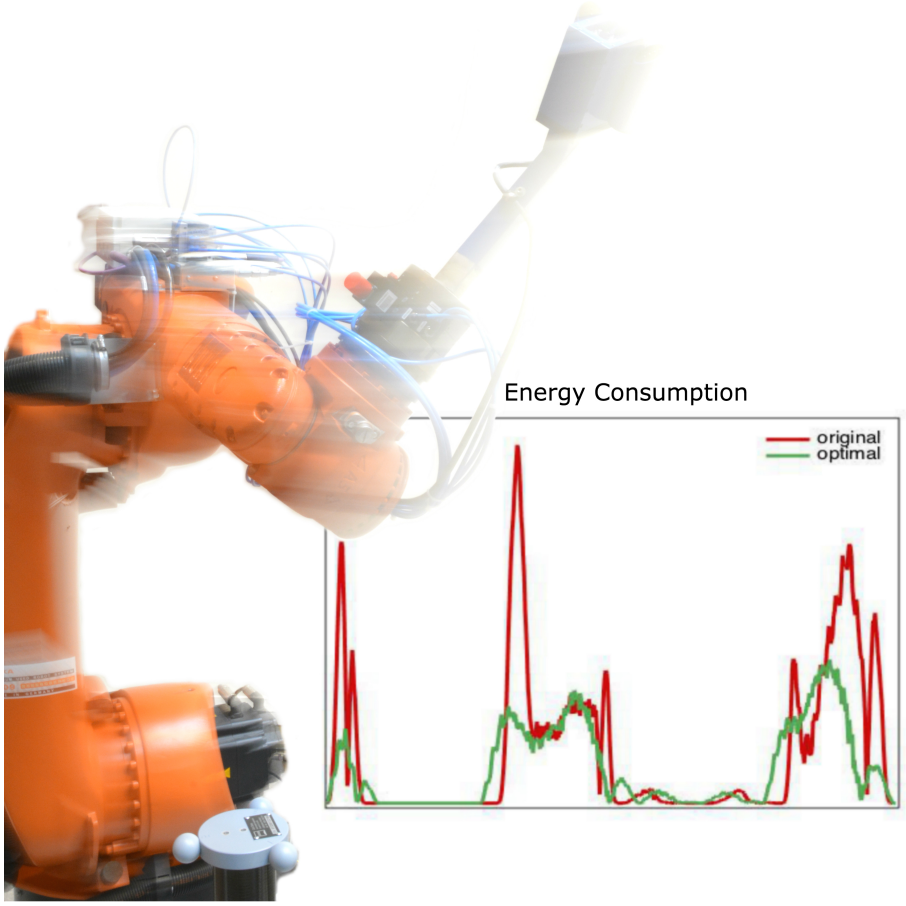
CHALMERS
UNIVERSITY OF TECHNOLOGY



Experimental evaluation of energy optimized trajectories for industrial robots

Master's thesis in Systems, Control and Mechatronics

EMMA VIDARSSON

Department of Signals & Systems
Chalmers University of Technology
Gothenburg, Sweden 2015

Master's thesis EX025/2015

Experimental evaluation of energy optimized trajectories for industrial robots

EMMA VIDARSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Signals & Systems
Division of Automatic Control, Automation and Mechatronics
Chalmers University of Technology
Gothenburg, Sweden 2015

Experimental evaluation of energy optimized trajectories for industrial robots
EMMA VIDARSSON

Examiner: Bengt Lennartsson, Department of Signals & Systems

Master's Thesis EX025/2015
Department of Signals & Systems
Division of Automatic Control, Automation and Mechatronics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: The mechanical power for an original and an optimized trajectory

Typeset in LaTeX
Gothenburg, Sweden 2015

Experimental evaluation of energy optimized trajectories for industrial robots
EMMA VIDARSSON
Department of Signals & Systems
Chalmers University of Technology

Abstract

This thesis presents a method reducing the energy consumption in industrial robots up to more than 30%. The reduction is possible by the use of a smart optimization algorithm tuning the individual robot motions and improving the coordination of multiple robots.

To increase the sustainability of the automated manufacturing industry, the large energy consumption in industrial robots must be reduced. Various approaches, such as new control systems, lower robot weight, DC-based architectures and smart sleep mode, are being evaluated today. However, with over 1.3 million industrial robots operating worldwide, a solution for already existing robots is also desirable. The challenges are to develop an energy-optimized solution and to integrate it in existing robots.

The experiments and evaluations in this thesis have been conducted on real industrial robots available in the robotic laboratory at Chalmers University of Technology. Based on the current robot behavior, various approaches for reducing the energy consumption of predefined trajectories have been evaluated. The results show that the use of an optimization algorithm for minimizing the squared acceleration can reduce the energy consumption in a robot significantly. When coordinating multiple robots the reduction can be increased even further, by the use of a smart zone-booking system reducing waiting time.

Furthermore, tools for measuring robot performance and for implementing and executing optimized trajectories in the robots have also been developed. The latter is an important contribution towards integrating the final optimization strategy in existing robots.

The resulting energy reduction of over 30% indicates, along with the execution method, that a fully integrated solution is feasible and likely to provide satisfying results.

Keywords: Industrial robots, energy consumption, power consumption, energy optimization, path planning, implementation method

Acknowledgements

I would like to thank everyone who has contributed to this thesis.

First of all, I would like to thank my examiner, Bengt Lennartsson, for letting me be a part of a project, as exciting and rewarding as this one. I appreciate your council, opinions and never failing positive attitude. I would also like to thank Oskar Wigström and Kristofer Bengtsson for their support and engagement. Oskar, thank you for getting me started and for all your help with the optimization. The fuzzy and deep conversations about happiness and what is important in life has been a most welcomed break in the work. Kristofer, thank you for proofreading the thesis and for always being available for discussions and questions. Your inputs have been of great value.

Thank you also to the employees at KUKA Gothenburg for the support and contribution towards this thesis.

Furthermore, I would like to direct a special thank you to Sarmad Riazi for the endless hours spend on the optimization. You get a golden star for your customer service but even more so for your sincere concern and kind personality.

I would also like to express my gratitude to all the friendly people at the Department of Signals & Systems, contributing to a relaxed and inspiring work environment and joyful coffee breaks. It has been fun. A special thank you to Linnéa Ahlman for being the best of friends, and also during this time a co-worker, conveniently located two doors down the hall to disturb in time and no time. I apologies for the distraction but truly appreciated the company.

Finally I would like to thank my family for always being there. You are the reason I am who I am. You are my rock and my inspiration. I love you.

Emma Vidarsson, Gothenburg, June 2015

Glossary

Cartesian Space	Space describing position and rotation of objects in global x-, y- and z-coordinates.
End-effector	The outermost part of the robot arm used to attached tools for part manipulation.
Joint space	Space describing position and rotation of objects in joint angles.
Multi-robot system	A system of robots containing multiple robots and other machines and peripherals.
Path	A geometric description of motion only denoting points in space
Pose	The position and orientation of the robot's end-effector
Posture	The angles of all joint in one instance determines the posture of the robot in that instance.
Trajectory	The path motion, defining the velocity and acceleration of each joint in each point of the path
Work cell	The area within the robot operates, containing the robot and other peripherals such as, other machines, static obstacles and safety environment.
Workspace	The area defined by all points in space reachable by the end-effector of the robot.

List of notations

Symbols

θ	angle
ω	angular velocity
$\dot{\omega}$	angular acceleration
$\ddot{\omega}$	angular jerk
E_m	mechanical work
h	sampling time
I	electrical current
$I_{log}^{\%}$	percentage electrical current
P_d	heat dissipated power
P_m	mechanical power
S_{E_m}	mechanical work reduction
S_{W_d}	heat dissipated savings
W_d	heat dissipated energy

Abbrevations

CIRC	Circular motion
DOF	Degrees of Freedom
KRL	KUKA Robotic Language
LIN	Linear motion
NLP	Non-Linear Programming
PLC	Programmable Logic Controller
PPO	Pick-and-Place Operation
PTP	Point-to-Point motion
SP	Sequence Planner
TCP	Tool-Center-Point

Contents

1

Introduction

In the manufacturing industry much manual work has been, and still is, replaced by industrial robots due to their repeatability, accuracy, speed and efficiency. As the technology has evolved the industry has become more automated to provide efficient, adaptive and evolving production lines. Unfortunately industrial robots are very energy intensive and their presence has a negative impact on the factories sustainability. Much effort is therefore directed towards designing more energy efficient manufacturing systems.

1.1 Background

A more effective and adaptable but yet sustainable manufacturing industry can be reached through joint effort from both the industry and the academic world. One of those efforts is the European research project AREUS (Automation and Robotics for European Sustainable manufacturing) [1]. The project aims to develop sustainable solutions for industrial robotics while increasing energy efficiency and reducing cost. It can be divided into four major work packages.

- WP1: Revolutionary energy reduction technologies.
- WP2: Eco-efficient design and simulation tools.
- WP3: Sustainable path and operation optimization.
- WP4: Life cycle sustainability assessment .

In short, WP1 aims to develop a new bi-directional electrical power supply to be able to exchange, harvest, store and recover energy at the factories. WP2 aims to create tools that consider sustainability when designing robotized automation by integrating advanced mathematical models for simulations, optimization and control design. WP3

aims to create tools to optimize the complex multi-robot cells and WP4 aims to assess the automation systems life cycle [1].

Ten major partner organizations from six European countries are involved in the project. The Automation research group at the Department of Signals and Systems at Chalmers University of Technology is one of the organizations and is responsible for WP3. In this package the group is focusing on developing algorithms and tools to optimize an industrial working cell containing robots, machines and other moving parts such as conveyor bands. Through advanced scheduling optimal operation sequences for robots and other moving devices can be determined. Areas multiple robots, machines and other moving parts can access, so called shared zones, are generated and shared automatically between robots and all moving devices within the cell to avoid collisions. Furthermore, the start times and initial motion for idling machines are optimized as well as the motion of each robot on an individual level.

1.2 Previous work

The aim of a sustainable manufacturing industry through the development of more energy efficient industrial robots has been approached by both the robot vendors and the end users. According to Paryanto et. al, [2], the reduction of energy consumption in industrial robots can be achieved at different stages in the development of manufacturing systems: during production planning, commissioning processes and process optimization stages. Production planning improvement includes optimizing the scheduling and choosing robots with low energy consumption rates [3], while improvements during the commissioning process are obtained by reducing idle and waiting time in the coordination of robots. Process optimization usually includes implementation of optimal trajectories.

Given the large area of interest, much research has been conducted within all three stages. An overview of various methods for a more energy efficient use of industrial robots can be found in [4]. The development of industrial robots with low energy consumption rates and more efficient engines is well investigated in [5], [6] and [7]. The scheduling of operations can be optimized on an individual level where the sequence of operations for one robot is to be determined, as in [8], or on a higher level in the synchronization of multiple robots, moving machines and other parts within a robotic work cell. This kind of scheduling usually considers the cycle time of the system, as in [9] and [10]. It can also consider the reduction of velocity and acceleration of the robots by the use of idle time, as in [11], but here with no consideration to the energy consumption. Scheduling directed more towards reducing the energy consumption can be found in [12] and [13], where also the reduction of idle time is considered as a part of the scheduling. Trajectory planning strategies are usually directed towards finding a time-optimal solution as in [14]. Development of energy optimized trajectories can be found in [15], [16] and [17].

Given the intense research within the field of manufacturing industry new more energy efficient and sustainable systems are being developed. However, in addition to the development of new systems, improvements of existing systems and robots are also important. According to the study World Robotics 2014, conducted by the International

Federation of Robotics (IFR), there were between 1.3 and 1.6 million industrial robots operational in the world by the end of 2013 [18]. Considering the 10-15 year long lifespan of an industrial robot it is also desirable to find a solution for reducing the energy consumption of already existing robots.

One possible alternative for energy reduction in existing robots is trajectory optimization. Most trajectory optimization though, requires the freedom of recreating the path of the robot. Such requirements are not desirable for optimization in existing robots. Trajectory generation in robots is usually created to maximize production rate while bounded to take surrounding obstacles into consideration, as well as other robots and machines moving in close proximity. The procedure to create and coordinate operations in an industrial setting is therefore often complex and time consuming. A solution reducing the energy consumption in existing robots should therefore be created with respect to already existing configuration, concerning path and execution time of the robot, to avoid collisions with objects in the robots proximity and to minimize extra work. Also, allowing the operator to create trajectories in a way familiar to them instead of learning new techniques is positive, with respect to time and cost.

1.3 Challenges and objectives

To design and develop an energy optimization strategy, possible to implement in existing robots and integrate with its technology, is a difficult challenge. It is desired to develop a strategy allowing for rapid energy optimization on existing trajectories without changing the configurations. Since detailed robot models require computationally heavy optimization procedures it is important to make smart simplifications and develop innovative tools and algorithms to achieve the desired result. The Automation research group at Chalmers University of Technology has access to an industrial robot making it possible to evaluate trajectory planning strategies through live experiments. To evaluate the trajectories, a method for capturing the energy consumption is necessary. Furthermore, the finally developed optimization strategy must be implemented and integrated in existing robots, desirably by the use of existing trajectory planning tools available in the robot.

Considering the research challenges, the purpose of this master thesis is to experimentally evaluate industrial robots with the aim to better understand the energy consumption. Different trajectory planning strategies are tested and compared to determine important criteria and find a sufficient strategy to minimize the energy consumption. There is also an interest in implementing an optimization strategy in an industrial setting using existing trajectory planning tools, why splines are investigated as a possible method. The results will give a better understanding of the energy consumption in industrial robots and how to reduce it to improve the sustainability of the manufacturing industry. The main objectives of this thesis are

- Create an experimental setup, enabling evaluation of trajectories and collection of

data, as described above

- Experimentally evaluate the trajectory planning strategies available in the robot today
- Implement and evaluate optimization strategies to determine optimization criteria reducing the energy consumption
- Investigate the possibilities for the integration of an optimization strategy in existing technology

1.4 Contribution

This thesis resulted in the following contributions, achieved in collaboration with the Automation research group at Chalmers University of Technology.

- The verification of a method reducing the energy consumption in industrial robots up to more than 30%. The reduction is possible by the use of a smart optimization algorithm tuning the individual robot motions and improving the coordination of multiple robots.
- The development of tools for measuring robot performance and for implementing and executing optimized trajectories in the robots. The method for implementing and executing the optimized trajectory is an important contribution towards integrating the final optimization strategy in existing robots.

1.5 Delimitations

The desire to optimize the trajectory with respect to energy consumption is limited by some constraints and requirements. The aim is to develop a technique for optimizing an existing trajectory, requiring only limited information and without altering the configurations. The path and the execution time have to remain undisturbed as well as the synchronization between joints. This implies that only the velocities and accelerations along the given path are being altered to achieve the optimization. This limits the technique to mainly be applicable on operations where a specific constant velocity is not crucial. To limit the work, given the time frame, the thesis is therefore focusing on material handling operations. The majority of motions in material handling operations are point-to-point why the initial work, investigating the behavior of the robot, is focusing on point-to-point motions and the optimization on pick-and-place operations. Furthermore the motions performed in the experiments are chosen due to the position of the robot in the laboratory in such manner to minimize the risk of accidents or collision with other objects as the robot and the laboratory is an area shared by many students and projects.

1.6 Structure of the report

Each chapter in this thesis is initiated with a brief introduction of the content, and ended with a short summary concluding the main contributions. Following this introduction to the subject and description of the challenges, objectives and contributions, is Chapter 2, giving a brief review of the fundamental principles concerning robotics and robotic programming. Chapter 3 describes the theoretical aspects of the optimization algorithm and Chapter 4 derives an expression for the energy consumption in industrial robots. In Chapter 5 the working procedure is explained along with the experimental setup. The chapter describes the development of tools needed to implement trajectories and the programs used to collect and analyze data. In Chapter 6, 7 and 8 the evaluation of the robot, the optimization and the splines as an implementation tool are described respectively, along with the results. The last chapter is dedicated the discussion of the results. Here, conclusions are drawn and proposals for future work are given.

2

Robots and Robotic Systems

To understand the work in this thesis some basic knowledge of robotics and robotic programming is required. This chapter briefly describes the fundamental principles concerning industrial robots and robot programming. It mentions the kinematics and dynamics of a robot, explains the differences between a path and a trajectory, and describes the programming of robots. Finally, it describes the setup and principles of multi-robot systems.

2.1 Robot manipulator

The earliest robots for industrial applications were developed in the 1960s [19]. The technology has, from then, greatly progressed and can now be said to have reached mature levels. Through the automation of processes the technology is not only trying to replace human operators in the execution of operations but also in the intelligent processing of information. Automation is the synthesis of mechanical technology used in the industry and computer technology allowing information management [19]. The industrial robot is, due to its programmability and flexibility, an essential component in the programmable automated systems growing larger within the industry.

By ISO 8373 [20] an industrial robot is defined as a re-programmable, automatically controlled, multipurpose manipulator programmable in three or more axes. A robot manipulator is classified as a robot with a fixed base (compared to a mobile robot with a mobile base) and consists of a number of rigid bodies referred to as links. These links are connected to one and another by joints that are either prismatic or revolute. Prismatic joints creates a relative translational motion between two links where revolute joints creates a relative rotational motion [19]. The displacement is referred to as joint offset and joint angle respectively.

Manipulators can be classified as Cartesian, cylindrical, spherical, SCARA or anthropomorphic depending on the type and sequence of the degrees of freedom (DOFs).

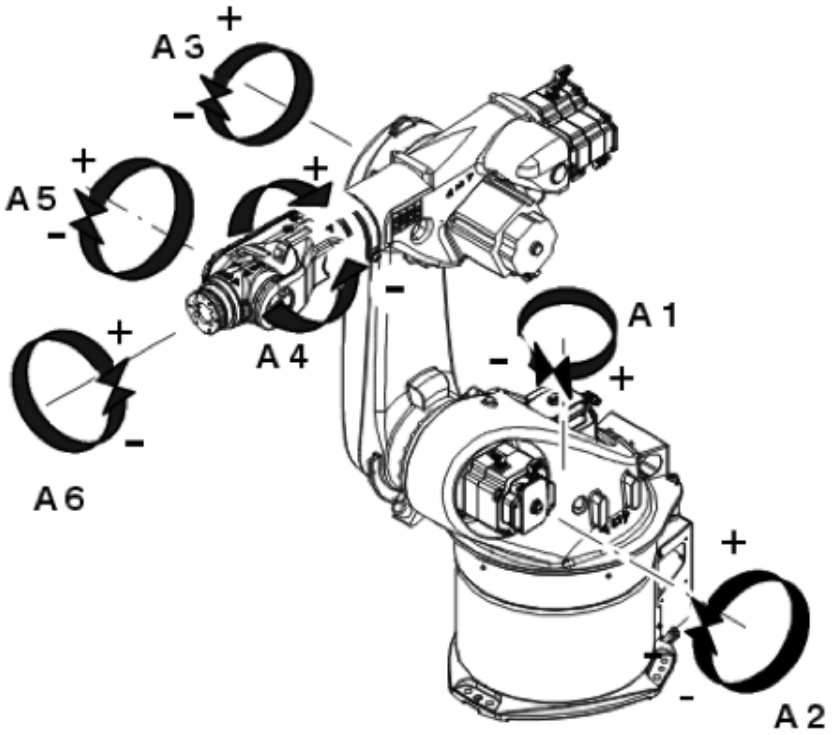


Figure 2.1: An anthropomorphic robot with the six joints described.

For a manipulator to perform an arbitrarily positioning and orientation of an object in a three-dimensional space six DOF are required, three to position the object and three to orient the object. The links in a manipulator can either form an open or closed kinematic chain. In an open kinematic chain there is only one sequence of links connecting the two end points compared to a closed kinematic chain where some sequence of links form a loop. In an open kinematic chain each joint provides the robot with one DOF.

The most common class in industrial settings is the anthropomorphic, because its high dexterous makes it useful in a wide range of application. The anthropomorphic robot contains six revolute joints in an open chain and can be divided into three parts, an arm, a wrist and an end-effector. By the use of a tool attached at the end-effector the robot can manipulate objects. Because of the similarity with a human arm the second joint is called the shoulder joint and the third joint the elbow joint. Its position and orientation of an object in space is only limited to the workspace of the robot. The **workspace** is the area defined by all the points reachable by the robots end-effector. An anthropomorphic robot can be seen in Figure 2.1.

The use of industrial robots in the manufacturing process industry can be divided into three major categories [19]:

- material handling

- manipulation
- measurement

Material handling refers to all the moments in a manufacturing process where material and parts are moved from one place to another for storage, manufacturing, assembly and package. A robot is ideal for pick and place operations such as:

- palletizing
- part sorting
- loading and unloading
- packing...

In a similar fashion manipulation is referring to all the moments during the manufacturing process where parts and material are altered, added and finally turned into a finished product. Typical manipulation applications where the robot can perform such tasks are:

- painting and coating
- arc and spot welding
- gluing and sealing
- sqrewing, wiring and fastening ...

Finally, within the process of manufacturing there are several moments where the parts and material are examined to ensure correct manufacturing and product quality. These kind of measurements can be performed by manipulators and some examples of applications are:

- object inspection
- operation supervision
- contour finding...

Independent of the task performed by the robot, the motions, as well as the present mechanisms and dynamics, have to be described and represented by some means.

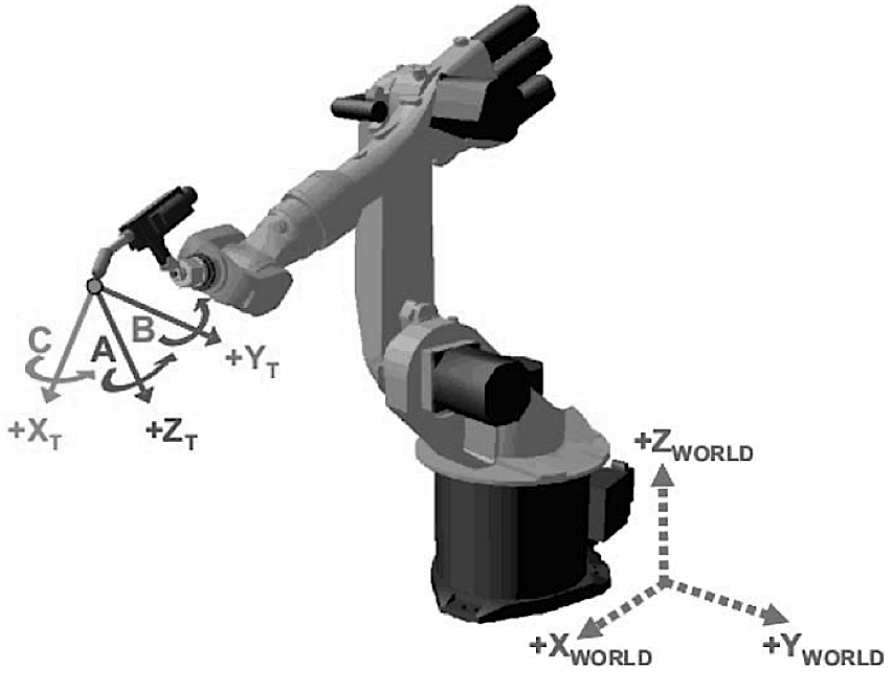


Figure 2.2: The base and tool frame of a robot. Photo: KUKA Robotics

2.2 Positioning and orientation

To be able to manipulate and handle objects it is necessary to describe the position and orientation of both the object and the robots end-effector in space. Assuming there exists a universal coordinate system to which everything can be referred to, and an object is provided a coordinate system or a frame, the position and orientation of the object can now be described with respect to the universal coordinate system [21]. It might be desirable to describe the position and orientation of the object with respect to some other reference system and since any other frame also is described with respect to the universal coordinate system it is possible to recalculate the position and orientation of the object with respect to the new reference frame. This is called transforming, mapping or changing the description of the object [21]. The tool frame and base frame of a robot can be seen in Figure 2.2.

The position and orientation of a robot's end-effector is usually referred to as the **pose**, [19], and can either be described in **Cartesian space** or in **joint space**. In Cartesian space the pose of the end-effector is described by its position in the global x-,y, and z-coordinates as well as its rotation around the global x-, y- and z-axis. In joint space the pose of the end-effector is described by the angles on each of the joints in the robot. [22]. The angles of each joint in the robot also determine the **posture** of the robot [19].

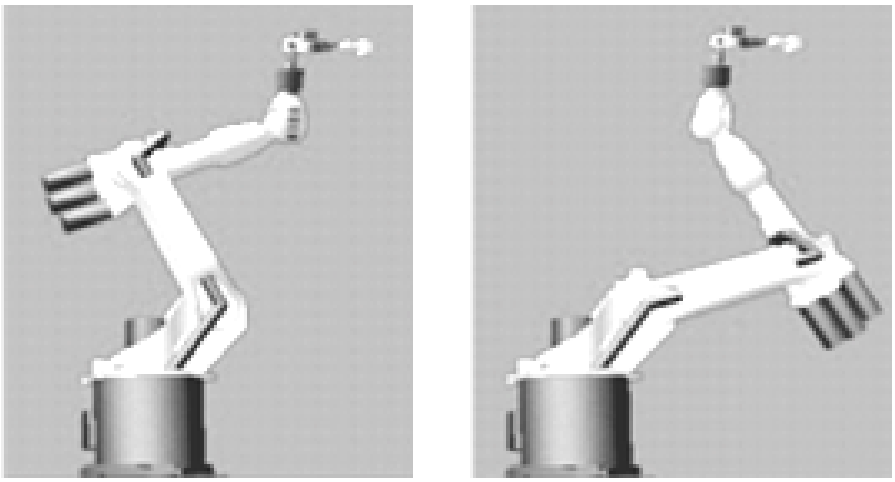


Figure 2.3: Two robot postures giving the same position and orientation of the end-effector. Photo: KUKA Robotics

2.3 Kinematics

For an industrial robot the relationship between the joints position and the end-effectors pose is described through its kinematics [19]. The kinematics of a robot refers to all the geometrical and time-based properties of a motion, such as position, velocity, acceleration and all higher order derivatives of position, while ignoring the forces causing it. The relationship between the joint motions and the motions of the end-effector is described through the Jacobian matrix and referred to as the robots differential kinematics [19]. The kinematic relationships can be divided into two categories, the direct, or forward, kinematics and the inverse kinematic.

The **direct kinematics** describes the end-effectors pose as a function of the joint angles, the posture of the robot, and is also known as transforming from joint space description to Cartesian space description [21] and is relatively straightforward. Describing the end-effectors tool frame relative the base frame is an example of a direct kinematic problem.

The **inverse kinematic** problem consist of computing the joint angles given the end-effectors pose. This is usually more complicated and might not provide a unique solution as a robot can reach the same end-effector pose through different joint angles and postures. Figure 2.3 shows one example of two different robot postures giving the end-effector the same position and orientation. The inverse kinematic problem might also not have a solution and the robots workspace is defined by the existence or nonexistence of an inverse kinematic solution [21]. If no solution exists the given end-point position lies outside of the robot's workspace.

2.4 Dynamics

A dynamic model of a robot describes the relationship between a motion, position, velocity and acceleration, and the forces required to cause the motion. Parameters such as mass of the robot, the inertia of different components, the geometry and gear characteristics (gear torque) as well as external forces such as gravitation takes into consideration and determines the dynamic performance of the robot [21]. The dynamic model is of great importance when designing a control system. In a similar way as the kinematics, there is a direct and inverse problem with the dynamic model. The inverse dynamic calculates the force and torques in each joint related to the position, velocity and acceleration while the direct dynamics calculate the motion with respect to internal or external forces and the torques acting on it [21]. In a direct model calculation the current joint position, velocity and torques along with active forces are entered in to the model to output the resulting joint acceleration.

2.5 Path and trajectory

The terms path and trajectory are commonly incorrectly used as synonyms. Since both are being used frequently throughout the thesis the difference between the terms are to be explained to avoid confusion. A **path** is a geometric description of motion and only denotes points in space, Cartesian or joint, that the end-effector has to move through when executing the assigned task. In each point the pose of the end-effector and the posture of the robot is defined.

A **trajectory** is the path motion and defines the velocity and acceleration of each joint in each point of the path [19]. The path can be seen as the road the car has to follow and the trajectory defines how the car is traveling along that road, where it slows down or speeds up.

2.6 Motion types

Both path and trajectory are crucial when programming robots. In principle the description and constraints of a path along with the dynamic constraints can be seen as the inputs to the trajectory planning algorithm, while the outputs are a time sequence for the end-effectors position, velocity and acceleration. The function of a trajectory planning algorithm is to create a smooth trajectory defining the motion of the robot without violation of any constraints.

In an operators point of view, a trajectory is created by defining one or more motions after each other. The motions can be of different type but the starting point of one motion is always the end point of the previous motion [23]. Depending on the desired application of a robot its trajectory is programmed differently. Some general motion types are:

- Point-to-point (PTP)

- Linear motion (LIN)
- Circular motion (CIRC)
- Spline motions (SPLINE)

Linear, circular and spline motions are all continuous path motions while point-to-point motions are not. Positioning motions, such as material handling operation or spot welding, are only considering the pick-up and the drop-off positions why these operation, called pick-and-place operations (PPO), usually are constructed with mainly point-to-point motions. On the contrary, manufacturing operations are often very path and velocity specific, why they are created using continuous path motions.

2.6.1 Continuous path motions (LIN, CIRC)

In a continuous path motion the path is defined in Cartesian space. When creating a linear (LIN) and circular (CIRC) motion the robot guides the tool-center-point (TCP) at a defined velocity along a straight and circular path between the start and end point respectively [23]. A linear path is defined by a start and end point while a circular path also needs an auxiliary point as seen in Figure 2.4 and 2.5.

When creating continuous path motions, problem with singularities and workspace limitations may arise. Even though a start and an end point lies within the workspace there might be intermediate points that do not. It would, as an example, be impossible for the robot to move in a linear path between two points positioned on either side of the robot since the body of the robot lies in its way. Singularities arise when the Jacobian, specifying the mapping of velocity in joint space to Cartesian space, is not invertible, that is, the recalculation from Cartesian coordinates does not provide one unique solution in joint space [23]. Two examples of postures of the robot where singularities arise is seen in Figure 2.7 and 2.8. Figure 2.7 shows an overhead singularity where the wrist point (the center of joint 5) is positioned directly in a vertical line of joint 1, preventing joint 1 to be uniquely determined. In Figure 2.8 an arm extension singularity is shown. The wrist point lies in the direct extension of joint 2 and 3 and the TCP is located in the outermost region of the workspace. Translating the position into joint space give unique joint angles but low Cartesian velocities provide very large joint velocities for joint 2 and 3 causing trouble [23].

2.6.2 Point-to-point motion

In a point-to-point (PTP) motion the robot moves the TCP in joint space from a start point to an end point without considering the path in between. Instead a PTP motion is time-optimized such that the robot moves the TCP along the fastest path between the start and end point as seen in Figure 2.6. As the motions of the robot joints are rotational, curved paths are executed faster than straight paths and the path, velocity and acceleration in Cartesian space is said to be uncontrollable [24].

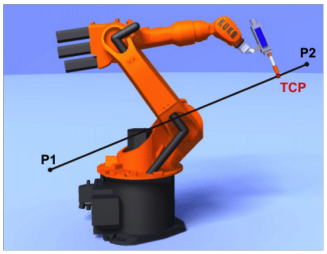


Figure 2.4: Linear motion. The robot guides the TCP with the predefined velocity along a straight line to the finishing point. Photo: KUKA Robotics

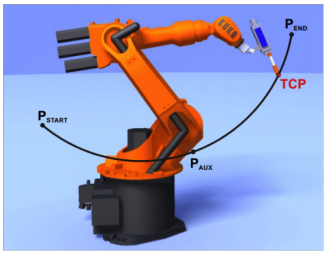


Figure 2.5: Circular motion. The robot guides the TCP with the predefined velocity along a circular path to the finishing point through the auxiliary point. Photo: KUKA Robotics

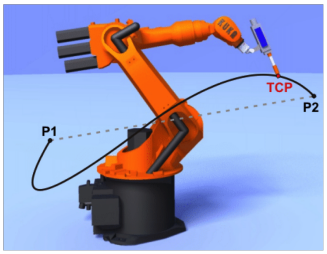


Figure 2.6: Point-to-point motion. The robot guides the TCP to the finishing point along the fastest path. The path is said to be non deterministic since it can not be predicted. Photo: KUKA Robotics

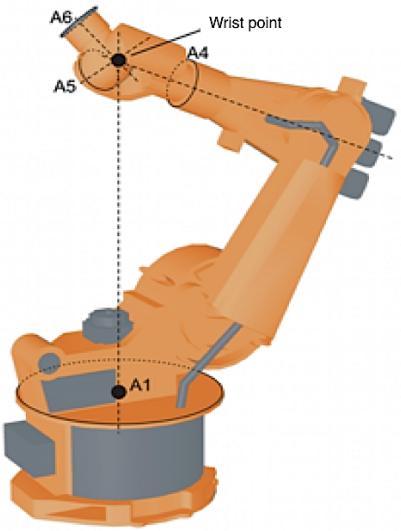


Figure 2.7: Overhead singularity. Photo: KUKA Robotics

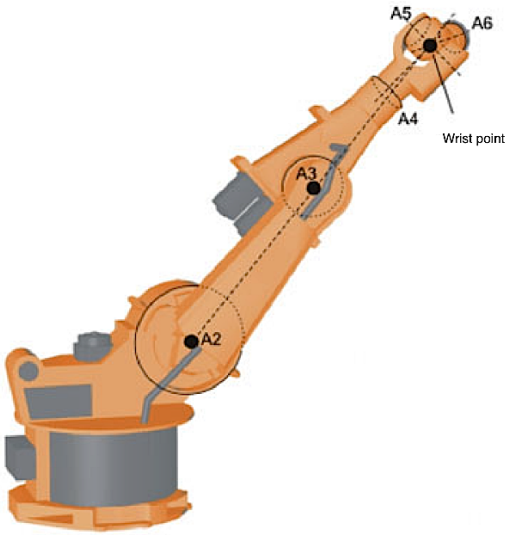


Figure 2.8: Arm extension singularity. Photo: KUKA Robotics

2.6.3 Continuous and non-continuous points

All motion types can be defined as continuous or non-continuous. A continuous point is a point the robot only have to come within a certain distance of before it can move on towards the next point where as a non-continuous point has to be reached before moving on toward the next point. The distance of which the robot has to come within for a continuous point is predefined in the robot. Continuous points can for an example be used as via-points to avoid collisions and to create linear motions along a curvature in continuous path planning as in Figure 2.9. For continuous points with PTP motions the side of which the TCP will pass the continuous point can not be determined [25].

2.6.4 Spline motions

For newer software a complex curved path can be created using splines instead of continuous LIN and CIRC. The spline motion, of Cartesian motion type, shows clear advantage over LIN and CIRC. For instance, for continuous LIN and CIRC motions the path is defined by approximated points not located on the path and the approximation is altered for different velocities, why generating the desired path is complicated and time consuming. For splines, the points are located on the path making it easier to generate and the path remains the same for different velocities. The defined velocity is better maintained for spline motions and the path remains the same irrespective of possible overwrite setting, velocity and acceleration [25]. Figure 2.9 and 2.10 show the differences in programming a complex path with continuous LIN and CIRC compared to spline blocks.

The spline motion will not be explained in detail but one of the most useful and versatile spline motion worth mentioning is the spline block. Each motion in the spline block is called spline segment and is similar to other motions. The differences lie in the execution of the trajectory. The robot plans and executes the whole block as one single motion instead of only planning three lines ahead and executing each single line as an individual motion, as in traditional trajectory planning using sequences of LIN, CIRC or PTP [25].

2.7 Motion Programming

To implement the different motions and create trajectories a robot programming language is used [21]. Most robot systems are equipped with an interface, including an inline form, designed to simplify the path- and trajectory planning for the operator. This operator interface is user friendly and equipped with predefined commands and functions which allow the user to create a trajectory by only defining a few settings such as motion type, velocity and acceleration for different sections of the trajectory. An operation is usually programmed by defining points along the path. A pick-and-place operation can, for example, be created as following:

Define Points P1-P6:

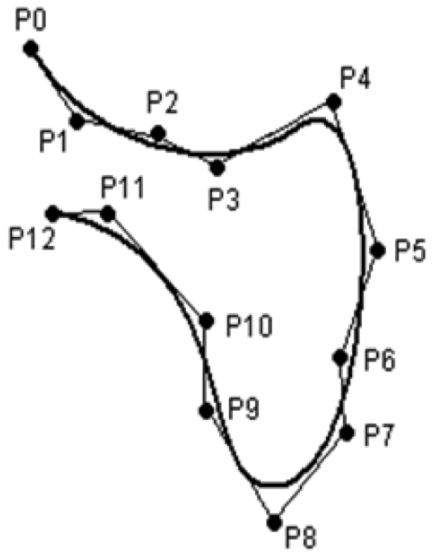


Figure 2.9: A complex path programmed using continuous LIN and CIRC motions. Photo: KUKA Robotics

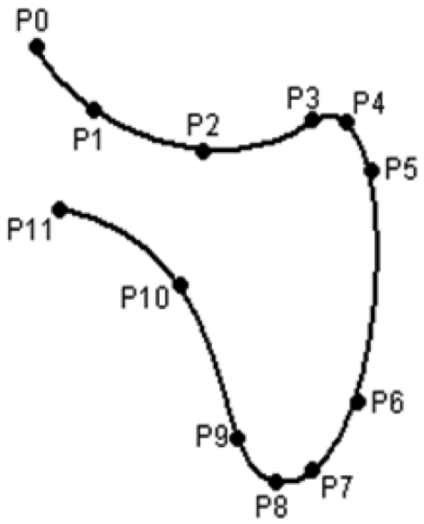


Figure 2.10: A complex path programmed using spline block segments. Photo: KUKA Robotics

P1 : Start at a secure location

P2 : 10 cm above bin A

P3 : At position to pick part in bin A

P4 : Point in space between bin A and B to maneuver around obstacle

P5 : 10 cm above bin B

P6 : At position to place part in bin B

Define trajectory

1. Start in P1
2. Move to P2 - PTP, 100 % velocity
3. Move to P3 - LIN, 2 m/s
4. Close gripper
5. Move to P2 - LIN 1 m/s
6. Move to P4 - PTP continuously 100 % velocity
7. Move to P5 - PTP 100 % velocity
8. Move to P6 - LIN 1 m/s
9. Open gripper
10. Move to P5 - LIN 2 m/s
11. Move to P1 and start over - PTP 100 % velocity

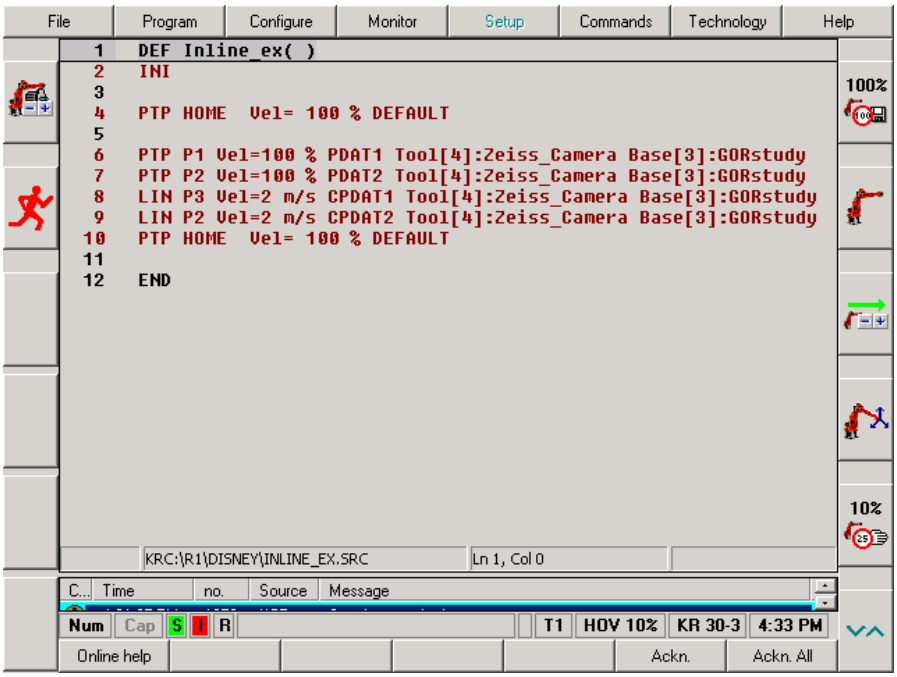


Figure 2.11: Coding the robot in the controller using the built in inline form. The motion setting line is shown where the movement, point, continuity, velocity and acceleration are defined.

At each point of the trajectory the motion type is defined as well as velocity, acceleration and other settings , such as whether the point is continuous or not. An example of a trajectory created using the inline form can be seen in Figure 2.11. For a more complex program where more flexibility and freedom is necessary the user can manually create the trajectory by using the KUKA Robotic Language (KRL) form. Here, each action has to be created manually by coding the desired behavior. A simple trajectory created using the KRL can be seen in Figure 2.12

2.8 Multi-robot systems

Independently of the area of use, an industrial robot is always interacting with, and have to consider, obstacles in its surrounding. There are both fixed obstacles, such as conveyor bands and drop off areas but most often also moving obstacles, such as material or parts being transported on the conveyor band or other robots working in the same area. The robot along with the obstacles in its surrounding and other peripherals such as the safety environment are, as mentioned earlier, referred to as a **robot cell** or a **workcell**. A system including multiple robots along with peripherals are, not surprisingly, referred to as a multi-robot system.

The coordination between individual robots as well as the coordination between robots and other machines and obstacles is most usually performed using a programmable

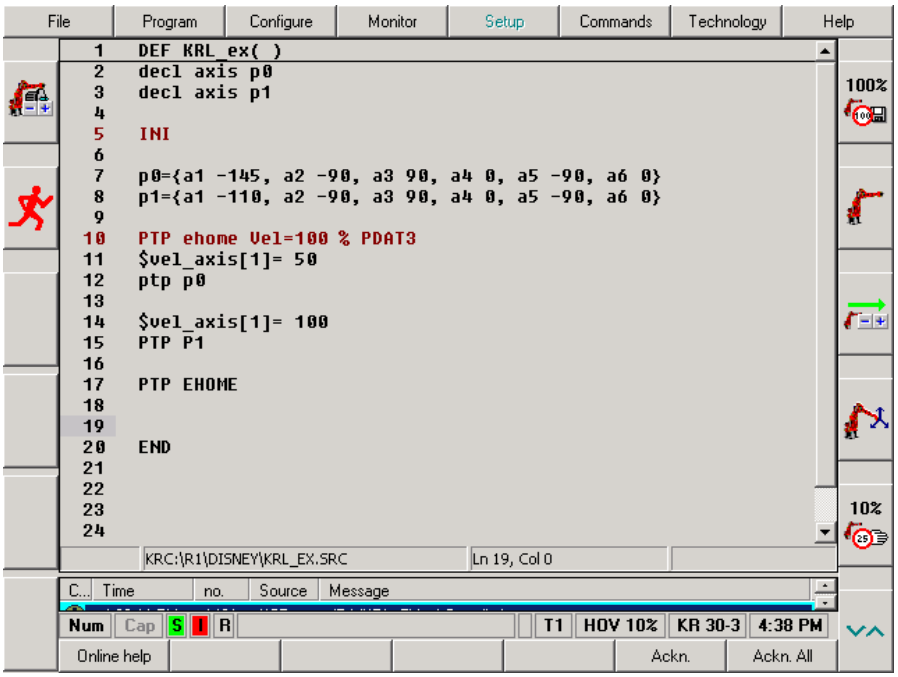


Figure 2.12: A simple example of coding the robot using KRL in combination with the inline form

logic controller (PLC). The interaction between robots, other machines and obstacles have to be programmed with regards to their positions in the cell but also with regards to the synchronization of motions and operations. The main objective of this coordination is to avoid collision between objects but also to determine work flow and sequence of operations [26]. To coordinate the robots and sequence of operation in a safe but yet efficient way is everything but simple.

2.8.1 Industrial working procedure

The traditional challenge for a robot cell in the industry is to perform as many operations as possible during a given cycle time. The coordination of machines and work flow can be determined in more than one way, but the state-of-the-art work procedure to achieve efficient multi-robot control system include three steps [26]:

- Define desired sequence of operation using CAD models
- Semi-automatically generate individual robot trajectories.
- Manual control code generation

Define desired sequence of operation

By using planning tools such as Process Simulate, Siemens PLM or Delmia Automation, Dassault Systems the whole multi-robot system is created as a detailed CAD model, including information and geometry of obstacles and moving equipment within the cell. The behavior of all equipment is defined in terms of tasks such as moving, placing and fixating operations. While some operations can be performed in an arbitrary order others have to be performed in a specific sequence to, for instance, ensure accurate assembly or packing.

Semi-automatically generate individual robot trajectories

For an individual robot, trajectory planning consist of creating a time-optimized trajectory along a collision free path, given static obstacles. For a multi-robot system when more than one robot is moving simultaneously and paths are intersecting, the velocities of each individual robot have to be tuned to avoid collisions between robots. Software such as Process Simulate, Siemens PLM and Delmia Robotic Path Planner, Dassault Systems are used to create the individual path planning as well as simulating the whole cell and generating the final robot control code. Unfortunately, little support is given for the tuning of velocity for intersecting trajectories why an accurate model and simulation of the whole system is critical to allow for easier manual tuning, including zone booking.

Manual control code generation

To obtain the desired sequence of operations, the start and coordination of different robot tasks have to be arranged appropriately. This coordination can be achieved through the PLC or through implementation using one of the robot controllers. To achieve more flexibility, a zone booking system can be introduced, allowing only one robot to enter a specific zone after booking it and being granted access. Such zone booking systems are mostly generated manually and therefore also only include basic functionality.

The complexity of creating efficient robot cells leave room for improvements and optimization and much research is focused on these areas. As this thesis will conclude, there are possibilities to reduce energy consumption within a larger system by optimizing the trajectories on an individual basis as well as by improvement of the zone booking scheduling.

2.9 Summary

This chapter introduces concepts important for the work in this thesis. The basic geometry and functionality of the robots are described. The positioning and orientation of the end-effector is defined as the **pose**, and the joint values also describe the **posture** of the robot. The position and orientation of the end-effector can be described in **Cartesian Space** by its position in, and rotation around, the global x-, y- and z-coordinates

or in **joint space** by the angle of each joint. Furthermore is the **workspace** of the robot defined by all points reachable by the end-effector. The **path** of the end-effector is a geometric description of motion and only denotes the points the robot has to move through. Each point along the path is also related to a specific posture of the robot. The **trajectory** defines the velocity and acceleration along the given path. Trajectories are created by defining a sequence of motions using motion types such as LIN, CIRC and PTP, each with different properties. The trajectories are created and executed using a robot programming language as such **KRL**.

In an industrial setting multiple robots usually work in close proximity of each as well as other obstacle and moving machines. This setup is referred to a **work cell** or **robot cell**. The coordination of robots and execution sequences, as well as the creation of trajectories, to achieve efficient multi-robot control systems can be determined through the state-of-the-art work procedure.

3

Optimization

The optimization of existing trajectories is depending on predefined configurations already implemented in the robot. This chapter describes the constraints limiting the optimization as well as the approach and optimization model used to reduce the energy consumption. For the optimization of a robot cell containing multiple robots the last section also describes the zone booking constraint used.

3.1 Optimization

With a better understanding for the motions of the robot, the next task is to find an optimization algorithm minimizing the energy consumption. It is desired to find an algorithm easy to implement and rapid enough to work online. The procedure includes acquiring the original trajectory from the manipulator, optimizing it with respect to the given constraints and feeding the optimized trajectory back to the robot. As mentioned before, the optimization is limited by two constraints.

- The execution time has to equal the execution time for the original trajectory.
- The original sequence of posture of the robot must not be altered along the path.

In further detail, this means the optimized operation has to be finished in the same cycle time as the original operation and move along the same path. The optimized trajectory is implicitly following the original path when fulfilling constraint two. The original operation defines the path and the trajectory by a sequence of robot postures between the starting and finishing point, sampled every Δh second, where Δh is a relatively short sampling interval. The optimized operation is then moving along that same path, with the same posture but with another time sequence. Basically the sequence of points along the path can be seen as a fixed input to the optimizer and the time to move between samples the degree of freedom. The relationship between the two trajectories can be seen as a mapping of the uniformly spaced original path onto the differently spaced

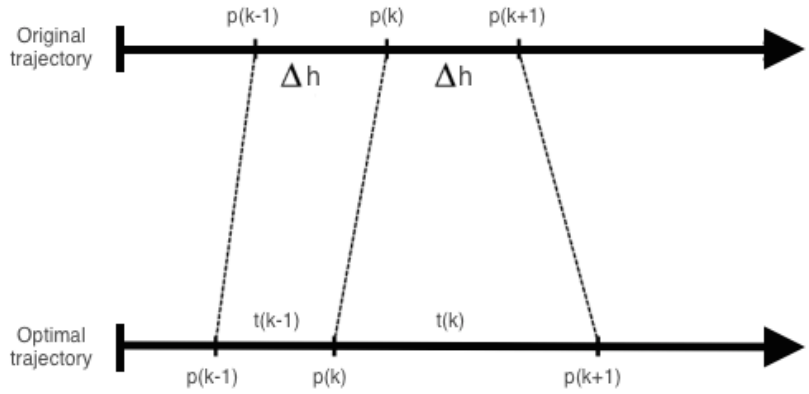


Figure 3.1: The mapping of original to optimal trajectory.

optimal path as in Figure 3.2. The spacing of the positions for the optimal trajectory is representing the new energy efficient trajectory. The optimal trajectory will have different velocity and acceleration in each position compared to the original trajectory, as seen in Figure 3.3 where the velocity is plotted as a function of position. Both trajectories consist of moving joint 1 from -90 to 0 degrees but with different velocity in each point. The path and total execution time are preserved but the velocity and acceleration along the path is different for the optimized trajectory.

The statement requiring the posture of the robot to be preserved along the path is indirectly fulfilled when the path is defined in joint space. In theory it means that the synchronization of joints have to remain the same. The original trajectory is divided into samples of joint positions with a fixed, Δh, sample time. The value of the joints in each sample defines the posture of the robot in that specific sample. The difference between two samples give a ratio between joints. This ratio must remain and is only allowed to be scaled. That is, if the robot reached a specific posture in two seconds with the original trajectory, the same posture have to be reached with the optimal trajectory as well, but it can happen after one or two and a half, or any other arbitrary time instants.

The optimization algorithm uses the original trajectory as an input and then minimizes the cost function quickly, using a non-linear programming (NLP) solver.

3.1.1 Modeling the optimization problem

Details about the optimization model is given in [27]. The model is generalized to handle multiple robots with different number of joints and separate paths and timespan. The constants and variables used to construct the model are organized in sets as follows:

- $\mathcal{R} = \{1,2,\ldots,r,\ldots,q\}$ is the index set for a set of robots where R_r denotes the r^{th} robot.

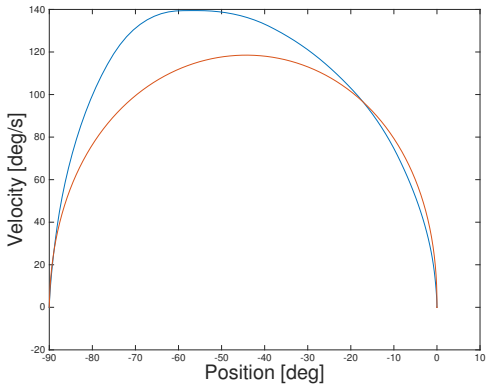


Figure 3.2: The velocity of the original (blue line) and optimal (red line) trajectory is plotted as a function of the position. The plot show how the path is preserved while the optimization alters the velocity, acceleration and jerk along the given path.

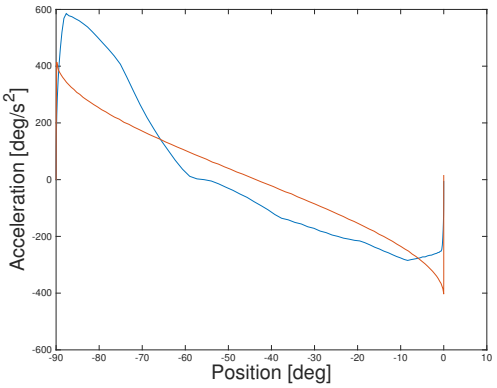


Figure 3.3: The acceleration of the original (blue line) and optimal (red line) trajectory is plotted as a function of the position. The plot show how the path is preserved while the optimization alters the velocity, acceleration and jerk along the given path.

- $\mathcal{J}^r = \{1,2,\ldots,j,\ldots,n_r\}$ is the index set for a set of joints in robot R_r
- $\mathcal{W}^r = \{w_1^r,w_2^r,\ldots,w_j^r,\ldots,w_{n_r}^r\}$ is a tuple of weights (costs) defined for each joint j in robot R_r.
- $P_r = \{1,2,\ldots,m,\ldots,l_r\}$ is the index set for the set of postures for the path of robot R_r.
- $p_m^r = (\theta_{1,m}^r,\theta_{2,m}^r,\ldots,\theta_{j,m}^r,\ldots,\theta_{n_r,m}^r)$ is the m^{th} posture for robot R_r defined as a set of n_r angular positions for n_r joints of robot R_r. $\theta_{j,m}^r$ is the angle of the j^{th} joint of robot R_r at the m^{th} posture along its path. It is assumed that each joint has its own polar coordinate system, thus the angle of each joint is measured relative to its own polar axis.
- $T^r = (t_1^r, t_2^r, \ldots, t_m^r, \ldots, t_{l_r}^r)$ is a set of time variables for robot R_r. The value of t_m^r determines the time when the robot will pass through the m^{th} posture in the corresponding path.
- $v_m^r = (\dot{\theta}_{1,m}^r, \dot{\theta}_{2,m}^r, \ldots, \dot{\theta}_{j,m}^r, \ldots, \dot{\theta}_{n_r,m}^r)$ is a tuple of velocity variables for joints in robot R_r at posture p_m^r. The value of $\dot{\theta}_{j,m}^r$ determines the speed at which the j^{th} joint of robot R_r passes the m^{th} pose.
- $a_m^r = (\ddot{\theta}_{1,m}^r, \ddot{\theta}_{2,m}^r, \ldots, \ddot{\theta}_{j,m}^r, \ldots, \ddot{\theta}_{n_r,m}^r)$ is a tuple of acceleration variables for joints in robot R_r at posture p_m^r. Values of $\ddot{\theta}_{j,m}^r$ determines the acceleration at which the j^{th} joint of robot R_r passes the m^{th} posture.

The optimization problem is then formulated as

$$\min \sum_{r\in\mathcal{R}} \sum_{j\in\mathcal{J}_r} \sum_{m\in P_r} w_j^r \ddot{\theta}_{j,m}^{r\,2} \tag{3.1}$$

subject to:

$$\sum_{m\in P_r} t_m^r \leq \lambda \qquad r \in \mathcal{R} \tag{3.2}$$

$$t_{m+1}^r - t_m^r \geq \epsilon \qquad r \in \mathcal{R}, m \in P_r \tag{3.3}$$

$$\dot{\theta}_{j,m+1}^r = \frac{\theta_{j,m+1}^r - \theta_{j,m}^r}{t_{m+1}^r - t_m} \qquad r \in \mathcal{R}, j \in \mathcal{J}_r, m \in P_r \tag{3.4}$$

$$\ddot{\theta}_{j,m+1}^r = \frac{\dot{\theta}_{j,m+1}^r - \dot{\theta}_{j,m}^r}{t_{m+1}^r - t_m} \qquad r \in \mathcal{R}, j \in \mathcal{J}_r, m \in P_r \tag{3.5}$$

$$\left|\frac{\ddot{\theta}_{j,m+1}^r - \ddot{\theta}_{j,m}^r}{t_{m+1}^r - t_m}\right| \leq k_j^r \qquad r \in \mathcal{R}, j \in \mathcal{J}_r, m \in P_r \tag{3.6}$$

$$|\ddot{\theta}_{j,m}^r| \leq b_j^r \qquad r \in \mathcal{R}, j \in \mathcal{J}_r, m \in P_r \tag{3.7}$$

$$|\dot{\theta}_{j,m}^r| \leq c_j^r \qquad r \in \mathcal{R}, j \in \mathcal{J}_r, m \in P_r \tag{3.8}$$

$$\dot{\theta}_{j,1}^r = \ddot{\theta}_{j,1}^r = 0 \qquad r \in \mathcal{R}, j \in \mathcal{J}_r \tag{3.9}$$

$$\dot{\theta}_{j,l_r}^r = \ddot{\theta}_{j,l_r}^r = 0 \qquad r \in \mathcal{R}, j \in \mathcal{J}_r \tag{3.10}$$

$$t_m^r \geq 0, t_m^r \in \mathbb{R} \qquad r \in \mathcal{R}, m \in P_r \tag{3.11}$$

$$\dot{\theta}_{j,m}^r, \ddot{\theta}_{j,m}^r \in \mathbb{R} \qquad r \in \mathcal{R}, j \in \mathcal{J}_r, m \in P_r \tag{3.12}$$

Assuming the acceleration is closely related to the energy consumption the objective function in (3.1) represents an approximation of the energy consumption. The weights, w_j^r, are used to tune the influence from each joints acceleration on the overall energy consumption. The constraint in (3.2) limits the execution time for the optimized trajectory to the original execution time, predefined as λ. (3.3) is a physical constraint forcing the trajectory to spend time moving between two samples. To avoid numerical instability ϵ should be assigned an small positive arbitrary number larger than 0. The constraint in (3.4) and (3.5) defines the velocity and acceleration between two samples respectively. The maximum jerk, acceleration and velocity for each joint, j, are bounded by the values of k_j^r, b_j^r and c_j^r in (3.6), (3.7) and (3.8) respectively. The true values of k_j^r, c_j^r and b_j^r can not easily be obtained as they depend on the configuration, load and posture of the robot but given the original data rough approximations can be calculated. It is assumed that the trajectories start and ends in a stationary position with no velocity or acceleration present, represented by the constraints (3.9) and (3.10). Finally the domain for time, velocity and acceleration are defined in (3.11) and (3.12).

The algorithm above is used for the work of optimizing existing trajectories in the thesis. The implementation of the trajectories is described in Chapter 5.

3.2 Shared Zone Constraints

In the coordination of multiple robots moving within the same area, a set of shared zones is introduced to represent areas available to several robots. Collisions are then avoided by the use of a booking system only allowing one robot to access the shared area at a time and only after requesting and being granted access. If another robot is occupying the zone the robot requesting the zone has to wait. The scheduling and coordination of accessibility is an important part in the optimization of multiple robots. The optimizer decides in what order the robots should enter the zone. With this sequence specified, constraints necessary in the optimization of the trajectories are easily formulated. Giving an example with two robots, each needing to enter a shared zone once, the posture time index of the first robot leaving the zone and the posture time index of the second entering the zone are then designated by f_{leave} and s_{enter} respectively. There are two possible scenarios where either robot one enters first and robot two thereafter, or the opposite where robot two enters first. The corresponding time constraint then become

$$t^1_{s_{enter}} - t^2_{f_{leave}} \geq \epsilon \quad \text{or} \quad t^2_{s_{enter}} - t^1_{f_{leave}} \geq \epsilon \tag{3.13}$$

stating that the first robot have to leave the zone ϵ time instances before the second robot enters, where ϵ is a small positive number providing a safety margin to the transition.

3.3 Summary Optimization

The chapter describes the optimization model used to generate energy optimized trajectories given an original trajectory. The generation of an optimized trajectory is limited by the existing original configurations. The original path and execution time is not allowed to be altered. The optimization can be seen as a mapping of the sequence of postures, sampled with a fixed time interval, onto a time sequence of not fixed time intervals. The energy optimal time sequence is generated by the use of an NLP solver.

For a setup with multiple robots with shared zones, a constraint is set to define the time a robot is allowed to enter the shared zone.

4

Energy calculations

This chapter derives the power and energy consumption in electrical engines and briefly describes the different contributions to the total energy consumption.

4.1 Electrical motors

According to [7] the industrial sector consumes the largest amount of energy worldwide with the major contributor identified as industrial motors and, more specifically, electrical engines. With the growing concern about energy consumption and its impact on the environment more effort is put into improvements of this situation. Improvements can either be achieved by reducing the total energy or increasing the efficiency by reducing the energy losses. An electrical motors function is to convert electrical energy to mechanical energy necessary to perform a task. The efficiency of the engine determines how well the engine converts the electrical energy to mechanical energy. The efficiency of standard motors range between 83% and 92% while energy-efficient engines perform even better [7]. The losses in an engine consists, among other things, of stator, rotor and stray load losses. These losses are load dependent and results in heat generation. While the rotor and stator losses depend on the resistance to current flow and are proportional to the resistance of the material and the square of the current (RI^2) the stray load losses are more difficult to calculate, as they arise from a variety of sources, but are generally also proportional to the square of the rotor current [7].

Though comprehensive literature in the technology, policy and energy savings of electrical engines can be found in [6] a correct calculation of energy use in an engine is difficult to achieve. The calculations in this thesis are based on the work in [13], [15] and [28], along with some further assumptions and simplifications explained below.

4.2 Calculating the energy consumption

The power consumption, $P(t)$, of an electrical engine can be described as:

$$P(t) = V(t)I(t) \tag{4.1}$$

where $V(t)$ and $I(t)$ are the DC voltage and current respectively. With the voltage defined as

$$V = RI(t) + K_v K_R \omega(t) \tag{4.2}$$

the power $P(t)$ can be described as a function of the electrical current, I, and the angular velocity ω such as

$$P(t) = (RI(t) + K_v K_R \omega(t))I(t) = RI^2(t) + K_v K_R \omega(t) I(t) \tag{4.3}$$

where R is the stator resistance, K_v the electrical (back emf) constant and K_R the transmission gear ratio.

The power consumption can, as mentioned before, be divided into two terms, one describing the heat dissipated losses, P_d and one describing the mechanical power, P_m used to perform work such as:

$$P(t) = P_d(t) + P_m(t) \tag{4.4}$$

where the heat dissipated losses and the mechanical power are

$$P_d(t) = RI^2(t) \tag{4.5}$$

$$P_m(t) = K\omega(t)I(t) \tag{4.6}$$

respectively and the constant K in $P_m(t)$ describing the product of $K_v K_R$.

Given the power consumption, the energy is simply equal the power consumption over time such as:

$$E = \int_0^{t_f} P(t)dt = \int_0^{t_f} (P_d(t) + P_m(t))dt \tag{4.7}$$

Equivalent to the power the energy consumption can be divided into two terms [15] describing the heat dissipated energy, W_d and the mechanical work, E_m, as

$$E = W_d + E_m \tag{4.8}$$

where

$$W_d = \int_0^{t_f} P_d(t)dt = R\int_0^{t_f} I(t)^2 dt \tag{4.9}$$

$$E_m = \int_0^{t_f} P_m(t)dt = K\int_0^{t_f} \omega(t)I(t)dt \tag{4.10}$$

This give a simplified, but yet sufficient, model of the energy consumption in an electrical engine powering the different joints in an industrial robot. For the energy calculations in this thesis some further simplifications and and assumptions are necessary. This is explained further in Chapter 5 describing the experimental setup and the data analysis.

4.3 Summary Energy calculations

The power consumption in an electrical engine can be described as a function of the electrical current and the angular velocity. The total power consumption can also be divided into terms describing the heat losses and the mechanical power respectively. Given the expression for the power, the energy consumption is derived as the total power consumption integrated over time.

5

Methodology and tools

Given the theoretical aspects described in Chapter 2, 3 and 4 the experimental setup can be created. This chapter describes the work procedure, the tools used for the experiments and the experimental environment. It gives a description of the robots and the programs used to evaluate trajectories and explains the data collection and analysis.

5.1 Evaluation methodology

The main objectives of this thesis are described as follows:

- Build an understanding of the existing trajectory planning strategy in the robot.
- Develop an optimization strategy to minimize the energy consumption.
- Investigate spline blocks as a possible implementation tool for the optimization strategy.

Furthermore, the work is divided into three tasks related to each objective. Before addressing any of the tasks, the experimental setup needs to be defined along with the methodology for evaluation. Chapter 2 provided a necessary fundamental understanding of industrial robots while Chapter 3 described the theoretical background of the optimization algorithm and Chapter 4 the theoretical aspect of energy calculations. This chapter describes the development of tools and methods to implement these theories. The three tasks described above are all following a similar experimental procedure.

1. Generate code to create a trajectory
2. Implement and execute the trajectory in the robot
3. Log data from the experiment
4. Derive the relevant results
5. Analyze the result

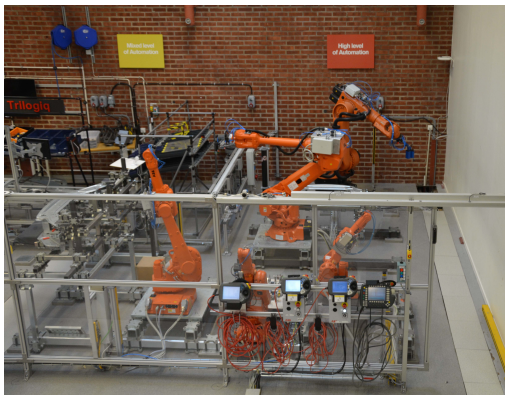


Figure 5.1: Overview of the Robotic Laboratory at Chalmers University of Technology

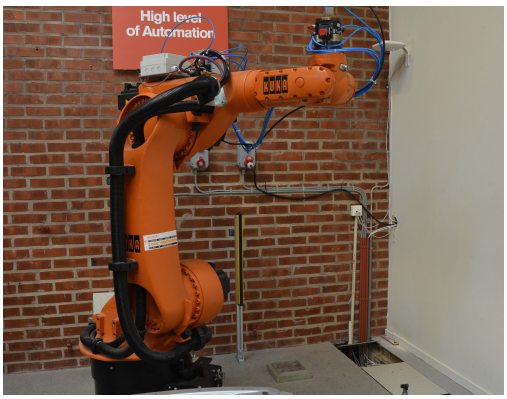


Figure 5.2: The KUKA robot in the Robotic Laboratory at Chalmers University of Technology.

The details of the procedure are explained in the following sections. The experiments are mainly performed in the robotic laboratory at Chalmers University of Technology. The setup of the robot and the robot cell can be seen in Figure 5.1 and 5.2. The evaluation of splines as an implementation method is performed at the KUKA office in Gothenburg as the necessary software is lacking in the robot at Chalmers.

5.2 The robots

The two robots used in the experiments are a KUKA KR30 with control system KRC2, provided by the laboratory at Chalmers University of Technology, and a KR16 with control system KRC4, provided by the KUKA office in Gothenburg. The KR16 is used to evaluate the spline blocks, as the KR30 is not equipped with these functions. Both robots are anthropomorphic six jointed, as described in Chapter 2.1 and seen in Figure 2.1. The joints are powered by transistor-controlled, low-inertia brushless AC servomotors. They are mounted to the floor and have a rated payload/supplementary payload of 30/35 kg and 16/30 kg respectively, which can be moved at maximum speed with the arm fully extended. As described in Figure 5.3, the rated payload is the maximum load the end-effector can handle, while the supplementary payload is the load mounted on the robot up to joint six (such as air pressure cylinders and cable packages). The range of motion and maximum speed of each joint can be seen in Table 5.1 and 5.2 given in [29] and [30].

5.3 Implementation of trajectories in the robot

The trajectories used for the three tasks have to be defined and implemented in the robots for evaluation. This is realized in a similar fashion for the first and third task. As both tasks involves evaluation of the robot and the existing software, trajectories are created using the controller in the robot. For task one, the points creating the path are defined

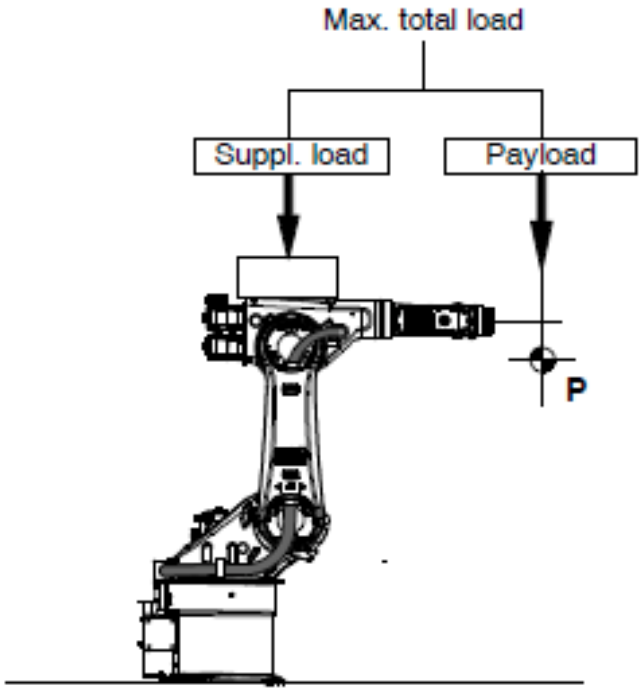


Figure 5.3: The rated payload and the supplementary payload.

in the initiation of the program and the trajectory is created and altered by choosing velocity and acceleration in the inline form. The code used for one of the trajectories in task one can be seen in Figure 5.4. Initially a sequence of simple trajectories is performed with varying settings for velocity and acceleration to build an understanding for the motions. The trajectories gradually become more complex to mimic real robot cell setups.

For task three the path of the trajectory is defined within a spline block. The points along the path is then moved to evaluate the response of the robot.

In contrast to the generation of paths and trajectories for task one and three, the trajectories used in task two are generated differently. In task two, the trajectories from task one is used as input to the optimizer which then create optimized trajectories with the same posture and execution time. The trajectories from task one are therefore hereon referred to as original trajectories and the trajectories from the optimizer as the optimized trajectories. With the original trajectories as a starting point, optimized trajectories are generated, executed and compared. The optimized trajectories are generated offline, that is, not in the robot, and therefore also implemented differently.

Table 5.1: The range of motion and maximum speed for each joint of the KR30-3 with in-line wrist and rated payload 30 kg.

Joint	Range of motion software-limited	Speed
1	$\pm 185°$	$140°/s$
2	$-135° - 35°$	$126°/s$
3	$-120° - 158°$	$140°/s$
4	$\pm 350°$	$260°/s$
5	$\pm 119°$	$245°/s$
6	$\pm 350°$	$322°/s$

Table 5.2: The range of motion and maximum speed for each joint of the KR16 with in-line wrist and rated payload 16 kg.

Joint	Range of motion software-limited	Speed
1	$\pm 185°$	$156°/s$
2	$-155° - 35°$	$156°/s$
3	$-130° - 154°$	$156°/s$
4	$\pm 350°$	$330°/s$
5	$\pm 130°$	$330°/s$
6	$\pm 350°$	$615°/s$

5.3.1 Generating optimized trajectories

The generation of optimized trajectories differs greatly from the generation of the original trajectories. In addition to the state-of-the-art procedure described in Chapter 2 a software named Sequence Planner, developed at Chalmers University of Technology, is used to generate the desired sequences of operations. Full details about Sequence Plan-

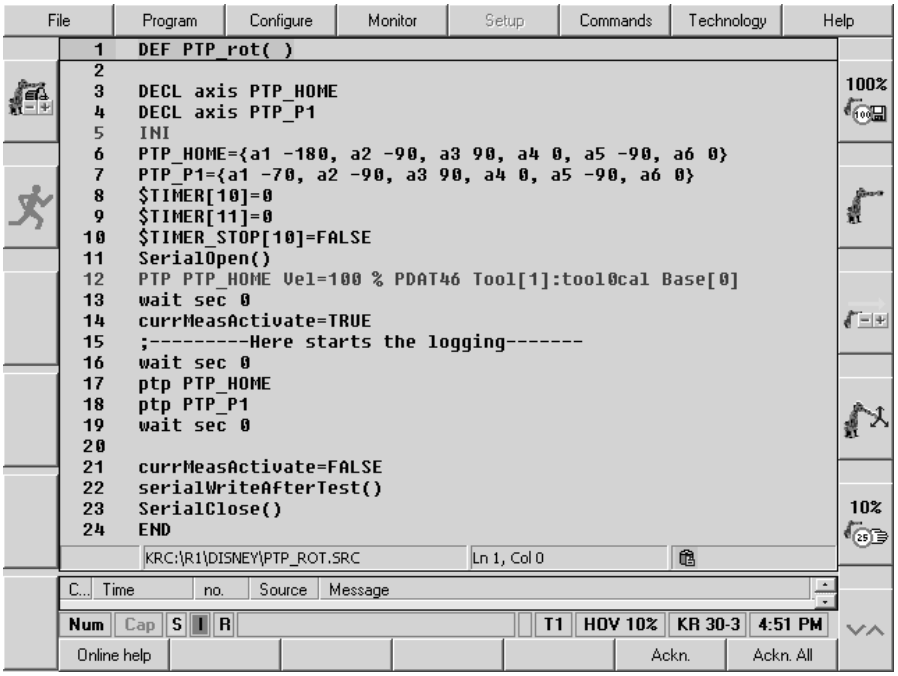


Figure 5.4: The code used to capture the motion where joint 1 is rotating 110 degrees.

ner can be found in [31] but in short is Sequence Planner featuring a number of functions related to visualization of operation sequences projected from various perspectives such as product, resource, human or safety [26]. The procedure generating optimized trajectories now roughly consists of six steps [27]:

1. Generate an original trajectory using the robot or a simulation software such as Delmia V5 [32] or Visual Component 3DCreate [33]
2. Use the generated trajectory as input to Sequence Planner [31]. Manually add extra execution conditions if necessary, for example requirements on standstills or order of operations if more than one robot is optimized.
3. Through Sequence Planner the sequence of operations is identified, the logical behavior verified and operations split into multiple parts based on the motions.
4. The Ipopt library [34] solves the optimization problem, defined in Chapter 3, and the Sequence Planner returns an optimized trajectory
5. The optimized trajectory is time shifted compared to the original trajectory and therefore interpolated using cubic spline to generate the optimal trajectory with data samples matching the specific sampling time equal the original trajectory.
6. The interpolated trajectory is converted to an ascii file possible to execute in the robot using an ASCII program.

The interpolation in the second to last point in the list is due to the setup and constraints in the optimization algorithm. The position samples in the original trajectory are used as fixed inputs and the time samples as the degree of freedom. As a result, the optimized trajectory, generated by the optimizer, is containing the same amount of samples with the same positions but with other time samples. As the generated optimized trajectory is implemented and executed in the robot by other means than the original trajectory the time sampling has to be shifted to be compatible with the implementation program.

5.3.2 Implementing optimized trajectories

The optimized trajectories are executed in the robot using an ASCII program created by the programmers at KUKA Gothenburg. The optimization algorithm writes the trajectory to an ascii file with each line representing a time instant followed by the angular position of the six joints. The time interval between each samples is 12ms. The velocity and accelerations are not defined but rather a result of the distance traveled between the fixed time interval. To implement the trajectory in the robot the ascii file is imported to the predefined location path.emi on the robots hard drive. With the trajectory implemented at the correct location, the ASCII program executes the trajectory by loading the data located at path.emi to the robot. Given the text file is correctly formatted, otherwise a message is displayed explaining the error, the program pauses at the first position in the text file before running through the following positions. Finally the program unloads the ascii file before ending the program. An example of the ascii file path.emi and a part of the ASCII program can be seen in Figure 5.5 and 5.6. To execute a new optimal trajectory the related ascii file has to be imported to path.emi and thereby automatically overwrite the old trajectory.

path.emi.txt — Redigerad

```
[HEADER]
GEAR_NOMINAL_VEL = 1.0
NUM_ROB_JOINTS = 6
CRC = 2942578096
[RECORDS]
0       -160     0              0              0       -90     0
0.012   -160    -0.0034009      0.0034009      0       -90     0
0.024   -160    -0.0183981      0.0183981      0       -90     0
0.036   -160    -0.0519634      0.0519634      0       -90     0
0.048   -160    -0.11025        0.11025        0       -90     0
0.06    -160    -0.19546        0.19546        0       -90     0
0.072   -160    -0.308459       0.308459       0       -90     0
0.084   -160    -0.450061       0.450061       0       -90     0
0.096   -160    -0.620771       0.620771       0       -90     0
0.108   -160    -0.820697       0.820697       0       -90     0
0.12    -160    -1.0498         1.0498         0       -90     0
0.132   -160    -1.30805        1.30805        0       -90     0
0.144   -160    -1.59529        1.59529        0       -90     0
0.156   -160    -1.91136        1.91136        0       -90     0
0.168   -160    -2.2561         2.2561         0       -90     0
0.18    -160    -2.62925        2.62925        0       -90     0
0.192   -160    -3.03062        3.03062        0       -90     0
0.204   -160    -3.45991        3.45991        0       -90     0
0.216   -160    -3.91689        3.91689        0       -90     0
0.228   -160    -4.40126        4.40126        0       -90     0
0.24    -160    -4.91269        4.91269        0       -90     0
0.252   -160    -5.45095        5.45095        0       -90     0
0.264   -160    -6.0156         6.0156         0       -90     0
0.276   -160    -6.60639        6.60639        0       -90     0
0.288   -160    -7.22292        7.22292        0       -90     0
0.3     -160    -7.8648         7.8648         0       -90     0
0.312   -160    -8.53174        8.53174 0      -90     0
0.324   -160    -9.2233 9.2233  0       -90    0
0.336   -160    -9.93907        9.93907 0      -90     0
```

Figure 5.5: The header and the first 25 samples of an optimized trajectory in the text file path.emi used to execute the ASCII program.

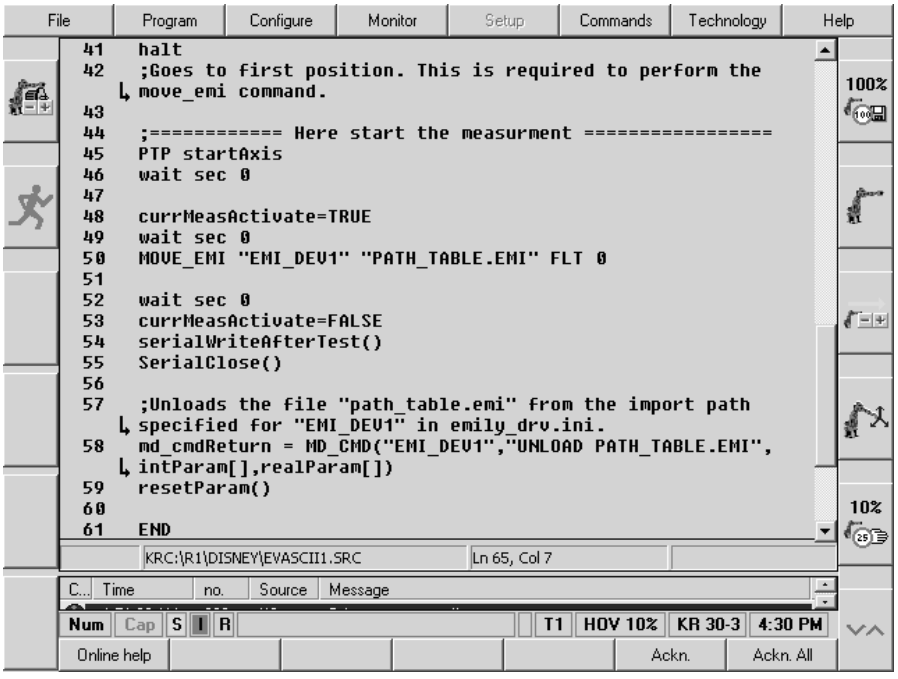


Figure 5.6: One section in the ASCII program used to run optimized trajectories in the KUKA robot.

5.4 Data Collection

To evaluate the executed trajectories, it is crucial to collect relevant data. The data collection is limited by the variables present and available in the robots. Two variables the robot automatically measure are the true position of all joints and the electrical current fed to each of the six motors powering the joints. According to [29], [30] the positions of the joints are sensed by means of a cyclically absolute positioning sensing system, featuring a resolver for each axis in the robots, and captured in the global variable $\$AXIS\_ACT\_MEAS$ in the robots system file. In a similar fashion are the actual current of the joints captured in the global variable $\$CURR\_ACT$ as a percentage of the lower value of maximum amplifier or motor current. This data is useful to evaluate the trajectory and energy consumption and therefore desired to extract from the robot. This is achieved through the development of a program named the logger.

5.4.1 The logger

The program is designed to capture the position and electrical current in two matrices every 12 millisecond and export the data to a text file. The program consists of a number of functions, each designed to perform a specific task, executed sequentially. The program logs the values of the electrical current in each joint (in percent of max ampere) followed by the position of the six joints (in degrees) and the time instant (in milliseconds). An example of a logged file can be seen in Figure 5.7. The package of programs is added to both the code used to generate original trajectories as well as the ASCII program executing the optimized trajectories. This allows data from each trajectory to be collected and exported for later analysis and comparison.

1												
2	├0.3662	2.4964	−0.8423	−3.3876	1.6022	−2.9847	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9828
3	−0.3723	2.4140	−0.9003	−3.3509	1.4740	−2.9817	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9840
4	−0.4089	2.3988	−0.8301	−3.4394	1.6449	−2.9756	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9852
5	−0.4517	2.3957	−0.8850	−3.4669	1.4557	−2.9847	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9864
6	−0.4089	2.4079	−0.6958	−3.3692	1.6419	−3.0183	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9876
7	−0.3906	2.3225	−0.9430	−3.5249	1.5748	−2.9725	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9888
8	−0.3937	2.2828	−0.6531	−3.3662	1.5625	−2.9725	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9900
9	−0.3967	2.3865	−0.8087	−3.3418	1.6419	−2.9359	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9912
10	−0.2838	2.3560	−0.9095	−3.4761	1.6816	−2.9633	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9924
11	−0.4273	2.3347	−0.7630	−3.3967	1.5809	−3.0030	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9936
12	−0.4761	2.4934	−0.8301	−3.4486	1.4954	−2.9847	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9948
13	−0.3754	2.3957	−0.9796	−3.4150	1.5168	−2.9969	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9960
14	−0.9705	2.3927	−0.8484	−3.3815	1.4161	−2.9786	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9972
15	−0.4975	2.3957	−0.9339	−3.4974	1.5625	−2.9206	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9984
16	−0.4517	2.3225	−0.8850	−3.4486	1.4801	−2.9817	−180.0000	−90.0000	0.0000	0.0000	0.0000	0.0000	9996
17	−0.4547	2.3835	−0.8393	−3.4089	1.5503	−2.9878	−179.9989	−90.0000	0.0000	0.0000	0.0000	0.0000	10008
18	−8.6978	2.3560	−0.8484	−3.3692	1.5198	−2.9847	−179.9872	−90.0000	0.0000	0.0000	0.0000	0.0000	10020
19	−15.6011	2.3499	−0.7569	−3.4822	1.6572	−2.9695	−179.9539	−90.0000	0.0000	0.0000	0.0000	0.0000	10032
20	−24.9153	2.4049	−0.7447	−3.4211	1.5687	−2.9725	−179.8865	−90.0000	0.0000	0.0000	0.0000	0.0000	10044
21	−33.7138	2.3438	−0.7477	−3.4211	1.5931	−2.8443	−179.7731	−90.0000	0.0000	0.0000	0.0000	0.0000	10056
22	−38.4747	2.3316	−0.9003	−3.3540	1.5290	−2.9664	−179.6018	−90.0000	0.0000	0.0000	0.0000	0.0000	10068
23	−49.3515	2.1851	−0.8118	−3.4333	1.5656	−2.9298	−179.3610	−90.0000	0.0000	0.0000	0.0000	0.0000	10080

Figure 5.7: The data logged at every executed trajectory. Column one through six are the percentage of electrical current on joint 1-6, column 7-12 are the position in degree for joint 1-6 and column 13 is the time in milliseconds.

5.5 Data analysis

Given the data from the logger the position, velocity acceleration and any higher derivatives, as well as execution time and power consumption can be calculated and analyzed using MATLAB. Since the logger starts recording a few instances before the trajectory starts, and stops recording a few instances after the trajectory has finished, the data is groomed manually to give a fair comparison. The trajectories are assumed to start moving immediately why the first sample in the collected data will contain the starting position instantly followed by a sample with values separated from the starting position. All trajectories are designed to have no motion in the final position. To verify that the robot have stopped and neither velocity nor acceleration is present three samples containing equal values are needed. Therefore, the trajectories are defined to contain three equal values in the end. All values after that are erased.

With the clean data the calculations are performed as described below and the result is stored in -mat files for later analysis.

5.5.1 Calculating execution time, path and trajectory

With the constant sampling time $h = 0.012$ sec and the number of samples n known the execution time t_f is calculated as:

$$t_f = nh = 0.012n \tag{5.1}$$

Further, given the position, or more exact, the angle, θ [°], in each sample, k, the angular velocity, ω [°/s], and the angular acceleration, $\dot{\omega}$ [°/s^2], are calculated for each joint, i, as:

$$\omega_i^{deg}(k) = \frac{\theta_i(k) - \theta_i(k-1)}{h} \qquad i = 1,2\dots6 \quad k = 1,2,\dots,n \tag{5.2}$$

$$\dot{\omega}_i^{deg}(k) = \frac{\omega_i(k) - \omega_i(k-1)}{h} \qquad i = 1,2\dots6 \quad k = 1,2,\dots,n \tag{5.3}$$

In a similar fashion, the jerk, $\ddot{w}_i^{deg}$ [°/s^3], and snap, $\dddot{w}_i^{deg}$ [°/s^4], can be calculated. The path, velocity, acceleration and jerk profile can now be plotted and compared between trajectories. For the energy consumption calculation, the angular velocity is also converted to rad/s

$$\omega_i = \frac{\pi}{180}\omega_i^{deg} \tag{5.4}$$

5.5.2 Calculating the energy consumption

From the logger the position is given along with the electrical current expressed as percentage of the maximum electrical current available. To calculate the power and energy consumption as described in Chapter 4 the electrical current on each joint, I_i, and the angular velocity, ω_i, are needed. ω_i is derived with equation (5.4) while, given the maximum electrical current $I_{max,i}$ for the two robots in Table 5.3, the electrical current for each joint consumed in the trajectories can be derived from the percentage value $I_{log,i}^{\%}(k)$, given by the logger as

$$I_i(k) = I_{log,i}^{\%}(k) * I_{max,i} \tag{5.5}$$

Now, given the equations for electrical engines power and energy consumption in (4.4) and (4.8), the overall power and energy consumption for a robot could be derived as

$$E_{tot} = \sum_{i=1}^{N} E_i \quad i = 1,2,\dots,N \tag{5.6}$$

Table 5.3: The maximum electrical current [amp] on each joint for the KR16 and KR30.

Robot	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
KR16	20A	20A	14A	8A	8A	8A
KR30	32A	32A	32A	8A	8A	8A

where E_i is the total energy consumption for the i^{th} joint and N is the number of joints in the robot.

There are some few aspect needed to take into consideration though. Given the engines are not equal for all joints the constants, R and K are also not equal over the different engines why they better be expressed as R_i and K_i where $i = 1,2,\ldots,N$ is the number of the joint. Further more, the values of R_i and K_i and the ratio between the two are unknown and hard to estimate why the total energy consumption can not be derived exactly. The heat dissipated power and energy, and the mechanical power and work, are therefore derived separately. As one objective of this thesis is to compare the results, the comparison between different trajectories and, more exact, the energy savings are more critical than the true energy values. The true values for the power and energy consumption are therefore not presented. Instead all values are compared between trajectories and joints, allowing the unknown R_i and K_i to be eliminated as all trajectories compared are executed on the same robot and the constants can be assumed to be equal over the different runs. That is, the relation between the heat dissipated power, P_d, and the mechanical power, P_m, for an original and an optimized trajectory for each joint is derived respectively as

$$\frac{P_{d,opt,i}}{P_{d,org,i}} = \frac{R_i I_{opt,i}^2}{R_i I_{org,i}^2} = \frac{I_{opt,i}^2}{I_{org,i}^2} \tag{5.7}$$

$$\frac{P_{m,opt,i}}{P_{m,org,i}} = \frac{K_i I_{opt,i}\omega_{opt,i}}{K_i I_{org,i}\omega_{org,i}} = \frac{I_{opt,i}\omega_{opt,i}}{I_{org,i}\omega_{org,i}} \tag{5.8}$$

Furthermore, the relationship between the heat dissipated energy, W_d and the mechanical work, E_m for the original and optimized trajectories are given as

$$\frac{W_{d,opt,i}}{W_{d,org,i}} = \frac{\int_0^{t_f} P_{d,opt,i}dt}{\int_0^{t_f} P_{d,org,i}dt} = \frac{\int_0^{t_f} I_{opt,i}^2 dt}{\int_0^{t_f} I_{org,i}^2 dt} \tag{5.9}$$

$$\frac{E_{m,opt,i}}{E_{m,org,i}} = \frac{\int_0^{t_f} P_{m,opt,i}dt}{\int_0^{t_f} P_{m,org,i}dt} = \frac{\int_0^{t_f} I_{opt,i}\omega_{opt,i}dt}{\int_0^{t_f} I_{org,i}\omega_{org,i}dt} \tag{5.10}$$

Again, as the aim of this thesis is to decrease energy consumption, the results of interest are the energy savings why all data will be presented as a percentage of savings in relation to an original trajectory. The energy savings, expressed in percentage, for the heat dissipated term, $S_{W_d,i}$ and for the mechanical work, $S_{E_m,i}$ is then derived as

$$S_{W_d,i} = 100 * (1 - \frac{W_{d,opt,i}}{W_{d,org,i}}) \tag{5.11}$$

$$S_{E_m,i} = 100 * (1 - \frac{E_{m,opt,i}}{E_{m,org,i}}) \tag{5.12}$$

The total energy saving can not be derived as the relationship between the heat dissipated energy and the mechanical work is not known. Instead, the heat dissipated savings and the mechanical work reduction can be seen as the boundaries, and the total energy reduction should be somewhere in between.

This is the final step in the methodology describing the data analysis. With the energy consumption expressed as above, no emphasis is put on the true values and all results will be presented either normalized or, in a comparison, as a percentage savings. Provided the tools to generate and execute trajectories as well as a methodology to extract and analyze data, the experimental setup is completed and the experiments can be conducted.

5.6 Summary of methodology and tools

The work procedure is defined and the experimental setup created. Original trajectories are generated in the robot using the robot controller, while a program to execute optimized trajectories have been designed. The resulting data is collected as values on position and electrical current sampled each 12 milliseconds. As the constants R and K are unknown the true energy consumption can not be calculated. Instead are trajectories compared and the energy consumption is described as a percentage saving.

6

Evaluation of point-to-point motions

To build a better understanding of the robot, its movement and the energy distribution over the different joints some initial experiments are performed. The settings, available when programming the robots using the inline form, are manipulated to investigate their influence on the path and trajectory. This chapter describe the experiments and the conclusions given the results. The trajectories defined in this chapter are later used in the optimization.

6.1 Evaluation of the logger and ASCII program

Before starting the experiments, the logger and the ASCII program, used to run the optimized trajectories, are evaluated to ensure correct data collection.

By running an ascii file and comparing the logged data with the original data both the logger and the ASCII program is ensured to be working properly. The ascii file is computed with data samples every 12 millisecond and the logger is logging data with the same time interval why a plot of the position, velocity and acceleration of the two should conform.

Figure 6.1 show a trajectory created with an ASCII file along with the logged result. The two trajectories are consistent, confirming the reliability of the logger and ASCII program.

6.2 Motion configurations

As explained earlier a trajectory can be created in several different ways depending on the chosen motion and settings. In the optimization, the velocity and acceleration profiles are crucial, why the initial tests will focus on the effect different velocity and acceleration settings have on a PTP trajectory and the energy consumption. Two different motions are evaluated. First a rotational motion, exciting one joint and with no change in the gravitational force acting on the robot. The motion is evaluated with decreasing acceleration and velocity. In the second experiment the effects of the gravitational force are investigated by extending the robot to the outer bound of its workspace. Both the extending and the reversed retracting motion are evaluated with decreasing velocity.

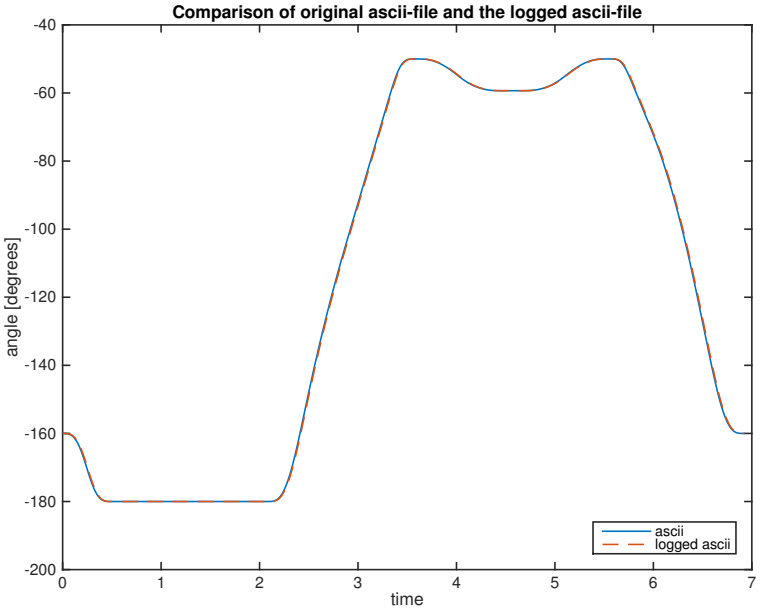


Figure 6.1: Comparison of the ascii-file exported to the robot and the logged result showing the reliability of the logger and ASCII program.

6.3 Rotational motion

In the first experiment only joint 1 is moving, resulting in a rotation of the robot around its vertical axis. This motion is chosen with the intent to keep the gravitational consistent out through the motion. The robot is positioned in a predefined home position and rotated 110 degrees. The start and end position of the robot can be seen in Figure 6.2 and 6.3. The motion is carried out in two configurations. First the acceleration is set to 100 percent and velocity decreased from 100 to 30 percent with steps of 10. In the second configuration the velocity is set to 100 percent and acceleration decreased from 100 to 30 percent with steps of 10. The resulting trajectory and electrical current are recorded and collected using the logger.

6.3.1 Velocity varied rotation

The resulting energy reduction for joint 1 with velocities decreasing from 100 to 30 percent, along with the resulting execution time can be seen in Table 6.1. The energy reduction is displayed both with the heat dissipating energy saving, $S_{W_d,1}$, and the mechanical work reduction, $S_{E_m,1}$. The reduction in energy is proportional to the reduction in velocity while the increasing execution time is not. In Figure 6.4 the heat dissipated energy and the mechanical work for each run is presented as a function of its execution time, normalized over the fastest run. The energy consumption decreases with the velocity but with the cost of longer execution time. The trade off between time loss and energy saving is better for higher velocities. As mentioned before the energy saving for the heat dissipated energy and the mechanical work gives the boundaries for the total energy saved. For this simple motion the savings for both the heat dissipated energy and the mechanical work, and therefore the total energy, is approximately equal the velocity

Figure 6.2: The starting position of the robot for the rotational motion.
$HOME = [-180 \quad -90 \quad 90 \quad 0 \quad -90 \quad 0]$

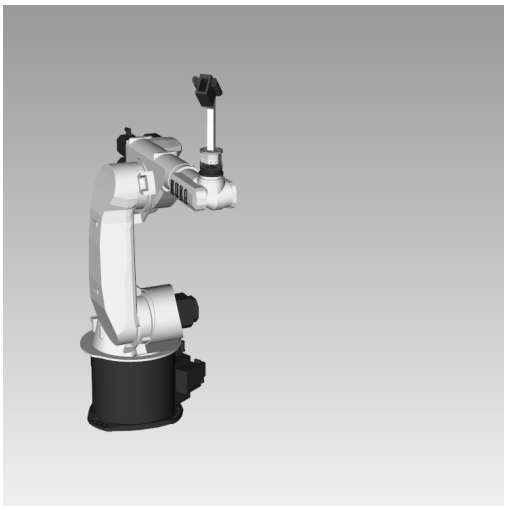


Figure 6.3: The finishing position of the robot for the rotational motion.
$P1 = [-70 \quad -90 \quad 90 \quad 0 \quad -90 \quad 0]$

Table 6.1: Energy savings for the rotational movement with decreasing velocity. The table show both the heat dissipated savings, $S_{W_d,1}$, and the mechanical work reduction, $S_{E_m,1}$, for joint 1.

Velocity	[%]	**100**	**90**	**80**	**70**	**60**	**50**	**40**	**30**
$\boldsymbol{T_f}$	$[s]$	1.164	1.224	1.308	1.428	1.596	1.836	2.196	2.832
$\frac{\boldsymbol{T_f}}{\boldsymbol{T_f^{100}}}$	[%]	100	105	112	122	137	158	189	243
$\boldsymbol{S_{W_d,1}}$	[%]	0	9	20	30	39	51	60	69
$\boldsymbol{S_{E_m,1}}$	[%]	0	9	20	29	38	47	56	64

reduction.

Figure 6.5 and Figure 6.6 show the velocity and acceleration profile for the different values of velocity. The velocity profiles are identical in the initial acceleration phase showing that the robot is using the same acceleration profile independently of the velocity settings. The same is true for the deceleration phase. The only difference in the trajectories are the the maximum velocity reached and therefore also the execution time. For the trajectory with 100 percent velocity the robot reaches a value of 240 degrees per second, consistent with the value for maximum velocity in Table 5.1. For the other trajectories the maximum velocity is consistent with the given percentage.

In Figure 6.7 the position, velocity and acceleration is presented for joint 1 rotating with 100 percent velocity, as well as the electrical current, I_i over all joints, the heat dissipated power consumption, $P_{d,1}$, and the mechanical power consumption, $P_{m,1}$. The motion can be divided in to three phases, acceleration, constant velocity and deceleration. Comparing the power with the motion in Figure 6.7, identifies the acceleration phase as the major mechanical power consumer and both the acceleration and deceleration phases as the major heat dissipated power consumer.

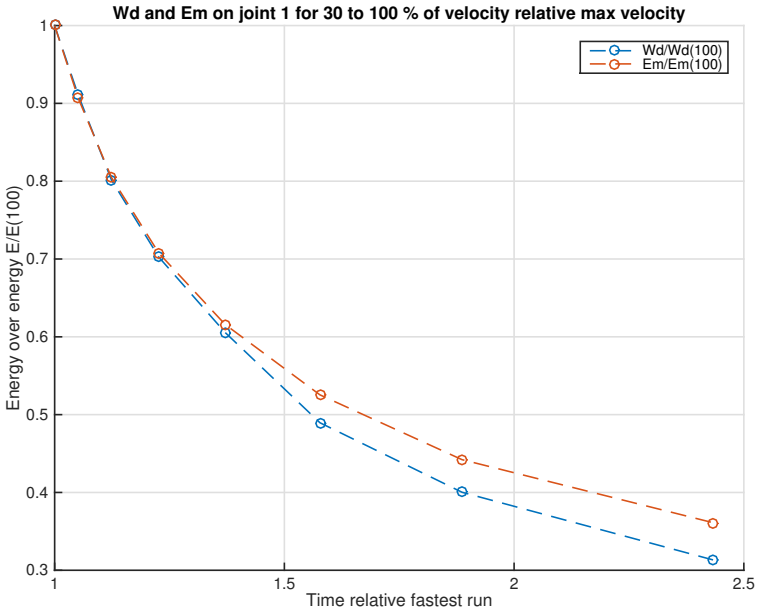


Figure 6.4: The heat dissipated energy consumption, W_d, and the mechanical work, E_m on joint 1 for the rotational movement with decreasing velocity (100-30 percent) as a function of the time normalized with respect to fastest run.

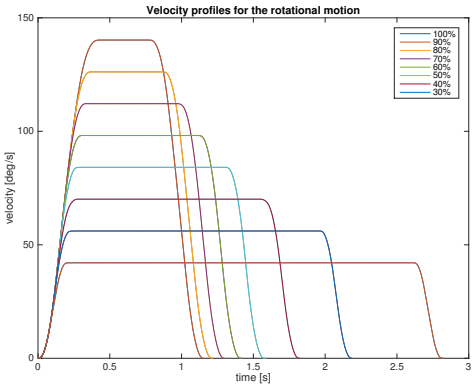


Figure 6.5: The velocity profiles for the rotational movement with velocity from 100 to 30 percent.

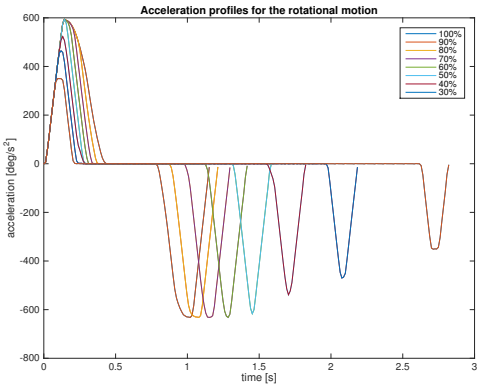


Figure 6.6: The acceleration profiles for the rotational movement with velocity from 100 to 30 percent.

Maintaining a constant velocity consume relatively little power in both plots but there is a significantly difference for the heat dissipated power where the constant velocity takes values approximately 1/10 of the maximum value. For the mechanical power this difference is only around 1/2. This is reasonable since the mechanical work is not only dependent on the current but also the velocity. The heat dissipated power and the mechanical power for the decreasing velocities can be seen in Figure 6.8 and 6.9.

There is also a very interesting and distinct similarity between the acceleration profile in the upper rightmost plot in Figure 6.7 and the electrical current profile for joint 1 in the lower leftmost plot in Figure 6.7(d). Since the motion is limited to joint 1 with all other joint configurations constant it would be reasonable to only have a variation in the electrical current on joint 1 but, as seen in th lower leftmost plot in Figure 6.7, describing the electrical current on each joint, there is also some variation for joint 2 and 4. This is believed to be a result of the centrifugal force acting on those joint due to the orientation and positioning of the robot, seen in Figure 6.2, during the rotation. As it is rotating the increased power consumption is preventing the arm to fall out.

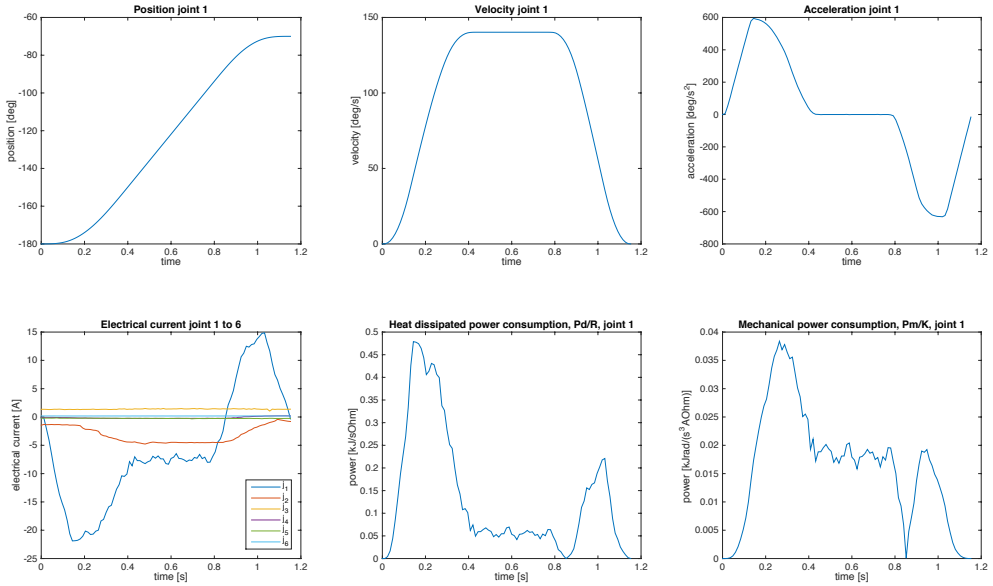


Figure 6.7: An overview of the rotational movement showing the position, velocity, acceleration and jerk for joint 1 as well as the power consumption on all joints and the total energy consumption.

6.3.2 Acceleration varied rotation

In the second configuration of the rotational motion, the same 110 degree rotation is executed but with acceleration decreasing from 100 percent to 30 percent in steps of 10 and velocity set to 100 percent. The corresponding execution time and energy savings are found in Table 6.2. Just as for the velocity varied trajectories the energy saving is described both as heat dissipating saving, $S_{W_d,1}$, and the mechanical work reduction, $S_{E_m,1}$.

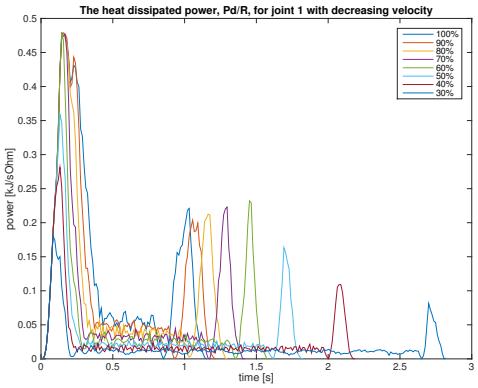


Figure 6.8: The heat dissipated power for the rotational motion with velocity from 100 to 30 percent.

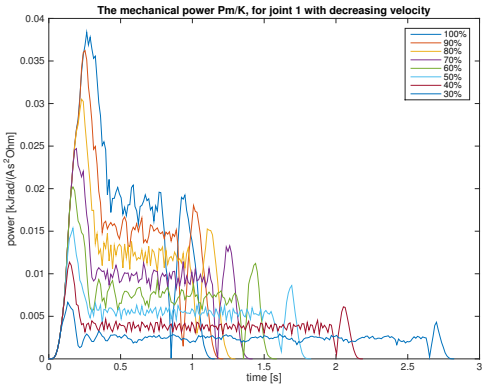


Figure 6.9: The mechanical power for the rotational movement with velocity from 100 to 30 percent.

Table 6.2: Energy savings for the rotational movement with decreasing acceleration. The table show both the heat dissipated savings, $S_{W_d,1}$, and the mechanical work reduction, $S_{E_m,1}$, for joint 1.

Velocity	[%]	**100**	**90**	**80**	**70**	**60**	**50**	**40**	**30**
$\boldsymbol{T_f}$	$[s]$	1.164	1.200	1.224	1.272	1.332	1.416	1.536	1.740
$\frac{\boldsymbol{T_f}}{\boldsymbol{T_f^{100}}}$	[%]	100	103	105	109	114	122	132	149
$\boldsymbol{S_{W_d,1}}$	[%]	0	8	16	23	31	38	47	58
$\boldsymbol{S_{E_m,1}}$	[%]	0	3	7	10	15	21	30	40

The energy consumption for decreasing acceleration is presented as a function of the time normalized with respect to the fastest run in Figure 6.10. In Figure 6.11 the same functions are compared to the energy consumption related to the velocity decreased trajectories. There are four interesting conclusions that can be drawn from the comparison. First, independently of reduced velocity or acceleration, the energy consumption is reduced. Secondly, the execution time is not as highly affected for decreased acceleration as for decreased velocity. Thirdly, the heat dissipated energy W_d shows significantly better results for decreased acceleration compared to decreased velocity. Last, the mechanical work is slightly higher for the acceleration decreased trajectories.

The velocity and acceleration profiles for the acceleration decreased trajectories can be seen in Figure 6.12 and 6.13. In a similar way as for the velocity profiles in Figure 6.5 the acceleration profiles show the same decreasing from the maximum value consistent with the settings, the 70%-trajectory reaches a maximum acceleration of $420°/s^2$, corresponding to 70% of the maximum acceleration of $600°/s^2$.

The heat dissipated power and the mechanical power for the acceleration decreased trajectories are displayed in Figure 6.14 and 6.15. The heat dissipated power in Figure 6.14 has a much more distinct decrease in power for the acceleration phase compared to the velocity decreased heat dissipated power in Figure 6.8. This indicates that decreasing the acceleration is more

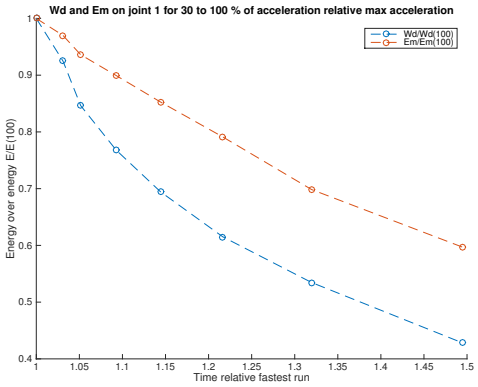


Figure 6.10: The heat dissipated energy, W_d and the mechanical work E_m of joint 1 for the rotational movement with decreasing acceleration (100-30 percent) as a function of the time normalized with respect to fastest run.

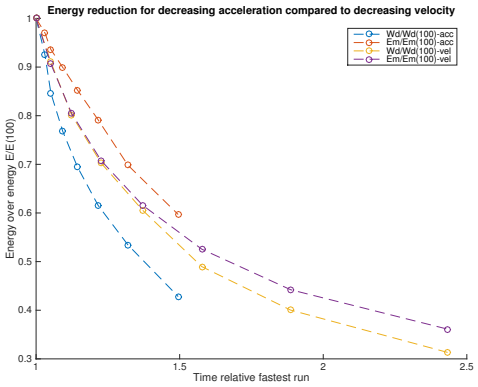


Figure 6.11: The heat dissipated energy, W_d and the mechanical work E_m of joint 1 for the rotational movement comparing decreasing velocity with decreasing acceleration. Both as a function of time normalized with respect to fastest run.

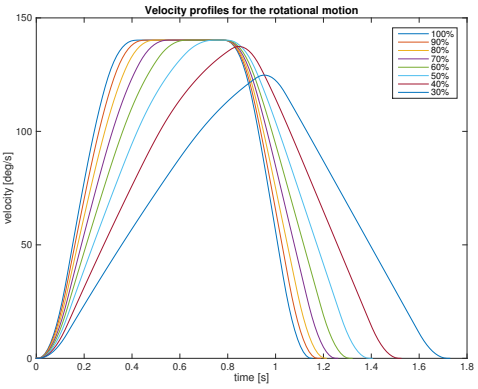


Figure 6.12: The velocity profiles for the rotational movement with acceleration from 100 to 30 percent.

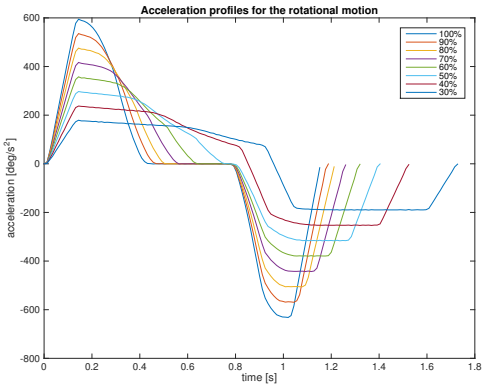


Figure 6.13: The acceleration profiles for the rotational movement with acceleration from 100 to 30 percent.

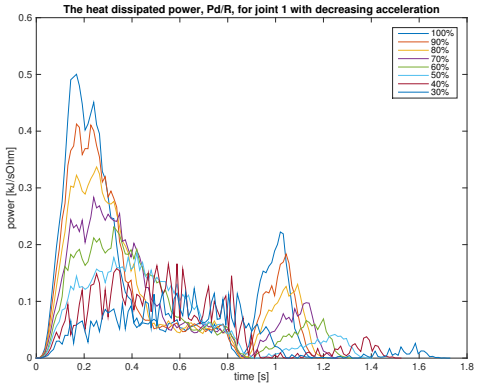


Figure 6.14: The heat dissipated power for the rotational motion with acceleration from 100 to 30 percent.

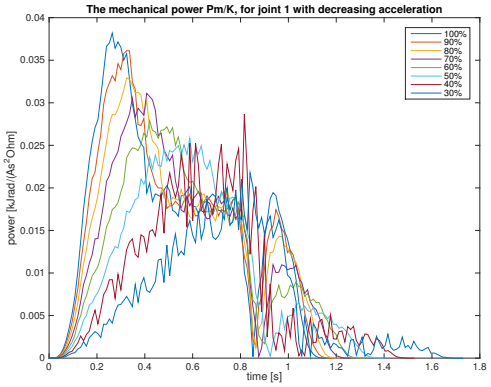


Figure 6.15: The mechanical power for the rotational motion with acceleration from 100 to 30 percent.

effective to reduce the heat dissipated power. On the other hand is the mechanical power not decreased as effectively but rather increased during the constant speed phase. This is due to the velocity being higher at decreasing acceleration compared to the decreasing velocities. Since the mechanical work is related to the velocity the values increases.

6.4 Gravitational affected movement

To investigate the effect of the gravitational force on the motion and power consumption two test are performed. One where the robot moves in the direction of the gravity (downward motion) and one where it moves in the negative direction of the gravity (upward motion). In the same way as for the rotational movement the robot is positioned in a home position and then moved to a second position, P2. The two positions can be seen in Figure 6.16 and 6.17 and the degrees in the both positions are stated below. The motions excite joint 2 and 3 and are conducted with the acceleration set to 100 percent and the velocity is decreased from 100 to 30 percent, by steps of 10.

$$\begin{aligned} \text{HOME} &= [\text{-180} \quad \text{-90} \quad 90 \quad 0 \quad \text{-90} \quad 0] \\ \text{P1} &= [\ \text{-180} \quad 0 \quad 0 \quad 0 \quad \text{-90} \quad 0] \end{aligned}$$

Figure 6.16: The starting position of the robot for the downward motion and the finishing position for the upward motion. $HOME = [-180 \quad -90 \quad 90 \quad 0 \quad -90 \quad 0]$

Figure 6.17: The finishing position of the robot for the downward motion and the starting position for the upward motion. $P1 = [-180 \quad 0 \quad 0 \quad 0 \quad -90 \quad 0]$

6.4.1 Energy consumption gravitational motion

In the same way as for the rotational motion the heat dissipated savings and the mechanical work reductions for the upward and downward motions can be found in Table 6.3 and Table 6.4 respectively.

Table 6.3: Energy savings for the upward gravitational motion with decreasing velocity. The table show both the heat dissipated savings, S_{W_d}, and the mechanical work reduction, S_{E_m}, for joint 2 and 3.

Velocity		[%]	**100**	**90**	**80**	**70**	**60**	**50**	**40**	**30**
Time	$\boldsymbol{T_f}$	$[s]$	1.440	1.440	1.452	1.500	1.596	1.740	2.004	2.470
	$\frac{\boldsymbol{T_f}}{\boldsymbol{T_f^{100}}}$	[%]	100	100	101	104	111	121	139	172
Joint 2	$\boldsymbol{S_{W_d}}$	[%]	0	1	6	10	16	19	20	18
	$\boldsymbol{S_{E_m}}$	[%]	0	0	6	11	18	23	29	34
Joint 3	$\boldsymbol{S_{W_d}}$	[%]	0	1	11	22	34	46	56	67
	$\boldsymbol{S_{E_m}}$	[%]	0	1	9	19	30	41	51	63

Table 6.4: Energy savings for the downward gravitational motion with decreasing velocity. The table show both the heat dissipated savings, S_{W_d}, and the mechanical work reduction, S_{E_m}, for joint 2 and 3.

Velocity		[%]	**100**	**90**	**80**	**70**	**60**	**50**	**40**	**30**
Time	$\boldsymbol{T_f}$	$[s]$	1.188	1.212	1.260	1.344	1.452	1.632	1.920	2.424
	$\frac{\boldsymbol{T_f}}{\boldsymbol{T_f^{100}}}$	[%]	100	102	106	113	122	137	161	204
Joint 2	$\boldsymbol{S_{W_d}}$	[%]	0	7	15	24	34	43	51	53
	$\boldsymbol{S_{E_m}}$	[%]	0	15	27	36	45	52	56	58
Joint 3	$\boldsymbol{S_{W_d}}$	[%]	0	4	11	17	21	26	30	32
	$\boldsymbol{S_{E_m}}$	[%]	0	2	7	12	16	22	28	36

The heat dissipated energy, the mechanical work and the total energy for joint 2 and 3 with different velocities are plotted as functions of the maximum energy for the upward an downward motions in Figure 6.18 and 6.19. The plots for joint 2 and 3 in 6.18 show a similar behavior to the opposite joints in 6.19. This is likely due to the similarity in the motion between the two runs, joint 2 moves downwards in a similar way as joint 3 moves upwards and vice verse.

If comparing the energy on joint 2 for the upward and downward motion more energy is required, as expected, to move upward, against the gravity. In Figure 6.20 and 6.21 the position, velocity and acceleration as well as the heat dissipated power and the mechanical power can be seen for joint 2 in the upward and downward motions with 100% velocity. For the upward motion the acceleration reaches a value of $-100°/s^2$ and then keeps that constant until it reaches a state where it has to start breaking to stop at the top. This acceleration profile is different from the acceleration profile for the downward or the rotational motion. The execution times for the upward motion is also quite much longer than the downward motion, due to the slow acceleration. In a similar fashion as for the rotational motion, the electrical current copy the acceleration profile. As mentioned before the downward motion is not as energy consuming as the upward motion. The acceleration profile for the downward motion show a fast acceleration and then a slower deceleration.

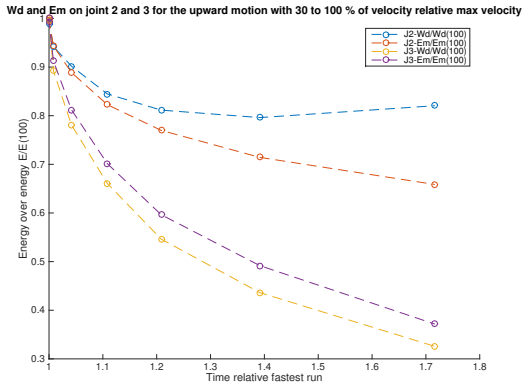


Figure 6.18: The heat dissipated energy and the mechanical work for the upward motion plotted over the maximum energy.

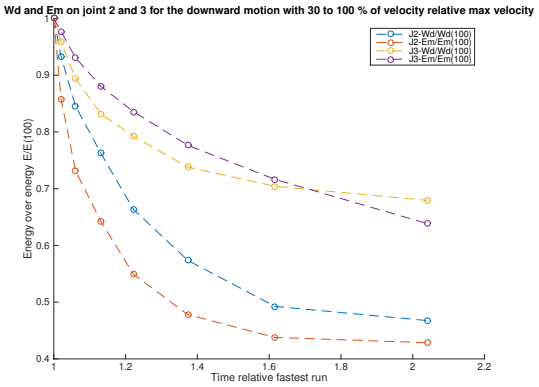


Figure 6.19: The heat dissipated energy and the mechanical work for the downward motion plotted over the maximum energy.

In Figure 6.22, 6.23, 6.24, 6.25 the heat dissipated power and the mechanical power is presented for the up- and downward motion respectively. The heat dissipated power does not decrease as distinctly as the mechanical power for the upward motion. The plots of the upward heat dissipated power are similar to the mirror image of the plot for the downward heat dissipated power, indicating that the heat loss is similar for deceleration in an outreached position and acceleration up from the same position. The same relationship is not found in the mechanical work for the up- and downward motion.

6.5 Summary of the point-to-point evaluation

The initial test verifies that the logger and the ASCII-program is working properly. The evaluation of PTP motions showed that a reduction of velocity or acceleration also reduces the heat dissipated energy and the mechanical work. The electrical current and the heat dissipated energy show a distinct relation to the acceleration profile of the motion. The rotational motion with decreased velocity have all identical initial acceleration profiles independently of desired maximum velocity. Furthermore, the up- and downward motion show that the gravitation negatively affects the energy savings as the downward motion showed better savings for reducing velocities than the upward motion with reducing velocities. For both the rotational and the up- and downward motion the robot shows a desire to, as fast as possible, reach the defined maximum velocity, maintain this velocity for as long as possible and then abruptly decelerate and stop. The results indicates that energy savings through energy optimization of the trajectories are feasible.

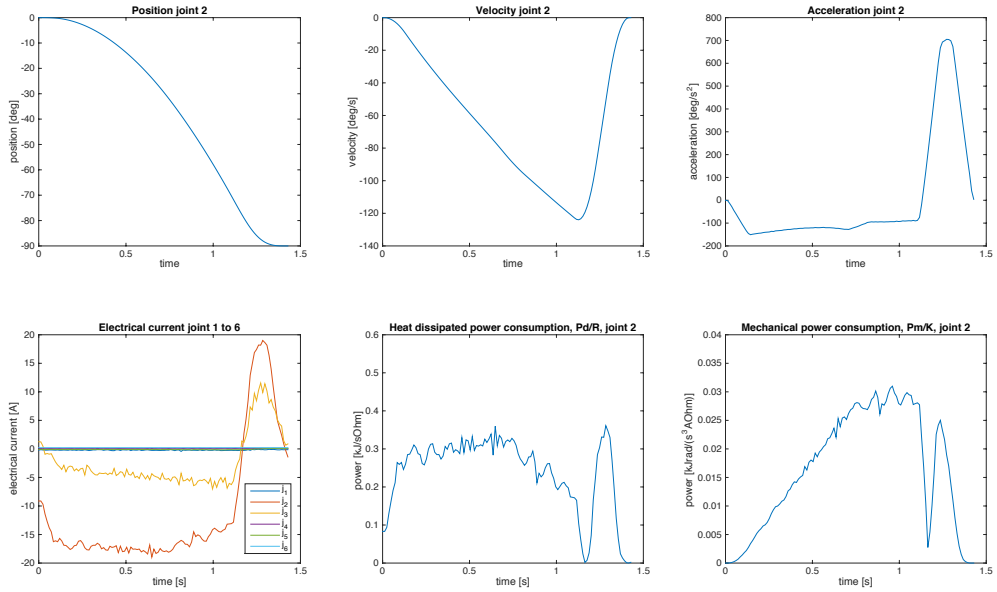


Figure 6.20: An overview of the upward motion showing the position, velocity and acceleration for joint 2 as well as the electrical current, the heat dissipated power consumption and the mechanical power consumption.

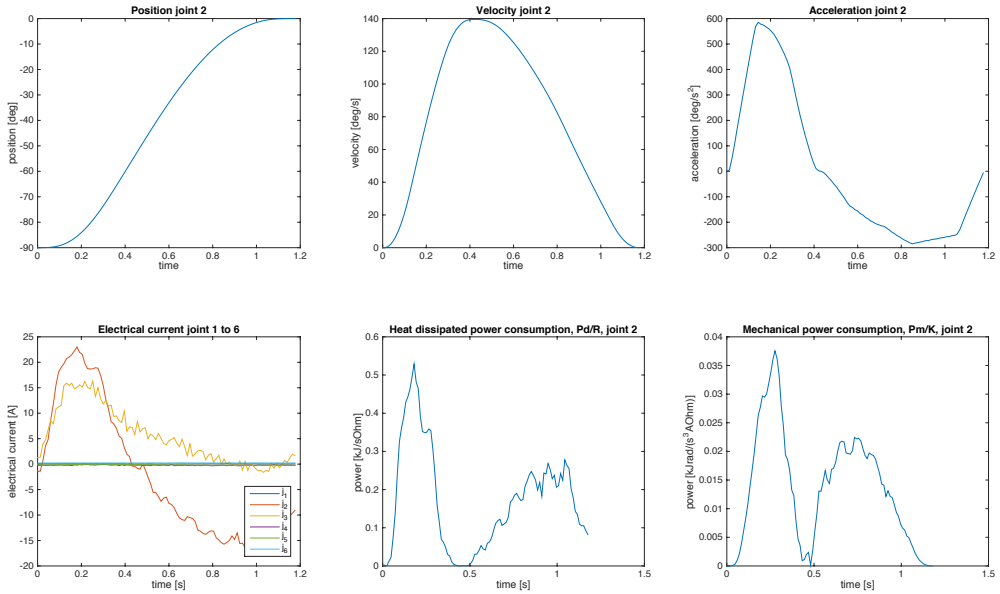


Figure 6.21: An overview of the downward motion showing the position, velocity and acceleration for joint 2 as well as the electrical current, the heat dissipated power consumption and the mechanical power consumption.

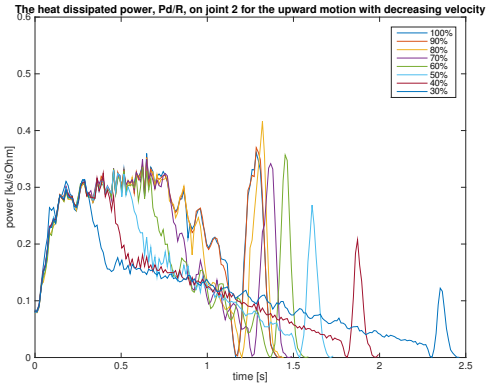


Figure 6.22: The heat dissipated power for the upward motion with decreasing velocity.

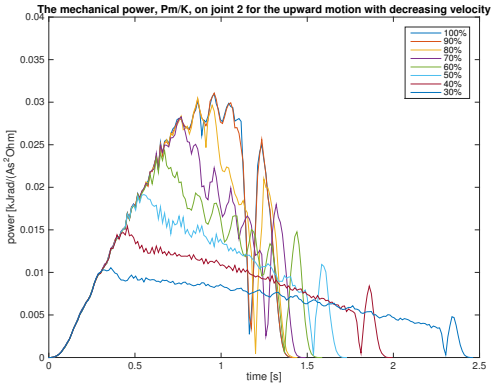


Figure 6.23: The mechanical power for the upward motion with decreasing velocity.

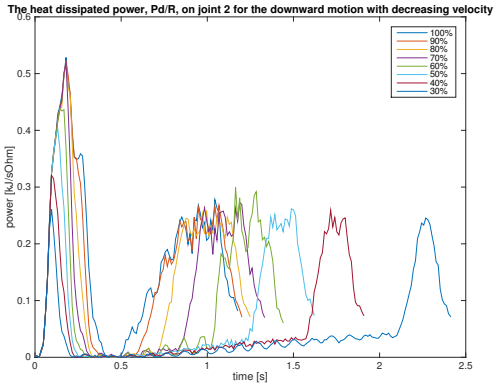


Figure 6.24: The heat dissipated power for the downward motion with decreasing velocity.

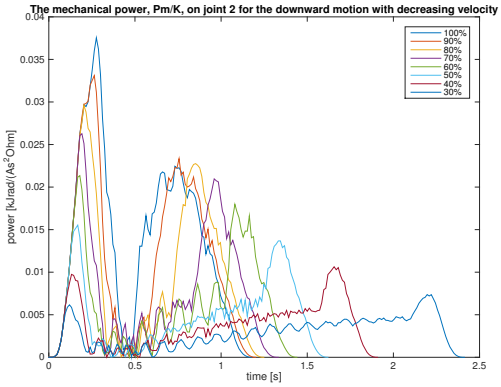


Figure 6.25: The mechanical power for the downward motion with decreasing velocity.

7

Evaluation of optimization strategy

This chapter provides information about the evaluation of optimization criterion and the resulting trajectories. Initially, the motions created in the evaluation of point to point motions are used as original paths for optimization and comparison. Furthermore, a more complex motion, mimicking a pick-and-place operation with several points exciting all joints, is also created and optimized to show the result of a more realistic situation. Finally the upward and downward motions are used to evaluate the energy consumption and optimization of a working cell with multiple robots and shared zones.

7.1 Evaluation of optimization criterion

Four different optimization criterion are formulated and evaluated for comparison. The algorithm in chapter 3 is used for the optimization with small changes in the optimization problem to fit it for the different optimization criterion. The criterion, C_i, evaluated are; minimizing the squared acceleration, minimizing the squared jerk, a combination of minimizing the squared acceleration and jerk and minimizing a model representing the electrical current by a combination of acceleration, velocity, jerk along with a gravitational term modeled as a sine curve.

$\boldsymbol{C_1}$: Simple acceleration

$\boldsymbol{C_2}$: Simple jerk

$\boldsymbol{C_3}$: Combined acceleration and jerk

$\boldsymbol{C_4}$: Combination of acceleration, velocity, jerk and sinusoidal gravity

To evaluate the optimization criterion the rotational motion from chapter 6.3 with 50, 70 and 90 percent velocity are set as original trajectories and the related path and execution times are used as inputs to the optimization algorithm. To not cause confusion in the comparison later the original motions are named Rot50, Rot70 and Rot90. The optimized trajectories are generated with respect to the four criterion $C_1 - C_4$.

The heat dissipated energy, $W_{d,1}$ and the mechanical work, $E_{m,1}$ for each of the optimized trajectory are compared to the original trajectory with the same execution time in figure 7.1 and 7.2 respectively. The energy percentage for the optimized trajectories relative the original trajectories are also stated in table 7.1. All four minimization criterion show a reduction in heat dissipated energy but, while criterion C_1 and C_4 both show a reduction in the mechanical work, criterion C_2 and C_3, including jerk, increase the mechanical work compare to the original trajectory. Given the slightly better reduction of energy for C_1 minimized acceleration, this criterion is chosen for further optimization and evaluation.

Looking closer at the results of the acceleration minimization criterion, C_1, in Table 7.1, it shows that a reduction in velocity gives an increased heat dissipated energy saving (from 20 to 29 percent) and a decreased mechanical work reduction (from 14 to 6 percent) compared to the original trajectories. This is due to the minimization criterion. The close relationship between electrical current and acceleration decreases the heat dissipated energy while the relationship between the mechanical work and the velocity is not accounted for in the optimization. This explains the better result for the heat dissipated energy compared to the mechanical work for the original trajectories with lower velocity settings. Firstly the relative velocities are increased for optimized trajectories with longer execution time (Rot50 compared to Rot90). Figure 7.3 show the velocity profiles of the optimized trajectories with minimized acceleration compared to the original trajectories Rot50, Rot70 and Rot90. The optimized velocity is 31 percent higher than the original velocity in Rot50, compared to only 11 percent higher for Rot90. As a result the acceleration is smoother, reducing the heat dissipated energy, while the increased relative velocity lowers the mechanical work reduction.

Given the savings of both the heat dissipated energy and the mechanical work, the total energy saving lie somewhere in between. That is, for the trajectory with 90 percent velocity the total energy saved with the optimization is somewhere between 14-20 % while the saving lies between 6-29% for the trajectory with 50 percent velocity. Figure 7.4 give the position, velocity, acceleration as well as the electrical current, heat dissipated power and the mechanical power for the Rot90 original and optimized trajectory. As seen in this figure, as well as in Figure 7.3 the optimized velocity achieve parabolic profiles instead of the original flat topped profile, allowing for a better acceleration profile and a less energy consuming trajectory.

In the figures for the electrical current, as well as in the figures for heat dissipated power and the mechanical power, more oscillation is observed for the optimized trajectory compared to the original one, especially in the acceleration phases. No mechanical vibrations have been observed and when comparing the oscillations to the result of lowered acceleration for the rotational motion in the PTP-evaltuation in figure 6.15 and 6.14 a similar oscillation is also observed for lower acceleration reducing the concerns about harmful behavior, but the effects might still need to be investigated further.

Table 7.1: Comparison of energy consumption for the original and optimized rotational movement with 50, 70, 90 percent velocity.

Trajectory	$\boldsymbol{S_{W_d,1}^{C_1}}$ [%]	$\boldsymbol{S_{W_d,1}^{C_2}}$ [%]	$\boldsymbol{S_{W_d,1}^{C_3}}$ [%]	$\boldsymbol{S_{W_d,1}^{C_4}}$ [%]	$\boldsymbol{S_{E_m,1}^{C_1}}$ [%]	$\boldsymbol{S_{E_m,1}^{C_2}}$ [%]	$\boldsymbol{S_{E_m,1}^{C_3}}$ [%]	$\boldsymbol{S_{E_m,1}^{C_4}}$ [%]
***Rot*90**	20	9	9	20	14	-3	-3	11
***Rot*70**	25	7	8	24	9	-13	-13	7
***Rot*50**	29	18	18	29	6	-9	-9	5

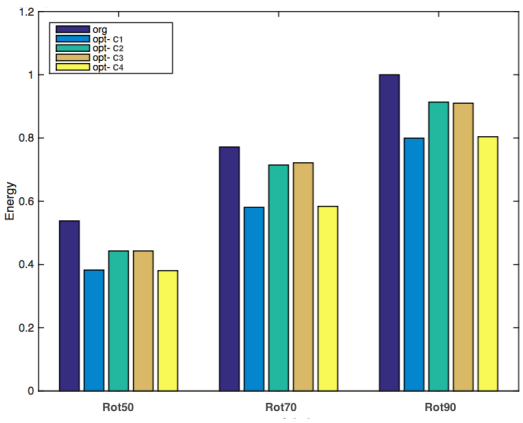


Figure 7.1: The heat dissipated energy consumption, $W_{d,1}$, for the original and optimized trajectories for the rotational motion with 50, 70 and 90 percent velocity normalized over the original Rot90 motion.

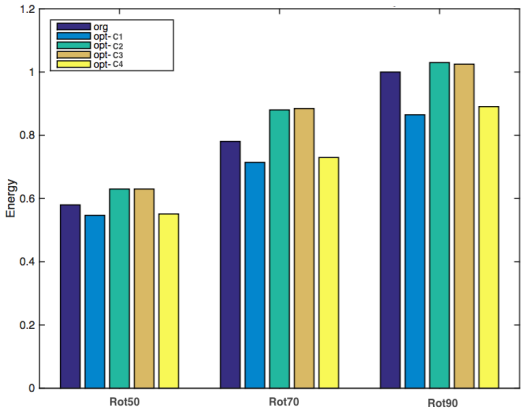


Figure 7.2: The mechanical work, $E_{m,1}$, for the original and optimized trajectories for the rotational motion with 50, 70 and 90 percent velocity normalized over the original Rot90 motion.

7.2 Optimized pick-and-place operation

The initial experiment was performed with a single point-to-point movement and only exciting one joint. As the experiment viewed positive results a longer trajectory with multiple points, movements and velocities is generated. The trajectory is created to mimic a pick-and-place sequence normal for an industrial robot. The robot moves from home position to a pickup zone, picks up a part and transports it to the drop off zone through a continuous point, added to avoid collision with an obstacle. The robot drops off the part and returns to home position through yet another continuous point. It enters and exits the pick up and drop off zone in a linear fashion. The sequence of motions can be seen in Table 7.2. The original trajectory is logged and used in the optimization algorithm. The comparison in energy for joint 1 to 6 can be seen in Figure 7.5 and 7.6 and Table 7.3. As seen in the table, joint 1-3 show an energy saving between 13-55% while joint 4-6 show little to no energy savings (0-16%). Firstly, it should be mentioned that there is nearly no motion, and therefore also very low energy consumption, in joint 4-6, as seen in Figure 7.5 and 7.6. Secondly, an optimization containing several joint have a priority in the algorithm since the optimization of one joint can contradict the optimization of another joint. This is solved with a weighting between joints, prioritizing the optimization. For the industrial robot the first three joints are actuated with larger engines compared to the last three joints. Assuming they affect the overall energy consumption greater than the last three, the optimization of joint 1-3 is prioritized higher than joint 4-6, also resulting in a better optimization result for joint 1-3. If comparing the energy reduction on joint 1-3 the result is significantly better for joint 1 and 3 compared to joint 2. This might be due to the overall lower movement of joint 2 compared to the other two or that the gravitational force on joint 2 allows less optimization.

With these considerations accounted for the optimization shows very promising results. Assuming that joint 1-3 are actuated with identical engines the total heat dissipated energy savings and the total mechanical work reduction for joint 1-3 is 38 and 26 percent respectively.

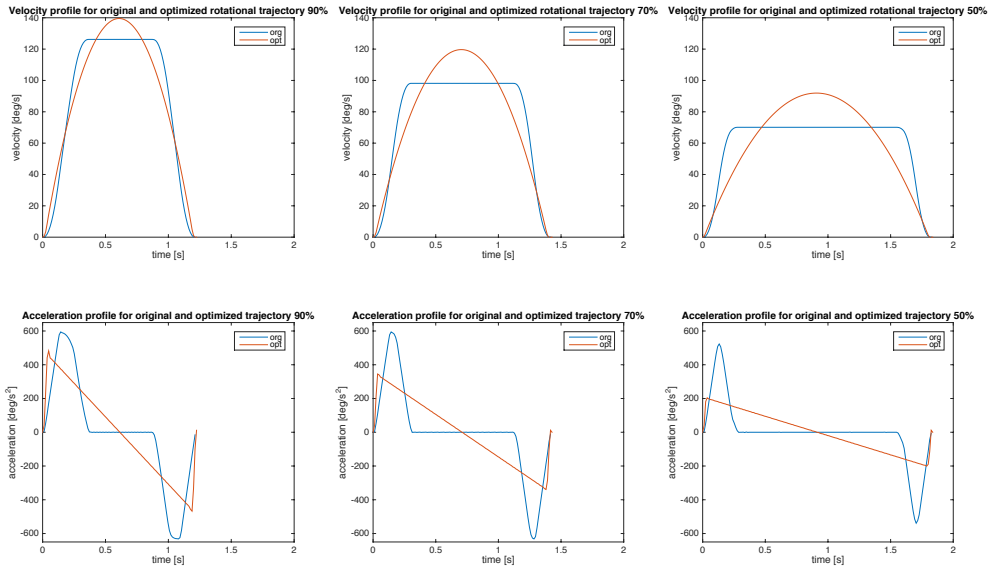


Figure 7.3: The velocity and acceleration profiles for the original and optimal trajectories for the rotational motions Rot50, Rot70 and Rot90.

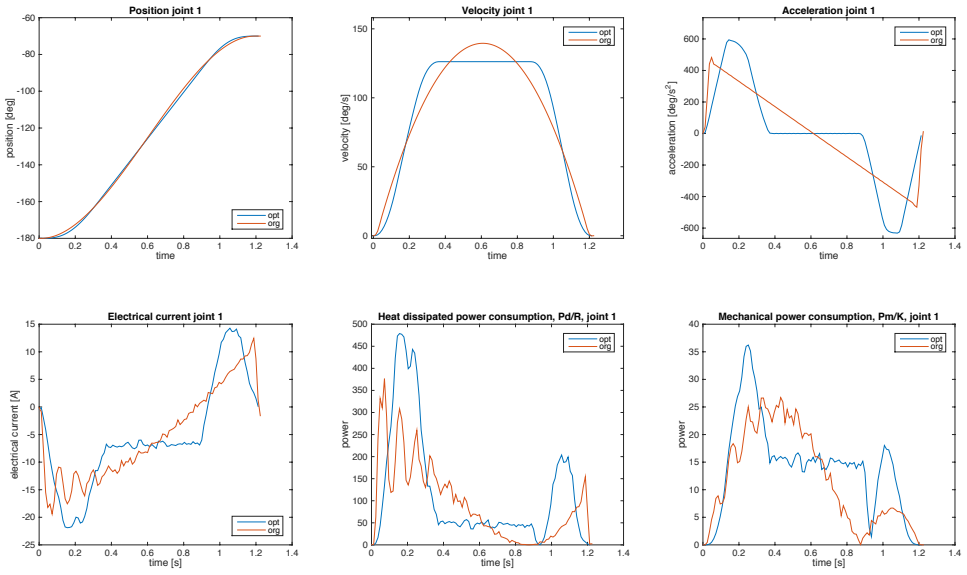


Figure 7.4: The position,velocity, acceleration as well as the electrical current, heat dissipated power and mechanical power for Rot90 original and optimized trajectory.

Table 7.2: The movement sequence for the pick-and-place trajectory

KRL			Joint position					
Motion	**Point**	**Settings**	**1**	**2**	**3**	**4**	**5**	**6**
PTP	**HOME**	100 %	-160	-90	90	0	-90	0
PTP	**P1**	100 %	-180	-95	95	0	-90	0
LIN	**P2**	2 m/s	-180	-80	80	0	-90	0
LIN	**P1**	2 m/s	-180	-95	95	0	-90	0
PTP	**P3**	100 % cont	-130	-95	40	0	-90	0
PTP	**P4**	70 %	-50	-100	110	40	-90	0
LIN	**P5**	2 m/s	-60	-85	100	40	-90	10
LIN	**P4**	2 m/s	-50	-100	110	40	-90	0
PTP	**P6**	50 % cont	-90	-85	110	0	-110	10
PTP	**HOME**	100 %	-160	-90	90	0	-90	0

Table 7.3: The energy reduction for the optimized pick-and-place operation with respect to the original trajectory, joint 1-6. Execution time $T_f = 6.9s$.

Joint	**1**	**2**	**3**	**4**	**5**	**6**
$\boldsymbol{S_{W_d}}$ [%]	55	15	44	16	8	0
$\boldsymbol{S_{E_m}}$ [%]	34	13	32	8	5	0

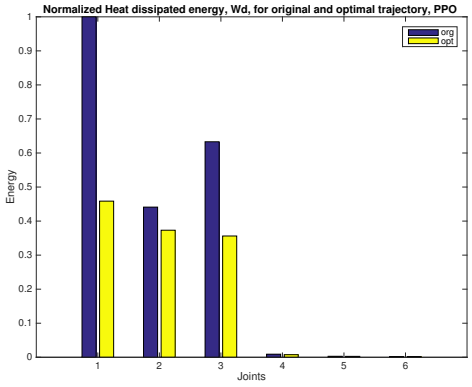


Figure 7.5: The heat dissipated energy consumption, W_d, for the original and optimized trajectories for the pick-and-place operation normalized over joint 1.

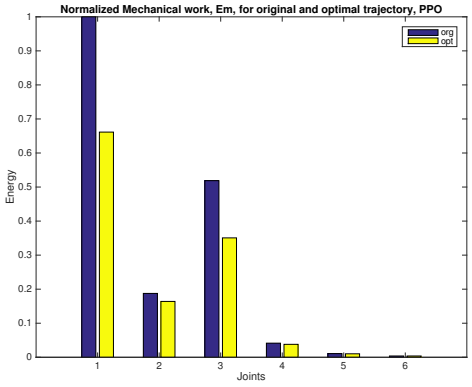


Figure 7.6: The mechanical work, E_m, for the original and optimized trajectories for the pick-and-place operation normalized over joint 1.

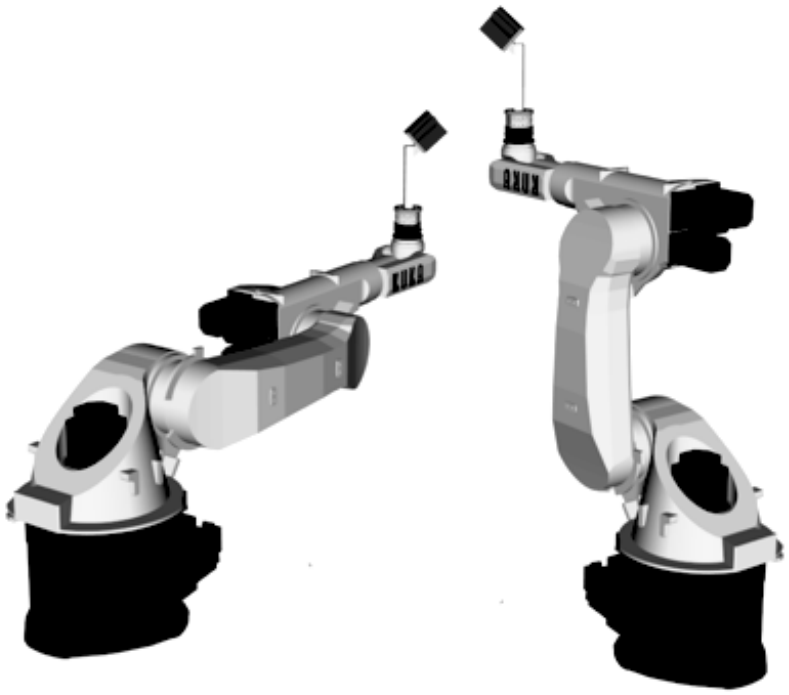


Figure 7.7: The cell with robots R1 and R2

7.3 Multi-robot movement

Finally a two-robot cell with a shared zone is created and evaluated. The cell can be seen in Figure 7.7. Both robots perform one motion each, robot 1 (R1) is moving from an upright vertical position down to an outreached horizontal position using joint 2 and 3 and robot 2 (R2) makes the reversed motion, starting in the outreached horizontal position and move up to the vertical upright position. If they move at the same time they will collide as they share workspace. The shared zone is defined as the area where joint 2 is between -30 and -60 degrees and joint 3 simultaneously is between 30 and 60 degrees for both robots.

The original motion is created with KRL giving two possible sequences to avoid collisions. Sequence 1 (S1) where R1 is moving before R2 and sequence 2 (S2) where the order is reversed and R2 is moving before R1. Both sequences are logged and evaluated. The logged original trajectories are used as input to the optimizer and the optimized sequences are also evaluated.

In addition to optimizing the two sequences where only one robot is moving at a time, two sequences are also generated where the robots, instead of standing still and waiting for the other to finish its task, are allowed to move simultaneously without colliding. The first robot perform its task as before (R1 in S1 and R2 in S2) but the second robot in each sequence is optimized such that it is allowed to start its motion at once and only have to ensure that the first robot have left the shared zone before entering the same zone. In this way the execution time for the second robot will increase to the total time of both original motions added the zone-constraint defined in Equation 3.13 in Chapter 3, preventing it from entering the shared zone before the first robot has exit.

The heat dissipated energy and the mechanical work for the two robot motions, can be seen

Table 7.4: The heat dissipated energy saving and the mechanical work reduction on joint 2 and 3 for sequence 1 and 2 and the original (org), optimal sequential (opt-seq) and optimal non-sequential (opt) trajectories. The energy is displayed in relationship to $S1_{org,2}$ and $S1_{org,3}$ for joint 2 and 3 respectively.

		S_{W_d}			S_{E_m}		
		org	***opt − seq***	***opt***	***org***	***opt − seq***	***opt***
Joint 2	**S1**	0	17	27	0	20	28
	S2	26	43	47	0	20	35
Joint 3	**S1**	0	20	33	0	17	31
	S2	0	20	33	0	17	27

in Figure 7.8 and 7.9 and Table 7.4 where the original motions are named $S1_{org}$ and $S2_{org}$, the optimized sequential motions are named $S1_{opt}^{seq}$ and $S2_{opt}^{seq}$ and the final less constraint optimized motions are named $S1_{opt}$ and $S2_{opt}$. In the figures the first two sets of bars show the energy on joint 2 while the two last sets of bars show the energy for joint 3. Each set of bars show the original trajectory, the sequential optimized and the fully optimized trajectory respectively.

If looking at Figure 7.8 the first two sets of bar show the heat dissipated energy for joint 2 for sequence 1 and sequence 2 respectively. Significant energy can be saved depending on the sequence of the motions. If comparing the two original motions $S1_{org,2}$ and $S2_{org,2}$ the later is more energy efficient. The difference is due to the position the robots wait in. In sequence 1 both robots are waiting for the other to finish in an outreach position compared to sequence 2 where the robots wait in a less demanding upright position. This can also be seen in Figure 7.10 and Figure 7.11 where the heat dissipated power for the waiting phase is larger for S1 than S2. The position in which the robots are waiting is only affecting the heat dissipated energy as the mechanical work is directly related to the velocity, equaling it to zero when the robot is standing still, as seen in the velocity and mechanical work plots in Figure 7.10 and 7.11. Also, the mechanical work for the original and the sequential optimized motions are identical in sequence 1 and 2, as seen in Figure 7.9. This is reasonable since the only difference between the two sequences are the waiting position which is not affecting the mechanical work.

When looking at the sequences for the sequential optimized trajectories, $S1_{opt}^{seq}$ and $S2_{opt}^{seq}$, there is an improvement for both the heat dissipated energy and the mechanical work, compared to the original trajectories. But while the mechanical work saving does not differ for the two sequences, the heat dissipated energy saving is quite larger for sequence 2 compared to sequence 1, again a consequence of the waiting position.

Finally when comparing the fully optimized path to the original path, sequence 2 show larger savings than sequence 1 due to the second robots early start. As seen in the upper rightmost plot in Figure 7.11, both robot start moving at the same time and the second robot enters the shared zone slightly after the first robot has left it. The early start of the second robot allows it to have a softer acceleration and therefore save energy.

It should also be mentioned that the small difference for sequence 1 and 2 for joint 3 is most likely due to the identical resting position of joint 3 for the two sequences. Still, the full optimization show an energy reduction of 27-33% for joint 3.

Consequently it can be said that by choosing the right sequence and optimizing the sequence with respect to a shared zone as much as 35-47% can be saved.

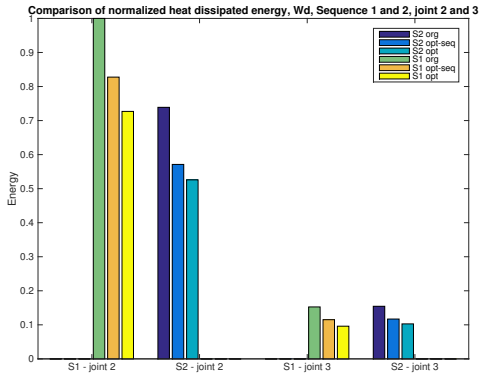


Figure 7.8: The heat dissipated energy consumption, $W_{d,1}$, on joint 2 and 3, for the two sequences S1 and S2 normalized over S1:org.

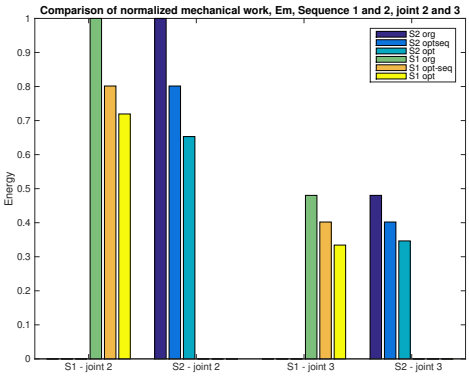


Figure 7.9: The mechanical work, $E_{m,1}$, on joint 2 and 3, for the two sequences S1 and S2 normalized over S1:org.

7.4 Summary of the Optimization Evaluation

The evaluation of optimization criterion show that a minimization of acceleration give a sufficient energy saving and allow for rapid and simple optimization, making it ideal to implement in existing robot programs. The optimization algorithm shows reduction in energy for both heat dissipated energy as well as the mechanical work for all trajectories. For the pick-and-place operation the reduction is as large as 26-38% over joint 1-3, assuming identical actuation engines and as large as 33-55% if only looking at joint 1. The experiment with two robots shows that the order of operations highly affect the energy consumption as well as a shared zone optimization, allowing the robots to start early to get a smoother acceleration. The optimized trajectory achieve an parabolic velocity profile, compared to the flat topped original velocity profile. These profiles are crucial for the optimization.

To implement optimized trajectories in an industrial robot ascii-files have been used in these experiments but it would be desirable to use existing trajectory planning tools why PTP splines are investigated next.

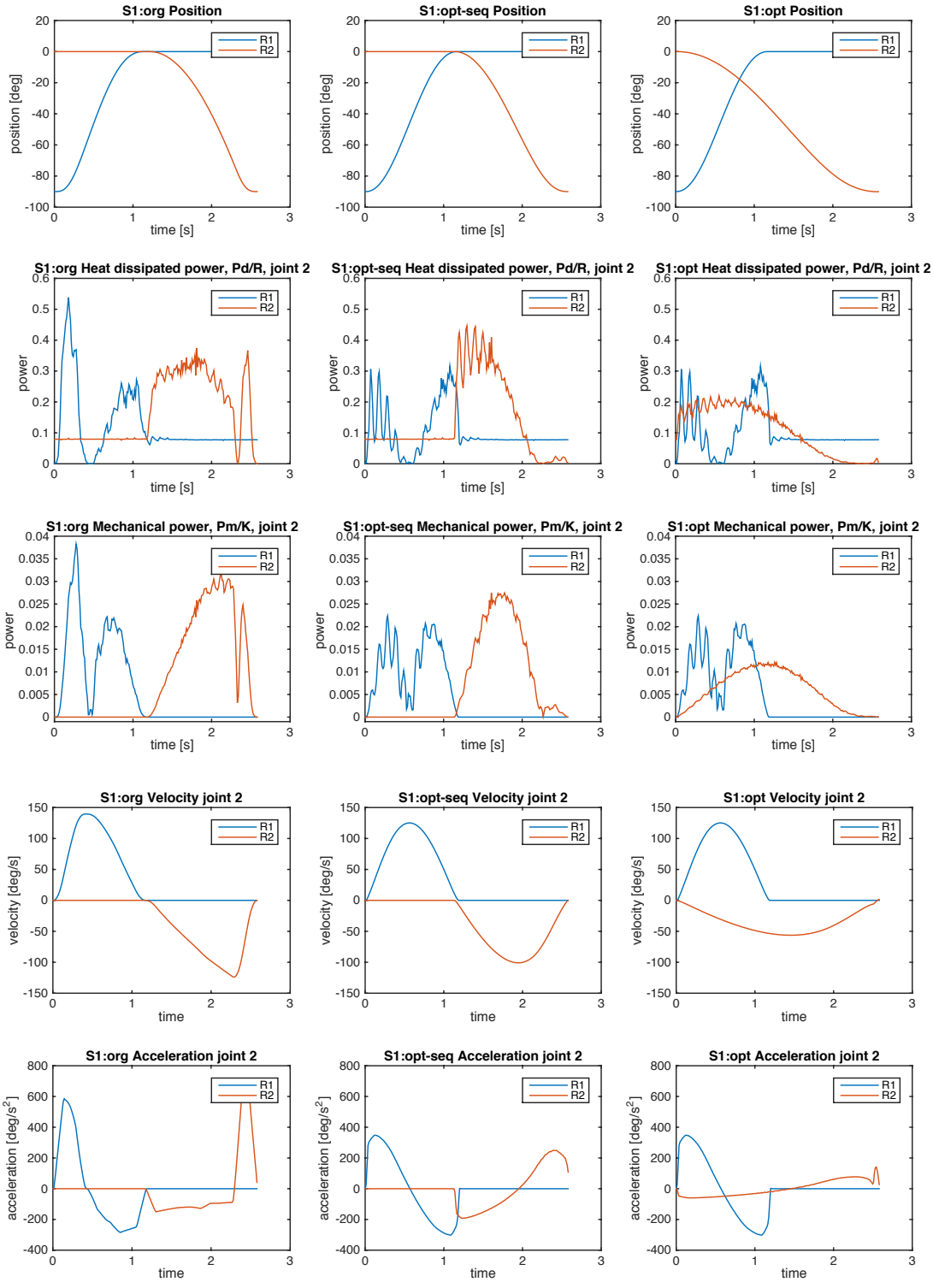


Figure 7.10: Position, heat dissipated energy, mechanical work, velocity and acceleration for Sequence 1. From left to right: original, sequential optimized and optimized trajectory respectively.

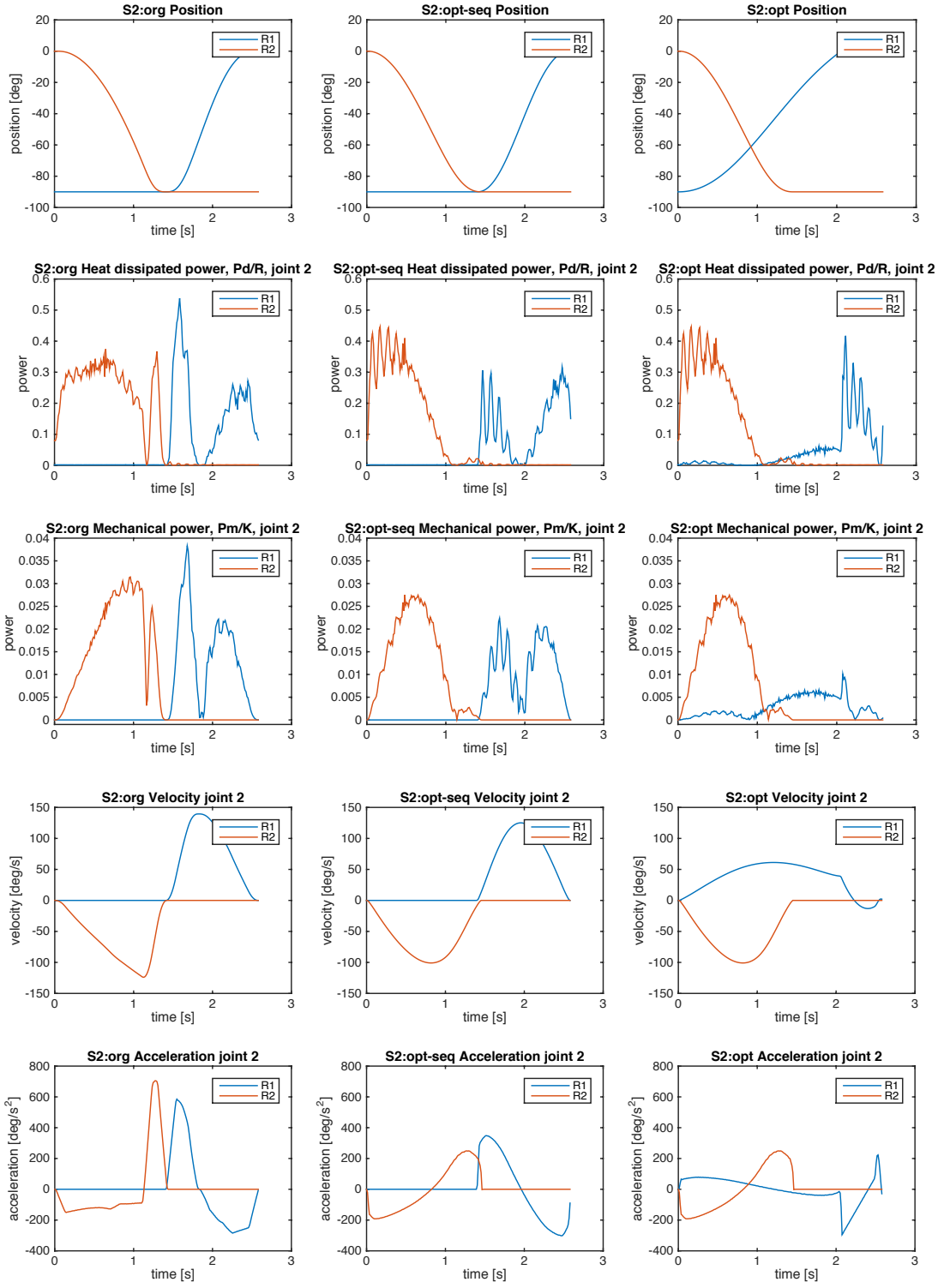


Figure 7.11: Position, heat dissipated energy, mechanical work, velocity and acceleration for Sequence 2. From left to right: original, sequential optimized and optimized trajectory respectively.

8

Evaluation of Splines as an implementation method

In this chapter the functionality and behavior of SPLINE and time blocks are investigated to determine if they are suitable as implementation tools for the optimization algorithm.

8.1 The implementation hypothesis

Giving the satisfying energy optimization it is desirable to implement the algorithm on robots already installed in the industry. It is desirable to use an online approach for the implementation. This means that an operator creates a trajectory as usual using the KRL. As the robot executes the trajectory a plugin installed in the robot records the path and optimizes it. For the following executions the plugin take over and the optimized trajectory replaces the original.

For the experiments the optimized trajectories have been implemented using an ascii-file but for the real industry this approach bring some issues. When implementing a trajectory with an ascii-file the trajectory can not be interrupted or manipulated and the trajectory does not take any inputs or outputs into considerations, which all are crucial features. It would therefore be desirable to use existing programming tools for the implementation. Based on the result from the investigation of PTP motions in Chapter 6, it seems like the parabolic velocity profile connected with the optimized trajectories in Chapter 7 are not possible to generate using PTP motions. Instead, one other option is the motions SPLINE and SPLINE blocks present in software KRC4. In the ASCII-program the velocity and acceleration is controlled continuously by the sequential definition of position and time. A SPLINE block, defining the path, can be controlled using time blocks and thereby might allow for the control of velocity and acceleration in a fashion similar to the ASCII program.

Given these conditions it is of interest to understand how the robot implement spline blocks and time blocks and how these can be manipulated. The hypothesis is that a trajectory can be manipulated in a spline block using time block and intermediate time block parts. By adding intermediate points with time block parts along the path, the trajectory can be controlled and manipulated in a desired way. To control the manipulation it is important that only the added time block parts and not the added points affected the trajectory.

Table 8.1: The points defined along the path for the different experiments

	Point values											
Setup 1	60	-50										
Setup 2	60	50	-50									
Setup 3	60	50	40	30	20	10	0	10	20	30	40	50
Setup 4	60	0	-5	-10	-12	-15	-22	-30	-31	-38	-40	-50

The spline blocks are PTP spline blocks and the tests have been conducted on the KUKA KR16.

8.2 Experimental setup

To be able to use spline blocks as implementation tool it is important to be able to divide a given path into smaller fractions to later be able to define the velocity in those fractions using time blocks. It is therefore crucial that a path is not affected by the number of points along it as long as the velocity, or time stamp, is not manipulated. To investigate how the number of points along a path, defined within a spline block, affect the trajectory, an experiment is conducted through a single rotational motion where joint 1 is moving from 60 to -50 degrees. The motion is implemented with four different settings. In the first experiment the spline block contains only a start and an end point. In the second experiment one point is added along the path, close to the starting point. In the third experiment ten points are evenly distributed between the starting and finishing point and in the forth experiment ten points are distributed randomly along the second half of the path. The points can be seen in Table 8.1 and the full code for each setup can be found in appendix A. The resulting trajectories are recorded using the logger.

8.3 Results

A comparison of the four trajectory show a negative result. Firstly, the final time are different for the four trajectories.

$$t_f = [1.1880 \quad 1.1520 \quad 1.1280 \quad 1.1640]$$

Secondly, an overview of the position, velocity, acceleration and jerk for all trajectories can be seen in Figure 8.1. The trajectories are clearly changing depending on the number of points in the spline block and also depending on the distribution of the points. When the trajectory with only an initial and final point is compared to the other three trajectories the following observations are made.

With only a initial and final point the trajectory experience an overshoot and move off the desired path before converging towards the final value. When adding a point close to the initial point in the spline block the trajectory changes. The overshoot decreases and the robot moves in the right direction earlier. The two trajectories can be seen in Figure 8.2, where the third and fourth plots gives a detailed view of the position and velocity for the overshoot.

Adding ten points evenly distributed every ten degrees along the path alters the trajectory again. The trajectory is similar to the trajectory for one intermediate point as seen in Figure 8.3.

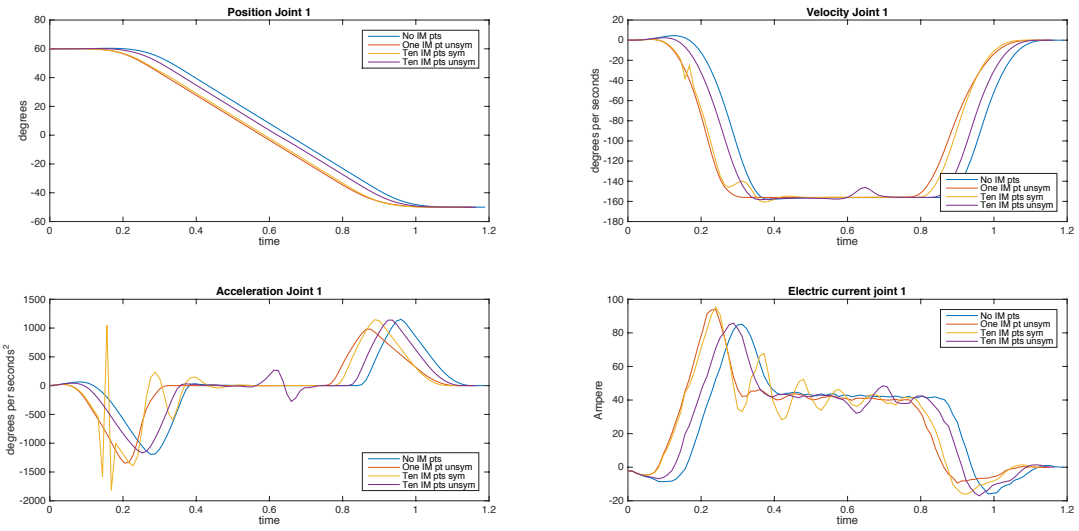


Figure 8.1: An overview of the trajectories for four different settings in a spline block.

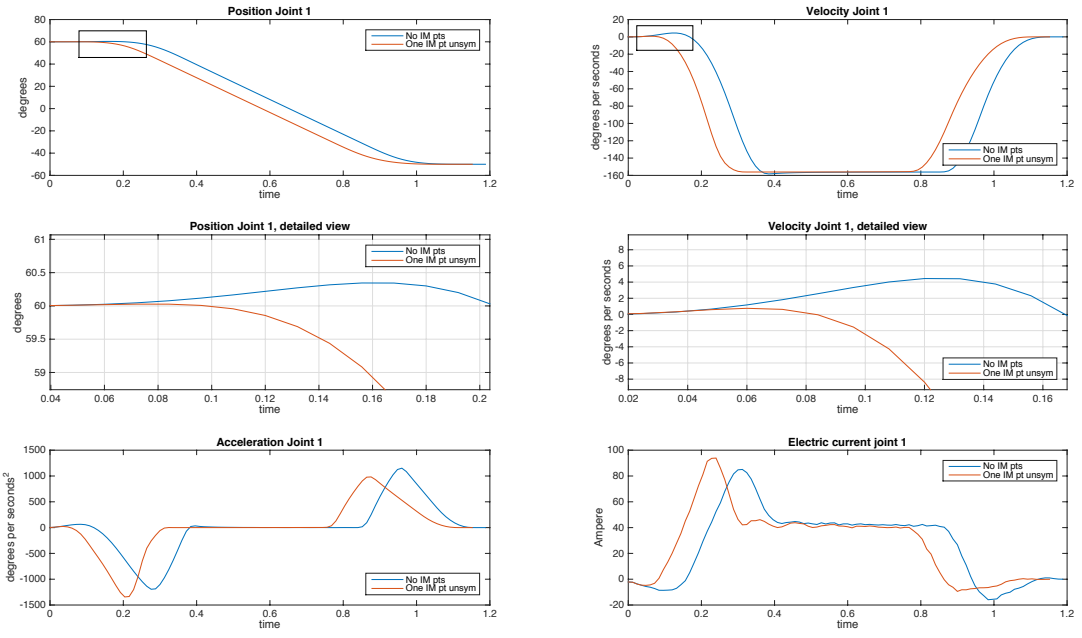


Figure 8.2: The trajectory with only one initial and final point compared to the trajectory where one point has been added between at J1=50 degrees.

Around time instant 0.15 seconds something happens and acceleration and velocity experience spiked values. The effect on the path can be seen in plot three in Figure 8.3.

In the last test ten points are added to the spline block with values between 0 and -50 degrees. The points are unevenly distributed along this second half of the path. The trajectory settle somewhere between the trajectory with only an initial and final point and the other two. The trajectory is compared to the first trajectory in Figure 8.4. It gets an overshoot larger than the one for the initial trajectory as seen in plot 3 in Figure 8.4. Also, right before the first intermediate point is reached at 0 degrees the robot decelerate and then accelerate. This can be seen in the velocity and acceleration profile at time instant 0.6. This ”bump” does not seem to affect the trajectory though.

8.4 Summary of the splines evaluation

Since the trajectory is changing with the number of points in the spline block, and not in an orderly fashion, no investigation of time block was conducted. The path changes following the addition of a point is unpredictable and therefore hard to compensate for. The spline blocks are, for now, concluded to not be useful for implementation of the optimized trajectories.

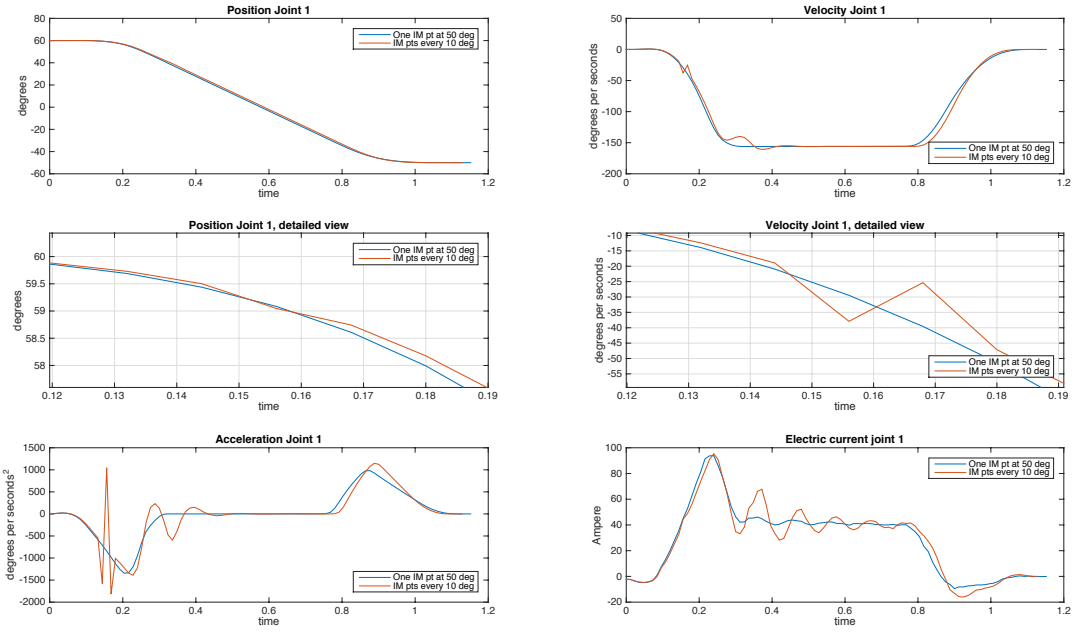


Figure 8.3: The trajectory with one intermediate point at 50 degrees compared to the trajectory with 10 points evenly distributed every 10 degrees along the path.

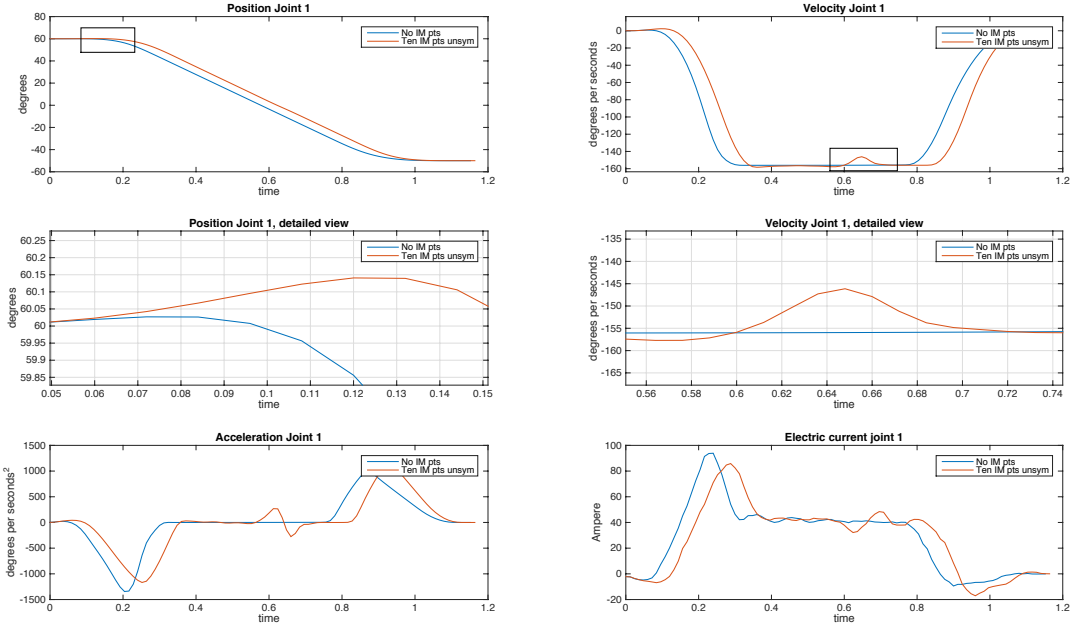


Figure 8.4: The trajectory with only an initial and final point compared to the trajectory with 10 points unevenly distributed along the path.

9

Conclusions

The result from the evaluations is discussed with respect to the objectives of the thesis. Remarks are made on the results and based on the conclusions, recommendations for future work are given.

9.1 Repeating the objectives

The objectives, earlier stated in the introduction, are

- Create an experimental setup, allowing evaluation of trajectories including collection of the energy consumption.
- Experimentally evaluate the trajectory planning strategies available in the robot today
- Implement and evaluate optimization strategies to determine optimization criteria reducing the energy consumption
- Investigate the possibilities for the integration of an optimization strategy in existing technology

9.2 Create an experimental setup

An experimental setup was successfully created with help of the personal at KUKA Gothenburg. By installing the ASCII program and the logger, optimized trajectories could be executed and relevant data collected for later analysis. The setup can be improved by further development of the data collection also adding alternative methods for measuring the energy consumption. A comparison of an ascii file and the logged results from the execution of the same ascii-file verified that the logger and the ASCII-program are working properly.

9.3 Energy calculations

The unknown relationship between the heat dissipated energy and the mechanical work, as well as the contribution between joints made it impossible to calculate the exact energy savings. If

wanted, the relationship between the two can be discussed and there might be ways to make an estimation of the relationship. For example, assuming the efficiency of the engine is approximately 90% the relationship between the heat dissipated energy and the mechanical work could be believed to be somewhere in the ration of 1:10. This could be further investigated though, through simulated robot models and better measurements.

It should also be mentioned that there most likely are other terms of energy losses not accounted for which might throw the results of, but given the assumptions and simplifications made, the result is still believed to give a reasonably correct picture of the energy reduction. Still, to verify the results, some complementary measurements should be done. As a simulation tool for a robot cell, developed by KUKA, now exists at Chalmers it is recommended to verify the results in this thesis, as well as future calculations from experiments, using the simulation. Furhtermore, KUKA has developed new technology to more accurately measure the energy in the robot. By installing these tools the experiments will be more reliable.

9.4 Evaluate the existing trajectory planning strategies

The evaluation of the existing trajectory planning method in the robot gave a great understanding and shows there is much room for improvement. The uniform acceleration profile of the rotational motion, along with the robot's desire to reach a constant velocity, is useful for manufacturing applications where it is important to reach and maintain a constant velocity. This kind of behavior would be reasonable to find in LIN and CIRC motions but unnecessary in PTP motions, where the shape of the velocity profile is of less importance. This leaves room for optimization and energy savings.

Even without the optimization algorithm the energy consumption can be reduced by allowing a slightly longer execution time. By reducing the acceleration, a small increase in execution time and an relatively large reduction in energy consumption is obtained. For an example, by reducing the acceleration to 60 % the execution time is increased with 14 percent while the energy reduction lies between 15-31% (Table 6.2). This is a quick and simple initial step towards a more sustainable industry. The effects of an acceleration reduced trajectory has only been evaluated for the rotational motion and the effect of more complex trajectories should probably be further investigated to give a more complete picture.

9.5 Optimization

The optimization showed results exceeding the expectations. It was satisfying to find an optimization algorithm with a minimization criterion requiring such small computational effort. Furthermore, the identification of the trajectory samples related to the entering and exiting of shared zones simplified the optimization of coordination of robots and further increased the energy reduction. The algorithms should be evaluated on more complex systems though, to give a more complete result.

The close relationship between the heat dissipated power, the electrical current and the acceleration explains the impressive results in energy savings using an optimization criterion minimizing the squared acceleration. The energy savings of the mechanical power is good but not in the same league as the heat dissipated savings. As the mechanical work is related to both the electrical current and the velocity an investigation of a minimization criterion of combined squared acceleration and acceleration times velocity would be interesting, to see if this could further improve the reduction of energy.

The optimized trajectories showed some oscillations in the figures showing the electrical current and energy consumption, raising some concerns. Similar increase in oscillations could also be observed in the figures with decreased acceleration in the evaluation of point-to-point motions why it is reasonable to believe such behavior should have been investigated by the robot company and proven not harmful.

For KRL and inline programming the velocity is limited to the 100 % defined in the robot. Trajectories executed through ASCII are allowed to reach velocities higher than the, by robot, predefined 100 percent, as seen in the optimization of the rotational motion. This possibility to reach higher velocities is crucial to optimize a trajectory but the effect of such violation of predefined limits should be investigated further to verify no long term harm, specially considering situations with higher loads. All evaluations have been conducted on robots under no load. The results should be complemented by investigating the optimization algorithm and energy reduction with the robot under load.

9.6 Implementation of optimization

The trajectory planning tools existing in the robot have shown to be hard to manipulate. Independently of the settings the robot executes with the same strategy, to reach the highest allowed velocity and keep it for as long time as possible before breaking. The experiments in Chapter 8 show that spline blocks are not an option to implement the optimized trajectories in the robot as it changes the path with changing number of intermediate points in the block. It is necessary to find a tool which allow a more controlled manipulation of velocity and acceleration than what is present in the robots today. The best option for full control of the trajectory is the ASCII-program. Further work should include investigating the possibilities to allow for signals and interruptions when implementing the optimized trajectory. The problems could possibly be addressed by taking possible interrupts into account in the ASCII-program. Either by some means remembering where in the text-file the interruption occurred and be able to resume the execution from that value, or by saving the remaining part of the text-file to a temporary file to execute after an interruption. Also, trajectories where the robot stops and wait for a signal could possibly be divided into several ascii-files loaded and executed sequentially allowing for I/O signals in between. These approaches need further investigation and consideration.

9.7 Future work

Given the result of this thesis it is realistic to believe that a fully integrated solution is feasible and likely to provide satisfying results. To arrive at such development more effort is needed. Firstly, it is recommended to verify the results in this thesis by conducting experiments on a robot under higher, preferably maximum, load and also by other means of measuring the energy consumption. Secondly, much of the work in this thesis haa been performed manually. For a final solution the work haa to be automated to be implement in the robot as an independent program. Also, the implementation has to be investigated further. As a first step, the possibilities with multiple ASCII programs should be evaluated thoroughly.

Finally, as a side step it should be mentioned that the evaluations in this thesis have mainly been conducted on PTP-motions defined in joint space. It would also be interesting to evaluate other existing motion types such as LIN and CIRC, or even SPLINES.

It is the author's believe that a solution, significantly reducing the energy consumption in existing industrial robots, is feasible, and reachable within a close future.

Bibliography

[1] M. Pellicciari, AREUS-Automation and Robotics for EUropean Sustainable manufacturing, [Online], Available: http://www.areus-project.eu/ (March 2015).

[2] M. Brossog, M. Bornschlegl, J. Franke, et al., Reducing the energy consumption of industrial robots in manufacturing systems, The International Journal of Advanced Manufacturing Technology (2015) 1–14.

[3] D. Meike, L. Ribickis, Energy efficient use of robotics in the automobile industry, in: Advanced Robotics (ICAR), 2011 15th International Conference on, IEEE, 2011, pp. 507–511.

[4] D. Meike, M. Pellicciari, G. Berselli, A. Vergnano, L. Ribickis, Increasing the energy efficiency of multi-robot production lines in the automotive industry, in: Automation Science and Engineering (CASE), 2012 IEEE International Conference on, IEEE, 2012, pp. 700–705.

[5] R. Visinka, Energy efficient three-phase ac motor drives for appliance and industrial applications, L. Goldberg, Green Electronics/Green Bottom Line (1999) 29–42.

[6] S. Nadel, R. Elliott, M. Shepard, S. Greenberg, G. Katz, A. d. Almeida, Energy-efficient motor systems: A handbook on technology, programs, and policy opportunities.

[7] R. Saidur, A review on electrical motors energy use and energy savings, Renewable and Sustainable Energy Reviews 14 (3) (2010) 877–898.

[8] S. Alatartsev, V. Mersheeva, M. Augustine, F. Ortmeier, On optimizing a sequence of robotic tasks, in: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, IEEE, 2013, pp. 217–223.

[9] Y. Crama, V. Kats, J. Van de Klundert, E. Levner, Cyclic scheduling in robotic flowshops, Annals of operations Research 96 (1-4) (2000) 97–124.

[10] A. Kobetski, M. Fabian, Time-optimal coordination of flexible manufacturing systems using deterministic finite automata and mixed integer linear programming, Discrete Event Dynamic Systems 19 (3) (2009) 287–315.

[11] A. Kobetski, M. Fabian, Velocity balancing in flexible manufacturing systems, in: Discrete Event Systems, 2008. WODES 2008. 9th International Workshop on, IEEE, 2008, pp. 358–363.

[12] O. Wigstrom, B. Lennartson, A. Vergnano, C. Breitholtz, High-level scheduling of energy optimal trajectories, Automation Science and Engineering, IEEE Transactions on 10 (1) (2013) 57–64.

[13] A. Vergnano, C. Thorstensson, B. Lennartson, P. Falkman, M. Pellicciari, F. Leali, S. Biller, Modeling and optimization of energy consumption in cooperative multi-robot systems, Automation Science and Engineering, IEEE Transactions on 9 (2) (2012) 423–428.

[14] D. Costantinescu, E. Croft, Smooth and time-optimal trajectory planning for industrial manipulators along specified paths, Journal of robotic systems 17 (5) (2000) 233–249.

[15] J. Park, Motion profile planning of repetitive point-to-point control for maximum energy conversion efficiency under acceleration conditions, Mechatronics 6 (6) (1996) 649–663.

[16] C. Hansen, J. Kotlarski, T. Ortmaier, Experimental validation of advanced minimum energy robot trajectory optimization, in: Advanced Robotics (ICAR), 2013 16th International Conference on, IEEE, 2013, pp. 1–8.

[17] S. Bjorkenstam, D. Gleeson, R. Bohlin, J. S. Carlson, B. Lennartson, Energy efficient and collision free motion of industrial robots using optimal control, in: Automation Science and Engineering (CASE), 2013 IEEE International Conference on, IEEE, 2013, pp. 510–515.

[18] International federation of robotics, [Online], Available: http://www.ifr.org/ (May 2015).

[19] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics- Modelling, Planning and Control, Springer, 2010, iSBN: 978-1-84628-641-4 (Print) 978-1-84628-642-1 (Online).

[20] International Organization for Standardization - Standard 8373:1994, Manipulating Industrial Robots, [Online], Available: http://www.iso.org (2015).

[21] J. J. Craig, Introduction to Robotics - Mechanics and Control, 3rd Edition, Pearson Prentice Hall, 2005, iSBN: 0-13-123629-6.

[22] J. Angeles, Fundamentals of Robotic Mechanical Systems - Theory Methods and Algorithms, 2nd Edition, Mechanical Engineering Series, Springer, 2002, iSBN: 0-387-95368-X.

[23] KUKA Robot Programming - Training Manual (April 2015).

[24] KUKA System Software 5.6 - Operating and Programming Instructions for System Integrators (April 2015).

[25] KUKA System Software 8.3 - Operating and Programming Instructions for System Integrators (April 2015).

[26] B. Lennartsson, K. Bengtsson, AREUS D3.2- Multi-robot scheduling and optimization software tools, AREUS WP3 (Deliverable 3.2).

[27] S. Riazi, K. Bengtsson, O. Wigsröm, E. Vidarsson, B. Lennartsson, Energy optimization of multi-robot systems, Accepted for publication in Automation Science and Engineering (CASE), 2015 IEEE International Conference on.

[28] Selection of servo motors and drives (rev.2), [Online], Available: http://www.drivetechinc.com/articles/pm96sizrev2.pdf (April 2015).

[29] Specification KUKA KR 30-3, [Online], Available: http://www.kuka-robotics.com (April 2015).

[30] Specification KUKA KR 16-2, [Online], Available: http://www.kuka-robotics.com/ (April 2015).

[31] K. Bengtsson, P. Bergagard, C. Thorstensson, B. Lennartson, K. Akesson, C. Yuan, S. Miremadi, P. Falkman, Sequence planning using multiple and coordinated sequences of operations, Automation Science and Engineering, IEEE Transactions on 9 (2) (2012) 308–319.

[32] Delmia V5, [Online], Available: http://www.3ds.com/products-services/delmia/products/V5/ (March 2015).

[33] Visual components 3D create, [Online], Available: http://www.visualcomponents.com (March 2015).

[34] A. Wächter, L. T. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, Mathematical programming 106 (1) (2006) 25–57.

A

The code for evaluating Spline blocks

The following code was used in the four experiments evaluating the SPLINE blocks in KR40.

A.1 KRL - No intermediate points

```
DEF Splines()
DECL axis EHOME
INI
EHOME={ a1 60, a2 -90, a3 90, a4 0, a5 -90, a6 90}
SPTP EHOME Vel=100% PDAT1 Tool[2]: Gripper01 Base [3]:BlueBase1.
startlog("file_name_ log")
PTP SPLINE S1 Vel=100% PDAT1 Tool[2]: Gripper01 Base [3]:BlueBase1
sptp ehome
sptp {a1 -50}
ENDSPLINE
stopLog()
END
```

A.2 KRL - one asymmetric intermediate point

```
DEF Splines()
DECL axis EHOME
INI
EHOME={ a1 60, a2 -90, a3 90, a4 0, a5 -90, a6 90}
SPTP EHOME Vel=100% PDAT1 Tool[2]: Gripper01 Base [3]:BlueBase1.
startlog("file_name_ log")
PTP SPLINE S1 Vel=100% PDAT1 Tool[2]: Gripper01 Base [3]:BlueBase1
sptp ehome
```

```
sptp {a1 50}
sptp {a1 -50}
ENDSPLINE
stopLog()
END
```

A.3 KRL - 10 evenly distributed intermediate points

```
DEF Splines()
DECL axis EHOME
INI
EHOME={ a1 60, a2 -90, a3 90, a4 0, a5 -90, a6 90}
SPTP EHOME Vel=100% PDAT1 Tool[2]: Gripper01 Base [3]:BlueBase1.
startlog("file_name_ log")
PTP SPLINE S1 Vel=100% PDAT1 Tool[2]: Gripper01 Base [3]:BlueBase1
sptp ehome
sptp {a1 50}
sptp {a1 40}
sptp {a1 30}
sptp {a1 20}
sptp {a1 10}
sptp {a1 0}
sptp {a1 -10}
sptp {a1 -20}
sptp {a1 -30}
sptp {a1 -40}
sptp {a1 -50}
 ENDSPLINE
stopLog()
END
```

A.4 KRL - 10 unevenly distributed intermediate points

```
DEF Splines()
DECL axis EHOME
INI
EHOME={ a1 60, a2 -90, a3 90, a4 0, a5 -90, a6 90}
SPTP EHOME Vel=100% PDAT1 Tool[2]: Gripper01 Base [3]:BlueBase1.
startlog("file_name_ log")
PTP SPLINE S1 Vel=100% PDAT1 Tool[2]: Gripper01 Base [3]:BlueBase1
sptp ehome
sptp {a1 0}
sptp {a1 -5}
sptp {a1 -10}
sptp {a1 -12}
sptp {a1 -15}
sptp {a1 -22}
sptp {a1 -30}
```

```
sptp {a1 -31}
sptp {a1 -38}
sptp {a1 -40}
sptp {a1 -50}
 ENDSPLINE
stopLog()
END
```