; Martins, A.
Brave new world: service robots in the frontline
In: Journal of Service Management, 29 (2018) 5, DOI 10.1108/JOSM-04-2018-0119, S. 907–931.

[YL84] Yang, D. C. H.; Lee, T. W.
Heuristic combinatorial optimization in the design of manipulator workspace
In: IEEE Transactions on Systems, Man, and Cybernetics, SMC-14 (1984) 4, DOI 10.1109/TSMC.1984.6313328, S. 571–580.

[YSK13] Yadmellat, P.; Shafer, A. S.; Kermani, M. R.
Design and development of a safe robot manipulator using a new actuation concept
In: 2013 IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany, 06.05.2013 - 10.05.2013, ISBN 978-1-4673-5643-5, S. 337–342.

[ZBH07] Zacharias, F.; Borst, C.; Hirzinger, G.
Capturing robot workspace structure: representing robot capabilities
In: 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, USA, 29.10.2007 - 02.11.2007, ISBN 978-1-4244-0911-2, S. 3229–3236.

[ZCY19] Zhang, D.; Cursi, F.; Yang, G.-Z.
WSRender: A Workspace Analysis and Visualization Toolbox for Robotic Manipulator Design and Verification
In: IEEE Robotics and Automation Letters, 4 (2019) 4, DOI 10.1109/LRA.2019.2929986, S. 3836–3843.

[ZHC20] Zumpe, V.; Hüsing, M.; Corves, B.
Höhere inhärente Sicherheit bei Mensch-Roboter-Kollaborationen durch geeignete Antriebskonzepte
In: 6. IFToMM D-A-CH Konferenz 2020, 2020.

[ZHH19] Zumpe, V.; Horeis, J.; Hüsing, M.; Corves, B.
Pre-Processing für taskbasierte Kinematik-Synthese
In: Fünfte IFToMM D-A-CH Konferenz 2019, 2019.

[ZLZ10] Zou, H.; Liang, Q.; Zhang, Q.
Theory and method of mechanism system design
In: Frontiers of Mechanical Engineering in China, 5 (2010) 4, DOI 10.1007/s11465-010-0116-8, S. 399–411.

[ZR20] Zumpe, V.; Riedel, M.
Verfahren und Vorrichtungen zum Befüllen von Fachböden eines Regals mit Warenstücken aus einem Regalwagen
Patent, DE102018218709A1, B65G 1/04 (2006.01).

[ZRK04] Zinn, M.; Roth, B.; Khatib, O.; Salisbury, J. K.
A New Actuation Approach for Human Friendly Robot Design
In: The International Journal of Robotics Research, 23 (2004) 4-5, DOI 10.1177/0278364904042193, S. 379–398.

[ZW20] Zumpe, V.; Wünsch, M.
Roboter mit wenigstens einer Abdeckung und wenigstens einem Kontaktsensor
Patent, DE102019201516B4, B25J 19/06 (2006.01).

List of Figures

List of Tables

Appendix

Table 22 c_robotList class details

Constructor summary	
c_robotList	Robotlist to store results from kinematic synthesis of different kinematic schemes
Property summary	
errorflag	False if solution exists, true if no config can reach all goals
goal	Goal positions; stored as matrix [x1 y1 z1,..., xn yn zn]
goalR	Goal orientation, rotation matrices; entries 3x3xn
goaltype	'pose' or 'position' if 'pose': goalR necessary for IK
linklengthCombi	All possible link length combinations
lmax	Maximum link length
lmin	Minimum link length
nrPrio	Nr that defines how many solutions to prioritize
robotPrio	Best solutions wrt minimum link length sum
robotSelection	Suiting and selected robot (as objects)
robotSorted	Robotlist sorted wrt min link lengths (list with robot number) [robotNr,sumLinkLength]
robotType	Stores robot type; jnt+ax: e.g. RzRyRy
robotlistEvaluation	Stores list with successful robots, sorted wrt link length
step	Step size for discretization
successKin	List with numbers of successful robots (for each position)
suitingLinklengthCombi	Eliminate link length combis that cannot reach goal positions
Method summary	
animatePrio	Animate best solutions, moving to all target positions
animatePrioReconfigure	Visualize the best solutions and animate them moving to all target positions with the option to reconfigure the robot
dimSyn	Function for dimensional synthesis
genenvironment	Load stl data and visualize in MATLAB environment
generateRdes	Calculate the rotation matrix based on vector input
generateSuitingLinks	Preselect just the suiting link length combinations
getCADpostitions	Get x, y and z values of the desired positions within the model
getPrios	Get the prioritized solutions
prepareLinks	Generate link length combinations
showAll	Show all solutions
showGoals	Visualize the target positions
showPrio	Show the prioritized objects
visualizeWs	Visualize the robot's workspace

Table 23 c_robot class details

Constructor summary	
c_robot	Robot object for calculations and visualization
Property summary	
CADpositions	Helper to get positions from model data
base	Robot base [x y z] in [m]
dof	Degree of freedom: int
dtcp	y,z offset of tcp, x is given by previous link length: [y z] in [m]
error	Error of tcp between desired and actual position
goal	Goal positions of the robot. vector: [x y z] in [m]
goalR	Goal-orientation defined as rotation matrix 3x3
goaltype	'pose' or 'position', if 'pose': goalR necessary for IK
jntAxis	Joint axis orientation x:[1 0 0] y:[0 1 0] z:[0 0 1]
jntHome	Joint home position: vector in [rad]
jntLim	Joint limits
jntType	Joint type: 'revolute' / 'prismatic'
links	Link lengths: vector in [m]
nrIer	Number of iterations to solve via SNS, scalar
qMat	3dim matrix to store qVec for more than one position
qVec	Matrix with sets of q to reach goal
qVecList	List for storing IK solutions for later use
robotRBT	Rigid body tree from robotics systems toolbox
steps	Steps for discretization on joints for workspace visualization
transform	Transformation matrix to reach goal
visDetJ	Matrix: row: px1 py1 pz1; ...; pxn
wsPT	Generated points in cartesian space for workspace visualization
Method summary	
animate	Animate robot moving to all target positions
animateReconfigure	Animate robot moving to all target positions; reconfigure option
build	Set up the robot (RBT) based on input data
genenvironment	Load stl data and visualize in MATLAB environment
generateRdes	Calculate the rotation matrix based on vector input
generateWsPt	Generate all workspace points based on joint discretization
getCADpositions	Get x, y and z values of the desired positions within the model
reconfigure	Reconfigure the robot kinematic (e.g., elbow up / down)
showGoals	Visualize the target positions
solveIKsns	Solve the inverse kinematics based on the SNS solver
visJ	Visualize values of det(J) via color code of points in space
visualizeWs	Visualize the robot's workspace

Table 24 Visualization of the REFILLS scenario solutions - I

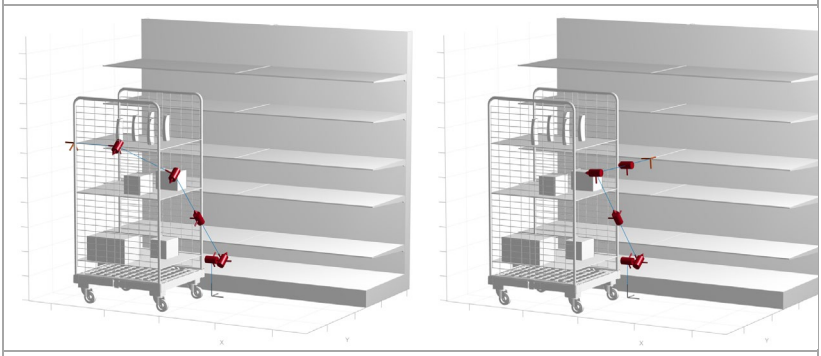
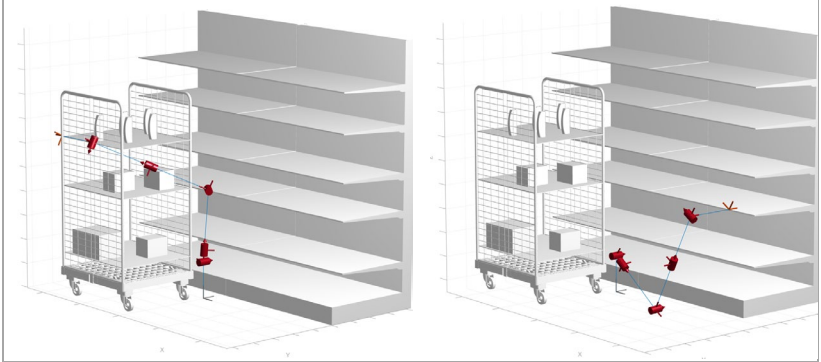
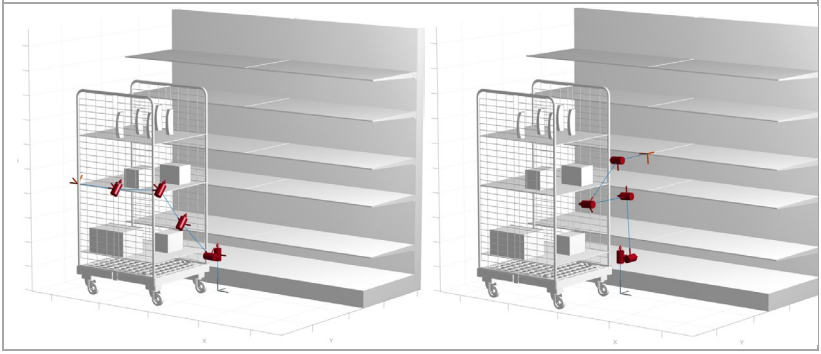
$no.02$

$no.04$

$no.09$


Table 25 Visualization of the REFILLS scenario solutions - II

$no.13$