



Hochschule Bochum
Bochum University
of Applied Sciences



**Faculty of Engineering and Technology
Mechatronics Engineering Department
TEMPUS-JIM2L Master Program**

**“Design and Implementation of a User-Friendly
Platform Interface (UFPI) for Controlling the KUKA
LBR iiwa Robot Via Real-Time Hand Guiding”**

**By
Omar Rateb Shehata**

**Supervisors
Prof. Dr. Rolf Biesenbach
Prof. Dr. Mohammad Baniyounis**

**This thesis was submitted in partial fulfilment of the requirements for
the master’s degree in Mechatronics Engineering**

To Bochum University of Applied Sciences

September / 2023

CERTIFICATE

Certified that the work contained in this report entitled
“Design and Implementation of a User-Friendly Platform (UFPI)
Interface for Controlling the KUKA LBR iiwa Robot
Via Real-Time Hand Guiding”

Was carried out by
Omar Rateb Shehata

Under our supervision and that in our opinion, it is fully adequate
in scope and quality, for the requirements of the master thesis in
Mechatronics Engineering Department.

Supervisors

“Prof. Dr. Rolf Biesenbach”

“Prof. Dr. Mohammad Baniyounis”

Supervisor, Head of Laboratory

“Dipl.-Ing. Jan Falkenhain”

Mechatronics Engineering Department
Faculty of Engineering and Technology
JIM2L Double-Degree Master Program

Bochum University of Applied Sciences

Bochum, Germany

Philadelphia University

Amman, Jordan

STATUTORY DECLARATION

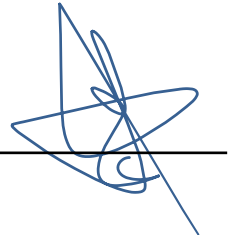
I affirm that I have personally undertaken this project and thesis, and I have not exploited anyone's work. Moreover, I can assure you that I have not received any external assistance or resorted to any prohibited methods, relying solely on the aids specified within the Master Thesis.

All information contained in this thesis has been written by me, and all of the literature and other publications and books have been referenced and identified.

I also declare that any individuals or institutions assisted in this project have been mentioned, either literally or in content.

Bochum, September 2023

Omar Shehata



DEDICATION

*To my beloved mother and family, who always supports me, and
believing in me all the time.*

*To my beloved uncle Murad Abu Markhieya, who is the reason of
what I become today.*

I dedicate this work to you.

Thanks for your endless encouragement and support.

ACKNOWLEDGEMENT

I would like to send my gratitude to everyone who has contributed to this thesis and to all who helped me in my master's journey.

Primarily, I would like to give many thanks to my thesis supervisor Professor Rolf Biesenbach, who helped me during my work on this thesis. I am also grateful for my second supervisor support, Professor Mohammed Bani Younis, in guiding me towards this project which has polished on my academic knowledge. I am honestly grateful for their support and praise for my work along the course of this thesis.

I want also to take the chance and thank Mr. Jan Falkenhain for his efforts and support throughout my stay in Bochum. Without his encouragement and support none of this work would have been achieved.

I would like to thank Mrs. Katrin Heymann, Miss. Andrea Claes, and student Lena Scholz from the International Office for resolving my organizational issues and for assisting me during my stay in Bochum.

I am also so thankful for the Mechatronics Engineering Family at the Faculty of Engineering and Technology – Philadelphia University for their patience and support during the past years.

Finally, this work could never have seen the light without the financial support from the European Tempus-JIM21 project and Bochum university of applied science.

TABLE OF CONTENTS

<i>CERTIFICATE</i>	ii
<i>STATUTORY DECLARATION</i>	iii
<i>DEDICATION</i>	iv
<i>ACKNOWLEDGEMENT</i>	v
<i>TABLE OF CONTENTS</i>	vi
<i>LIST OF FIGURES</i>	ix
<i>LIST OF TABLES</i>	xii
<i>GLOSSARY</i>	xiii
<i>ABSTRACT</i>	xv
<i>CHAPTER 1: INTRODUCTION</i>	1
1.1 Background:	1
1.2 Previous Work:	3
1.3 Problem Definition:	4
1.4 Aim:	4
1.5 Objectives:	5
1.6 Scope:	5
<i>CHAPTER 2: THEORETICAL BACKGROUND AND LITERATURE REVIEW</i>	6
2.1 Introduction:	6
2.2 Theoretical Background:	6
2.2.1 KUKA LBR iiwa 7 R800 Cobot:	6
2.2.2 KUKA LBR iiwa 7 R800 Specifications:	7
2.2.3 Collaborative Robot:	9
2.2.4 ROS-integrated for The KUKA LBR iiwa:	10
2.2.5 Sunrise Workbench:	10

2.2.6	<i>KUKA Sunrise Toolbox:</i>	12
2.2.7	<i>V-REP Software:</i>	13
2.2.8	<i>RoboDK Simulation:</i>	14
2.3	<i>Literature Review:</i>	16
2.4	<i>Conclusion:</i>	21
CHAPTER 3: ROBOT SYSTEM DESIGN AND SETUP		22
3.1	<i>Introduction:</i>	22
3.2	<i>System Requirements:</i>	22
3.3	<i>System Block Diagram:</i>	23
3.4	<i>System Kinematics and Mathematical Model:</i>	24
3.4.1	<i>D-H Parameters of The KUKA LBR iiwa 7 R800 Robot:</i>	24
3.4.2	<i>Forward Kinematic for The KUKA LBR iiwa 7 R800 Robot:</i>	26
3.4.3	<i>Inverse Kinematic for The KUKA LBR iiwa 7 R800 Robot:</i>	29
3.4.4	<i>Jacobian Matrix for The End-Effector Velocities:</i>	31
3.4.5	<i>Dynamic Model Due to Contact with Obstacles:</i>	31
3.5	<i>System Interface:</i>	33
3.5.1	<i>KUKA smartPAD Selection:</i>	33
3.5.2	<i>Controller Selection:</i>	37
3.5.3	<i>Gripper Selection:</i>	39
CHAPTER 4: IMPLEMENTATION OF UFPI AND DEVELOPMENT		41
4.1	<i>Introduction:</i>	41
4.2	<i>Control Process and System Flowchart:</i>	41
4.2.1	<i>Control Process and Safety Aspects:</i>	42
4.2.2	<i>System Flowchart:</i>	45
4.3	<i>Operation Manual:</i>	47

4.4	<i>Problems and Solutions:</i>	53
<i>CHAPTER 5: RESEARCH PLAN AND CONCLUSIONS</i>		54
5.1	<i>Research Methodology:</i>	54
5.2	<i>System Description:</i>	55
5.3	<i>Contribution and Significance:</i>	56
5.4	<i>Conclusions and Future Work:</i>	56
<i>REFERENCES:</i>		58
<i>APPENDIX A</i>		64
<i>APPENDIX B</i>		64
<i>APPENDIX C</i>		81
<i>APPENDIX D</i>		83
<i>ZUSAMMENFASSUNG</i>		84

LIST OF FIGURES

<i>Figure 1.1.1 The model of KUKA LBR iiwa 7 R800</i>	<i>2</i>
<i>Figure 2.2.1 The Axis-specific coordinate for the reference frame of KUKA LBR iiwa 7 R800.....</i>	<i>7</i>
<i>Figure 2.2.2 Workspace graphic [10]</i>	<i>8</i>
<i>Figure 2.2.3 The setup includes a measuring device provided by the Institute for Occupational Safety and Health of the German Social Accident Insurance Institute.....</i>	<i>9</i>
<i>Figure 2.2.4 Visualization of the KUKA ROS Interface Architecture</i>	<i>10</i>
<i>Figure 2.2.5 User interface for Sunrise Workbench.....</i>	<i>11</i>
<i>Figure 2.2.6 The architecture and communication scheme of the KST [20].....</i>	<i>13</i>
<i>Figure 2.2.7 The combination between V-Rep, MATLAB, and the KUKA Sunrise Toolbox [23].....</i>	<i>14</i>
<i>Figure 2.2.8 Added Python file to the RoboDK simulation to be synchronized with the robot controller using the Sunrise Workbench [27].....</i>	<i>15</i>
<i>Figure 3.3.1 Project block diagram.....</i>	<i>23</i>
<i>Figure 3.4.1 The coordinate reference frame for each joint parameter of the manipulator [56].....</i>	<i>24</i>
<i>Figure 3.4.2 media flange electrical dimensions.....</i>	<i>25</i>
<i>Figure 3.5.1 Front view of smartPAD</i>	<i>33</i>
<i>Figure 3.5.2 Rear view of smartPAD</i>	<i>34</i>
<i>Figure 3.5.3 KUKA smartHMI user interface</i>	<i>35</i>
<i>Figure 3.5.4 A visual model of the seven sliders in KUKA smartPAD from [59].....</i>	<i>37</i>
<i>Figure 3.5.5 KUKA Sunrise Cabinet robot controller overview</i>	<i>37</i>

<i>Figure 3.5.6 KUKA Sunrise Cabinet interfaces</i>	<i>38</i>
<i>Figure 3.5.7 Medien-Flansch Touch Elektrisch Gripper</i>	<i>40</i>
<i>Figure 4.2.1 Collision detection in safety configuration</i>	<i>42</i>
<i>Figure 4.2.2 Collision detection by coding</i>	<i>42</i>
<i>Figure 4.2.3 Hand Guiding Motion feature</i>	<i>43</i>
<i>Figure 4.2.4 Impedance Mode feature</i>	<i>43</i>
<i>Figure 4.2.5 While loop to start recording the user-desired movements</i>	<i>44</i>
<i>Figure 4.2.6 While loop to execute the recorded movements</i>	<i>44</i>
<i>Figure 4.2.7 User key programming</i>	<i>45</i>
<i>Figure 4.2.8 System Flowchart</i>	<i>46</i>
<i>Figure 4.3.1 Informational comment indicating to enable automatic mode</i>	<i>47</i>
<i>Figure 4.3.2 Question command indicating the type of motion</i>	<i>47</i>
<i>Figure 4.3.3 Question command asking about the approach to reach the destination point</i>	<i>48</i>
<i>Figure 4.3.4 Informational comment explaining how to use the Jog Keys</i>	<i>48</i>
<i>Figure 4.3.5 Question command to determine the specific axis</i>	<i>48</i>
<i>Figure 4.3.6 Question command to know the amount of offset</i>	<i>49</i>
<i>Figure 4.3.7 Right-Hand rule to know the direction of the coordinates for the EEf</i>	<i>49</i>
<i>Figure 4.3.8 Question command to indicate the sign of the offset value</i>	<i>50</i>
<i>Figure 4.3.9 Question command to ascertain the number of the previously designated reference point</i>	<i>50</i>
<i>Figure 4.3.10 Question command to save the new position</i>	<i>51</i>

<i>Figure 4.3.11 Question command to change the state of the Gripper.....</i>	<i>51</i>
<i>Figure 4.3.12 Question command to save the recorded point as a reference point.....</i>	<i>52</i>
<i>Figure 4.3.13 Question command to determine the number of re-executions.....</i>	<i>52</i>
<i>Figure 4.3.14 Question command to re-program the recorded movements or terminate the program.....</i>	<i>52</i>
<i>Figure B.1 Create new project</i>	<i>76</i>
<i>Figure B.2 Assign IP-address of the robot</i>	<i>76</i>
<i>Figure B.3 Select LBR iiwa 7 R800</i>	<i>77</i>
<i>Figure B.4 Select Medien-Flansch Touch elektrisch.....</i>	<i>77</i>
<i>Figure B.5 Safety configuration</i>	<i>78</i>
<i>Figure B.6 Customer PSM safety parameter.....</i>	<i>78</i>
<i>Figure B.7 KUKA PSM safety parameter.....</i>	<i>79</i>
<i>Figure B.8 StationSetup.....</i>	<i>79</i>
<i>Figure B.9 Software Tab.....</i>	<i>79</i>
<i>Figure B.10 Configuration and Parameter</i>	<i>80</i>
<i>Figure B.11 Installation.....</i>	<i>80</i>
<i>Figure D.1 KUKA LBR IIWA 7 R800 robot in the university laboratory</i>	<i>83</i>

LIST OF TABLES

<i>Table 2.2.1 Axis limitation for KUKA LBR iiwa 7 R800</i>	<i>7</i>
<i>Table 2.2.2 Specifications of the KUKA LBR iiwa 7 R800.....</i>	<i>8</i>
<i>Table 3.2.1 Project System Requirements</i>	<i>23</i>
<i>Table 3.4.1 DH parameters for KUKA LBR iiwa 14 R820 (7DOF).....</i>	<i>25</i>
<i>Table 3.5.1 Description for the front view of the smartPAD.....</i>	<i>34</i>
<i>Table 3.5.2 Description for the rear view of the smartPAD.....</i>	<i>35</i>
<i>Table 3.5.3 Description for the smartHMI user interface</i>	<i>36</i>
<i>Table 3.5.4 KUKA Sunrise Cabinet robot controller components</i>	<i>38</i>
<i>Table 3.5.5 Connection interfaces for KUKA Sunrise Cabinet robot controller</i>	<i>39</i>
<i>Table 3.5.6 Gripper Characteristics.....</i>	<i>40</i>

GLOSSARY

UFPI: User-Friendly Platform Interface

EEF: End Effector

TCP: Tool Centre Point

iiwa: Intelligent Industrial Work Assistant

IoT: Internet of Things

GUI: Graphical User Interface

CAD: Computer-Aided Design

DOF: Degrees of Freedom

API: Application Programming Interface

KUKA: Keller und Knappich Augsburg (German industrial robot manufacturer)

LBR: Leichtbauroboter in German

LWR: Lightweight Robot

ROS: Robot Operating System

V-REP: Virtual Robot Experimentation Platform

RoboDK: Robot Development Kit

LED: Light Emitting Diode

Nm: Newton-Meter

SMS: Safety-Rated Monitored Stop

HG: Hand-Guiding

SSM: Speed and Separation Monitoring

PFL: Power and Force Limiting

OOP: Object-Oriented Programming

KST: KUKA Sunrise Toolbox

ISO: The International Organization for Standardization

TS: Technical Specification

KMP: KUKA Mobile Platform

OLRP: Off-Line Robot Programming

LE: Lagrangian Euler formulation

USB: Universal Serial Bus

Cobots: Collaborative Robots

HMI: Human-Machine Interface

IDE: Integrated Development Environment

AR: Augmented Reality

UI: User Interface

HRC: Human-Robot Collaboration

PbD: Programming by Demonstration

SEAs: Series Elastic Actuators

UDP: User Datagram Protocol

D-H: Denavit–Hartenberg parameters

CIB: Cabinet Interface Board

SR: Small Robot

PMB: Power Management Board

KLI: KUKA Line Interface

ABSTRACT

This research presents a comprehensive study focused on the design and implementation of a user-friendly platform (UFPI) interface tailored for real-time control of the KUKA LBR iiwa 7 R800 robot. The aim of the study is to enhance the interaction and collaboration between humans and robots in industrial settings, particularly in manual assembly tasks. The research emphasizes in simplifying complex programming, ensuring safety, optimizing trajectory, enhancing accuracy, and facilitating seamless system integration. This study uses a comprehensive approach in qualitative data collection and analysis. Based on this data, the research develops a Java-based interface for the smartPAD control panel through using Sunrise Workbench software, enabling intuitive real-time control and monitoring of the robot's movements. The research underscores the significance of this advancement in promoting efficient and safe industrial operations, while also outlining future research directions, and highlighting its potential for wider industrial adoption.

Keywords: Industrial manipulator; Collaborative robots; KUKA LBR iiwa robot; Redundant manipulator; Precise hand-guiding; Obstacle avoidance; Intuitive interfaces

CHAPTER 1: INTRODUCTION

This chapter provides an introduction to the field of collaborative robotics, emphasizing the importance of the KUKA LBR iiwa 7 R800 robot. Where it presents relevant background information, reviews of previous work, defines the research problem and objectives, and establishes the scope of the study. It aims to create a clear context and framework for the subsequent chapters, in addition to highlighting the motivation and significance of the research.

1.1 Background:

The field of robotics in the industrial domain has witnessed significant advancements in recent years. Where the collaborative robots, also known as cobots, have emerged as a promising technology for enhancing the capabilities and performance of the robots in manufacturing environments [1]. These lightweight robots are equipped with advanced sensors and possess intelligent capabilities, making them specialized in delicate assembly work and suitable for safe and efficient human-robot interaction.

The LBR iiwa, introduced in 2013 [2], is the first series of sensitive collaborative robots that was produced in the world. Developed by KUKA, a renowned industrial robotics company [3]. LBR stands for "Leichtbauroboter" in German, which translates to lightweight robot in English [3]. With the LBR iiwa, humans and robots can collaboratively engage in highly sensitive tasks, enabling new applications and fostering improved cost-effectiveness and efficiency [4].

The collaborative and sensitive LBR iiwa robot is available in two versions with payload capacities of 7 and 14 kilograms. This research will specifically focus on the KUKA LBR iiwa 7 R800, depicted in Figure 1.1.1, which has gained recognition for its versatility and adaptability in various industrial and research applications, and designed to support the principles of Industry 4.0, also known as the fourth industrial revolution [5], and aims to the integration of digital technologies and automation in industrial processes.



Figure 1.1.1 The model of KUKA LBR iiwa 7 R800

Key features of the KUKA LBR iiwa 7 R800 that align with the goals of Industry 4.0 include [6]:

- **Collaboration and Safety:** The robot is equipped with sensitive sensors and control systems that enable it to detect and respond to human presence and interact safely with human workers. This collaborative capability allows for more flexible and dynamic work processes.
- **Flexibility:** The LBR iiwa 7 R800 has a lightweight design and seven degrees of freedom, enabling it to perform a wide range of tasks with precision and dexterity. It can handle delicate and complex operations, making it suitable for various manufacturing applications.
- **Connectivity:** The robot is designed to be easily integrated into networked production systems. It can communicate with other machines, sensors, and control systems, allowing for seamless data exchange and coordination in the production environment.
- **Advanced Control and Programming:** The LBR iiwa 7 R800 uses advanced control algorithms and software programming to enable sophisticated motion control and path planning. It can adapt to changing conditions, handle variations in tasks, and optimize its movements for efficiency.
- **Data Analysis and Optimization:** By leveraging its connectivity and advanced software capabilities, the LBR iiwa 7 R800 can collect and analyze data from its operations. This data can be used for process optimization, predictive maintenance, and continuous improvement of production systems.

By combining the capabilities of the KUKA LBR iiwa 7 R800 with the principles of Industry 4.0, manufacturers can achieve higher productivity, improved efficiency, greater flexibility, and enhanced worker safety in their operations [6]. The robot can assist human workers in performing tasks that require precision, strength, or repetitive motions, while also facilitating the integration of digital technologies to optimize overall manufacturing processes [7].

1.2 Previous Work:

Extensive research has been conducted on different aspects of collaborative robotics, with a specific focus on the KUKA LBR iiwa robot. Previous studies have addressed critical challenges, explored novel approaches, and proposed innovative solutions to enhance the integration and performance of these robots in manufacturing processes. Literature in this field includes investigations into evaluation criteria for lightweight robots, workspace monitoring, robot programming, path planning, impedance control, human-robot interaction, control interfaces, and calibration methods.

The International Organization for Standardization / Technical Specification (ISO/TS) combination, which encompasses safety-rated monitored stop (SMS), hand-guiding operation (HG), speed and separation monitoring (SSM), and power and force limiting (PFL), has been defined as four techniques for collaborative operation in collaborative applications. These techniques play a crucial role in ensuring the safety and efficiency of human-robot collaboration.

The integration of lightweight robots as assistance systems in assembly tasks has been a topic of discussion in previous works. By incorporating lightweight robots into manual assembly lines, this integration aims to improve efficiency and quality in assembly processes.

1.3 Problem Definition:

Despite the progress made in collaborative robotics, there are still several challenges that need to be addressed, particularly in the context of manual assembly lines. These challenges include:

- The complexity of robot programming, particularly when utilizing the Java language.
- Ensuring safety in human-robot collaboration.
- Optimizing robot trajectories.
- Improving accuracy and calibration.
- Integrating with other systems or fulfilling specific application requirements.
- Developing intuitive control interfaces.

Therefore, the challenges at hand to be addressed are to simplify the programming process for the KUKA LBR iiwa robot, enhance flexibility for users, enable real-time control through a user-friendly interface, address safety requirements, calibrate the robot, and simplify the process of establishing communication between the interface and the robot.

1.4 Aim:

The aim of this research is to develop an interface using the Java language on the smartPAD via the Sunrise Workbench software. The interface will enable real-time control of the KUKA LBR iiwa robot, through the hand guiding feature to improve movement precision in narrow spaces and crowded environments, as well as to simplify the programming process, making it accessible to individuals without prior programming knowledge. Furthermore, safety requirements such as automatic stop on collisions, impedance control, and Tool Centre Point (TCP) calibration methods will be taken into consideration.

1.5 Objectives:

The objectives of this research are as follows:

- Develop a user-friendly interface for real-time control of the KUKA LBR iiwa robot.
- Simplify the programming process through smartPAD by using hand guiding feature.
- Ensure safety by implementing automatic stop on collisions, impedance control, and TCP calibration methods.
- Improve the accuracy and calibration of the robot through advanced techniques in the hand guiding motion.
- Explore the potential for future advancements in the integration of collaborative robots into assembly processes.

1.6 Scope:

This research has a general focus on KUKA LBR iiwa robots, with specific emphasis on the Type 7 R800 and its integration into assembly tasks. The research aims to address the challenges associated with the programming process, safety requirements, and calibration methods for the robot. The development of a user-friendly interface using the Java language on the smartPAD, via the Sunrise Workbench software, will be the primary focus. The research will also provide insights and recommendations for future advancements in the field of collaborative robotics, particularly in the context of assembly processes.

CHAPTER 2: THEORETICAL BACKGROUND AND LITERATURE REVIEW

This chapter provides an introduction to the KUKA LBR iiwa R800 robot, accompanied by its theoretical background and a comprehensive literature review. It culminates with a comprehensive conclusion summarizing the key findings.

2.1 Introduction:

The following research presents a diverse collection of studies focusing on various aspects of robotics in the industrial domain. These studies aim to address critical challenges, explore novel approaches, and propose innovative solutions for enhancing the capabilities and performance of collaborative robots in manufacturing environments. Specifically, the focus is on lightweight robots that are equipped with advanced sensors and possess intelligent capabilities, most notably the KUKA LBR iiwa 7 R800, a collaborative robot designed for safe and efficient human-robot interaction. It has seven degrees of freedom, integrated force/torque sensors, and advanced control capabilities making it a versatile tool for a wide range of industrial and research applications.

2.2 Theoretical Background:

This section aims to provide a comprehensive overview of the framework, concepts, models, and specific theories that form the foundation of this project, as well as recognizing the historical context that bolsters the development of those theories.

2.2.1 KUKA LBR iiwa 7 R800 Cobot:

The KUKA LBR iiwa 7 R800 is a specific model of the KUKA LBR iiwa series of robots. The LBR iiwa (intelligent industrial work assistant) is a collaborative robot designed to work alongside humans in various industrial and research applications [8]. The "7 R800" designation refers to the robot's key specifications, including a 7 kg payload and an 800-millimeter reach as shown in Figure 2.2.1.

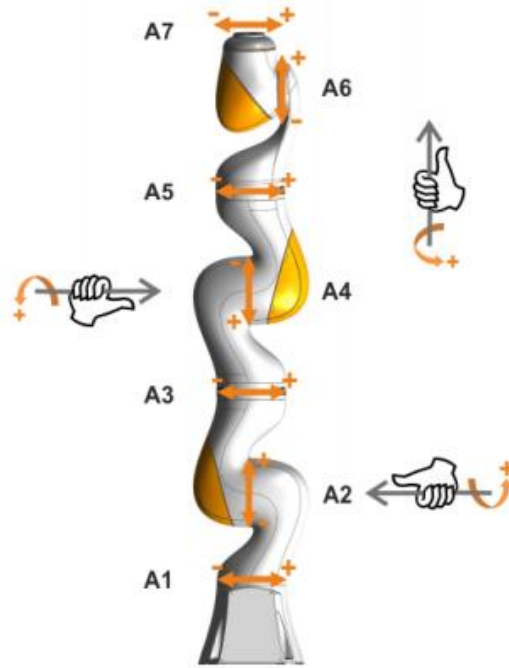


Figure 2.2.1 The Axis-specific coordinate for the reference frame of KUKA LBR iiwa 7 R800

2.2.2 KUKA LBR iiwa 7 R800 Specifications:

The term "degrees of freedom" refers to the number of independent directions in which a robot joint can move. In the case of the KUKA LBR iiwa 7 R800, it possesses seven degrees of freedom, indicating that it has seven independently controllable joints. These joints enable the robot to move in a wide range of directions, allowing for increased flexibility and manoeuvrability [9]. The repeatability of the KUKA LBR iiwa 7 R800 robot is ± 0.1 mm. Table 2.2.1 shows the axis limitations of LBR iiwa 7 R800 industrial robot, and Table 2.2.2 shows the specifications of it:

Table 2.2.1 Axis limitation for KUKA LBR iiwa 7 R800 [10]

Axis data	Motion range	Axis velocity rated with payload
Axis 1 (A1)	$\pm 170^\circ$	98 °/s
Axis 2 (A2)	$\pm 120^\circ$	98 °/s
Axis 3 (A3)	$\pm 170^\circ$	100 °/s
Axis 4 (A4)	$\pm 120^\circ$	130 °/s
Axis 5 (A5)	$\pm 170^\circ$	140 °/s
Axis 6 (A6)	$\pm 120^\circ$	180 °/s
Axis 7 (A7)	$\pm 175^\circ$	180 °/s

Table 2.2.2 Specifications of the KUKA LBR iiwa 7 R800 [10]

Basic Data	
Kinematic type	Articulated robot
Brand	KUKA
Model	LBR iiwa 7 R800
Cobot “Collaborative Robot”	Yes
Safety requirements	Category 3 and Performance Level d in accordance with EN ISO 13849-1
Operating conditions	5 °C to 45 °C (278 K to 318 K)
Protection rating (IEC 60529)	IP54
Controller	KUKA Sunrise Cabinet
Technical Data	
Maximum reach	800 mm
Rated payload	7 kg
Pose repeatability (ISO 9283)	± 0.1 mm
Number of axes	7
Weight	approx. 23.9 kg

In Figure 2.2.2 The Workspace domain for KUKA LBR iiwa 7 R800 robot is illustrated based on the Axis data shown in Table 2.2.2:

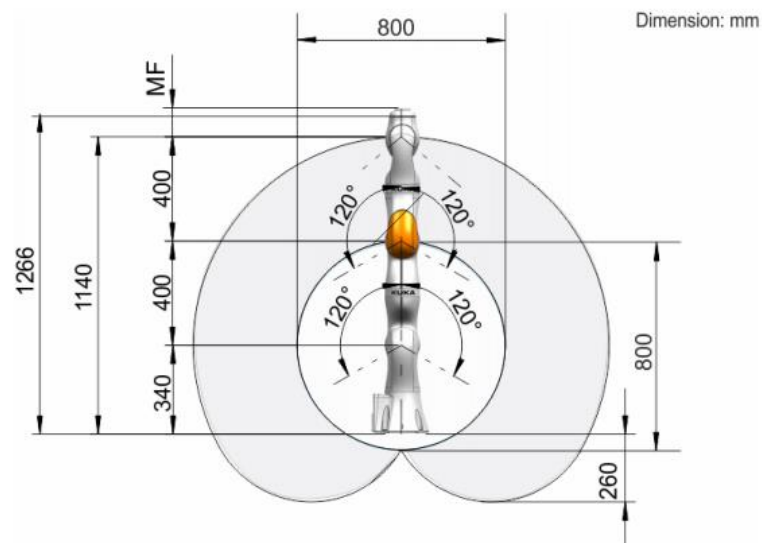


Figure 2.2.2 Workspace graphic [10]

2.2.3 Collaborative Robot:

The KUKA LBR iiwa 7 R800 is driven by electric motors located at each joint, providing precise and efficient actuation. Unlike hydraulic actuators, electric motors offer advantages such as reduced noise, improved energy efficiency, and increased control accuracy [11]. This makes the KUKA LBR iiwa 7 R800 well-suited for applications that require precise and controlled movements.

One of the key features of the KUKA LBR iiwa 7 R800 is its collaborative nature. The robot is designed to work in close proximity to humans without compromising safety. It is equipped with integrated force/torque sensors in each joint, enabling it to detect external forces and react accordingly [12]. These sensors provide feedback to the robot's control system, allowing it to adapt its movements based on the detected forces and ensure safe interaction with its human counterparts. Figure 2.2.3 shows an example of its use in the field of Occupational Safety and Health.

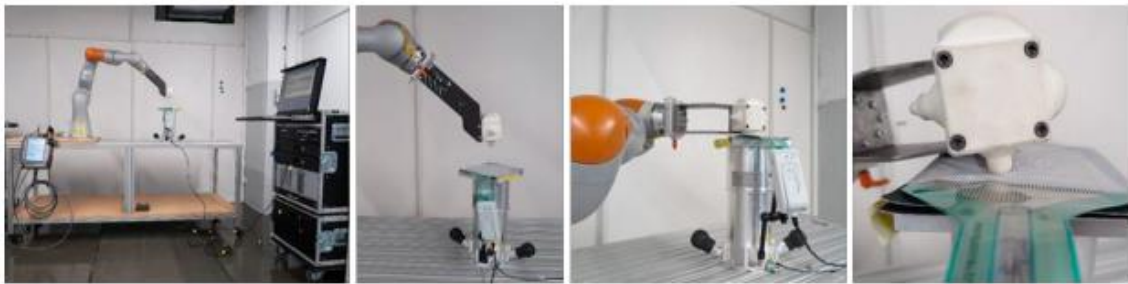


Figure 2.2.3 The setup includes a measuring device provided by the Institute for Occupational Safety and Health of the German Social Accident Insurance Institute [12]

The KUKA LBR iiwa 7 R800 finds applications in a variety of industries, including manufacturing, assembly, logistics, and research [13]. Its collaborative capabilities make it suitable for tasks that involve human-robot interaction, such as assembly line assistance, pick-and-place operations, and intricate manipulation tasks. The robot's advanced control, perception, and safety features contribute to an increase in productivity, an improved efficiency, and safer work environment.

2.2.4 ROS-integrated for The KUKA LBR iiwa:

The KUKA LBR iiwa 7 R800 is powered by advanced control and perception capabilities. It utilizes the Robot Operating System (ROS), an open-source software framework widely used in robotics. ROS provides a flexible and modular platform for control, communication, and integration of various robot components as shown in Figure 2.2.4. This allows developers to easily program and customize the robot's behaviour to suit specific applications [14].

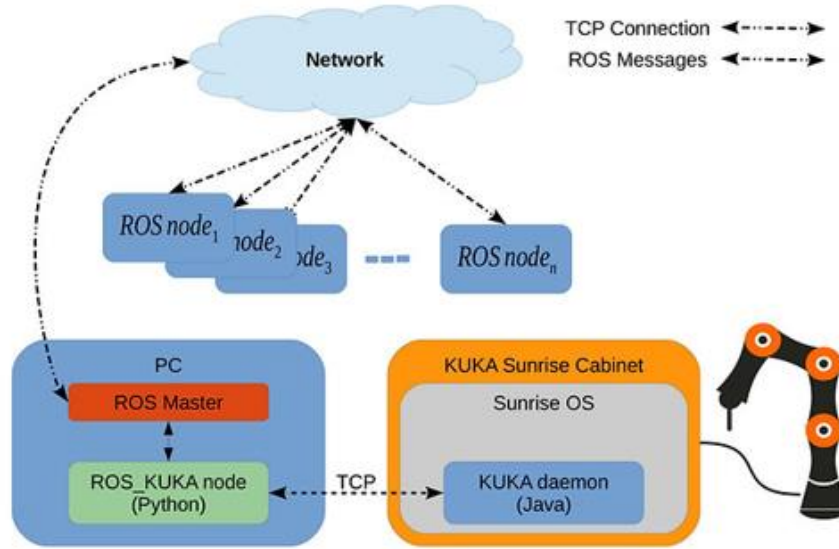


Figure 2.2.4 Visualization of the KUKA ROS Interface Architecture

Additionally, the KUKA LBR iiwa 7 R800 benefits from the integrated motion planning framework within ROS, which enables efficient trajectory planning and collisions avoidance, to ensuring smooth and safe robot movements [14]. This capability is crucial in collaborative environments where the robot needs to navigate around obstacles or work in close proximity to humans.

2.2.5 Sunrise Workbench:

The KUKA LBR iiwa 7 R800 robot utilizes the KUKA Sunrise Workbench, a Java-based integrated development environment (IDE) based on Eclipse IDE, for its programming needs. WorkVisual software handles bus configurations and mapping, while the autonomous navigation, control, management, and monitoring of mobile platforms rely on the additional KUKA Navigation Solution package [15]. Furthermore, for programming the KUKA mobile platform (KMP), the KUKA Sunrise Mobility software is necessary [16].

The KUKA Sunrise Workbench is a powerful software platform developed by KUKA Robotics, a renowned global supplier of industrial robots and automation solutions. The Sunrise Workbench serves as a comprehensive programming and control environment for KUKA robots, enabling users to easily and efficiently develop, program, and control robotic applications [17]. With its open-source software, user-friendly interface, and extensive functionality as shown in the Figure 2.2.5; the Sunrise Workbench has gained popularity among manufacturers and automation engineers worldwide.

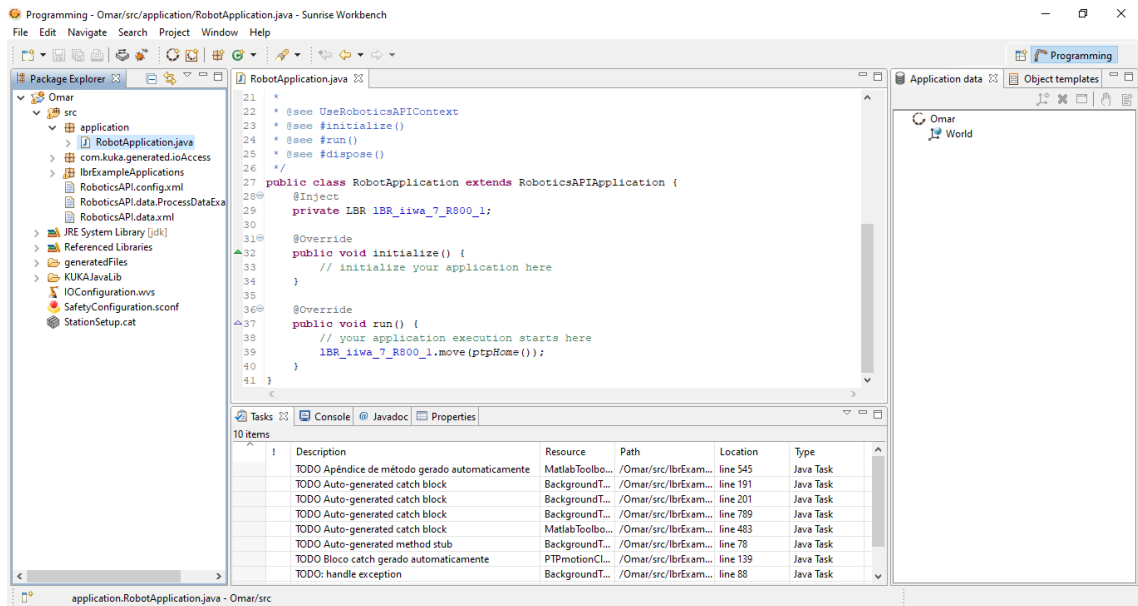


Figure 2.2.5 User interface for Sunrise Workbench

The KUKA Sunrise Workbench emerged as a result of KUKA Robotics' commitment to continuous innovation in the field of industrial robotics. As the demand for robotic automation increased across various industries, KUKA recognized the need for a flexible and intuitive software platform that would empower users to harness the full potential of their robotic systems [18].

The development of the Sunrise Workbench began with a focus on simplifying the programming process for KUKA robots. Traditionally, robot programming involved complex code development and extensive manual adjustments, posing challenges for both experienced programmers and those new to robotics [18]. KUKA aimed to overcome these barriers and provide a programming environment that would streamline the development process, reduce programming time, and enhance overall productivity.

The Sunrise Workbench incorporates modern programming paradigms, such as object-oriented programming (OOP), graphical user interface (GUI) design, and modular

programming concepts. This approach simplifies the programming workflow, making it more accessible to users with different levels of expertise. Additionally, the Sunrise Workbench supports Java programming language. Whereas, Java language is a direct way of programming the Collaborative Robots [19].

2.2.6 KUKA Sunrise Toolbox:

The KUKA Sunrise Toolbox (KST) is a MATLAB toolbox that is designed to interface with the KUKA Sunrise.OS controller. It is specifically developed for use with the KUKA LBR iiwa robot, which is a collaborative and sensitive robot equipped with the KUKA Sunrise.OS controller [20].

The KST provides a range of functionalities and features that facilitate the development and implementation of advanced robot applications. It includes approximately 100 functions that cover networking, real-time control, motion control, parameter setting/getting, general-purpose calculations, and physical interaction. These functions enable users to perform tasks such as networking with the robot controller, controlling the robot in real-time, defining point-to-point motions, setting and retrieving parameter values, performing kinematic calculations, and enabling physical interaction with the robot [21].

Compared to other MATLAB toolboxes in the robotics field, the KST offers several advantages. It allows for easy and fast interaction with the robot from an external computer, making it convenient for users to interface with the KUKA Sunrise.OS controller. It integrates diverse functionalities, enabling users to perform a wide range of tasks related to robot control and interaction. The toolbox is designed to accelerate the development of advanced robot applications by leveraging the capabilities of MATLAB, including complex algorithms and advanced mathematical tools. Additionally, the KST makes the KUKA LBR iiwa manipulators accessible to a wider range of users with different backgrounds, including academic, educational, and industrial applications [20]. It is compatible with different operating systems, including Windows, Linux, and macOS.

The KST operates by running inside MATLAB and communicates with the KUKA Sunrise.OS controller through TCP/IP over an Ethernet network as shown in Figure 2.2.6. It supports both the KUKA iiwa 7 R800 and KUKA iiwa 14 R820 manipulators.

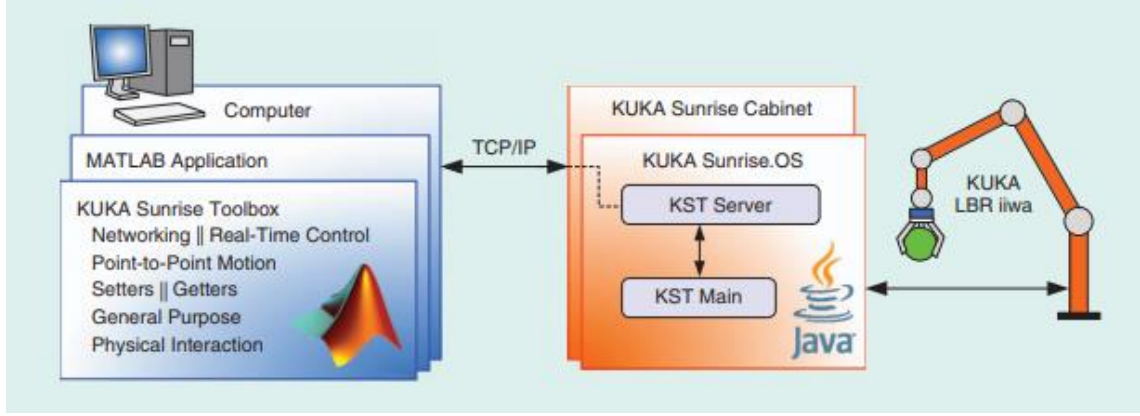


Figure 2.2.6 The architecture and communication scheme of the KST [20]

2.2.7 V-REP Software:

V-REP (Virtual Robot Experimentation Platform), now known as CoppeliaSim, is a versatile and powerful robotic simulation software. It is designed to provide a comprehensive environment for developing, testing, and simulating robotic systems. V-REP offers a wide range of features and capabilities that make it a popular choice among researchers, engineers, and educators in the field of robotics. One notable feature of V-Rep is its integration with MATLAB, which enables users to control their 3D simulations directly from MATLAB [21].

In the context of controlling the KUKA iiwa manipulator, the KUKA Sunrise Toolbox (KST) serves as a middle-ware between the simulation and the real robot. The example provided in [22] showcases how KST can be utilized to control the KUKA iiwa manipulator from a 3D simulation developed using V-Rep and MATLAB.

By leveraging the power of V-Rep and the integration with MATLAB through KST, users can achieve seamless control of the KUKA iiwa manipulator within their 3D simulation environment. This allows for the development and testing of complex robotic tasks and scenarios without the need for physical interaction with the real robot [21].

Overall, V-Rep is a robust robotic simulation environment that, when combined with the KUKA Sunrise Toolbox (KST) and MATLAB as shown in Figure 2.2.7, provides users with a comprehensive platform for simulating, controlling, and analyzing the behavior of the KUKA iiwa manipulator within a 3D simulation. This integration enables researchers and developers to optimize their robotic applications and algorithms before implementing them on the physical robot.

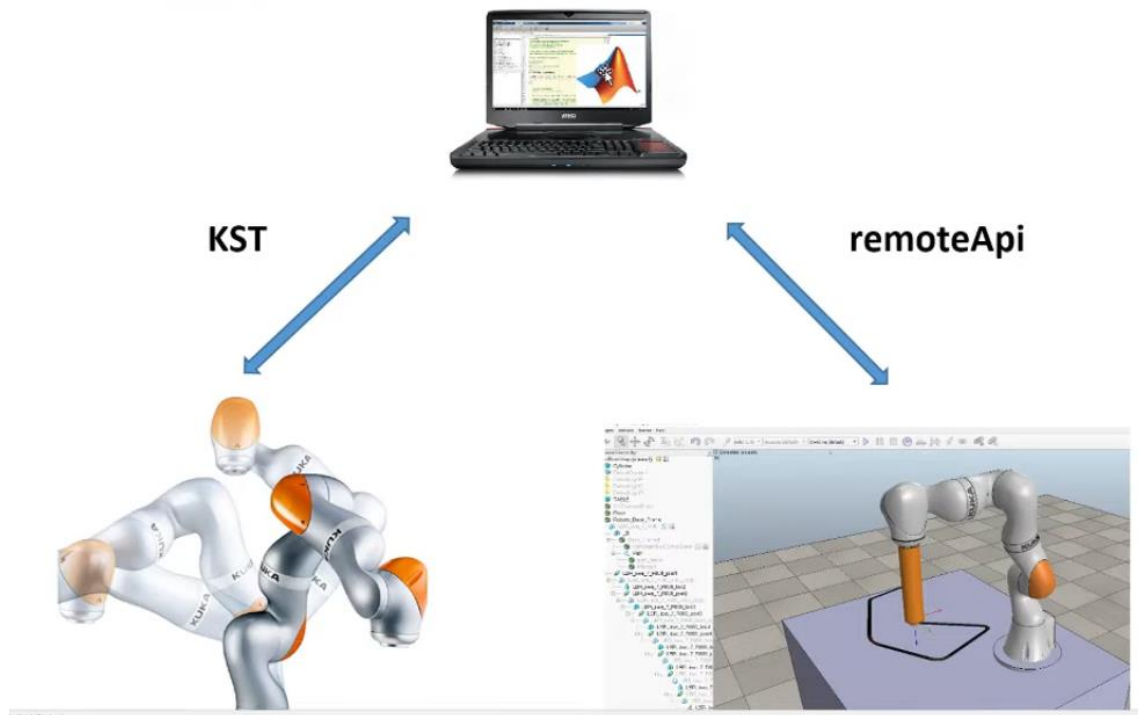


Figure 2.2.7 The combination between V-Rep, MATLAB, and the KUKA Sunrise Toolbox [23]

2.2.8 RoboDK Simulation:

RoboDK is a versatile and powerful software platform designed for robotic simulation, programming, and offline robot programming (OLRP). The name "RoboDK" stands for "Robot Development Kit" [24]. It provides a comprehensive environment for integrating, simulating, and programming industrial robots from various manufacturers. RoboDK offers a user-friendly interface that allows users to visually design, simulate, and generate robot programs for a wide range of applications.

The primary purpose of RoboDK is to facilitate the development and deployment of robotic systems by reducing the time and effort required for programming and setup. It allows users to create robot programs offline, away from the production line, which improves efficiency and reduces downtime. RoboDK supports a vast library of robot models, tools, and peripherals, making it compatible with a wide range of industrial robots and robotic applications [25].

The software enables users to import their own CAD models, define the robot's kinematics and behavior, simulate the robot's movements within a virtual environment, and synchronized it with the robot controller using the Sunrise Workbench for example as shown in Figure 2.2.8. The powerful simulation capabilities of RoboDK allow for collision detection, reachability analysis, cycle time estimation, and optimization of robot

trajectories [26]. Users can validate and refine their robot programs in the virtual world before implementing them on the actual robot, reducing the risk of errors and ensuring smoother operation.

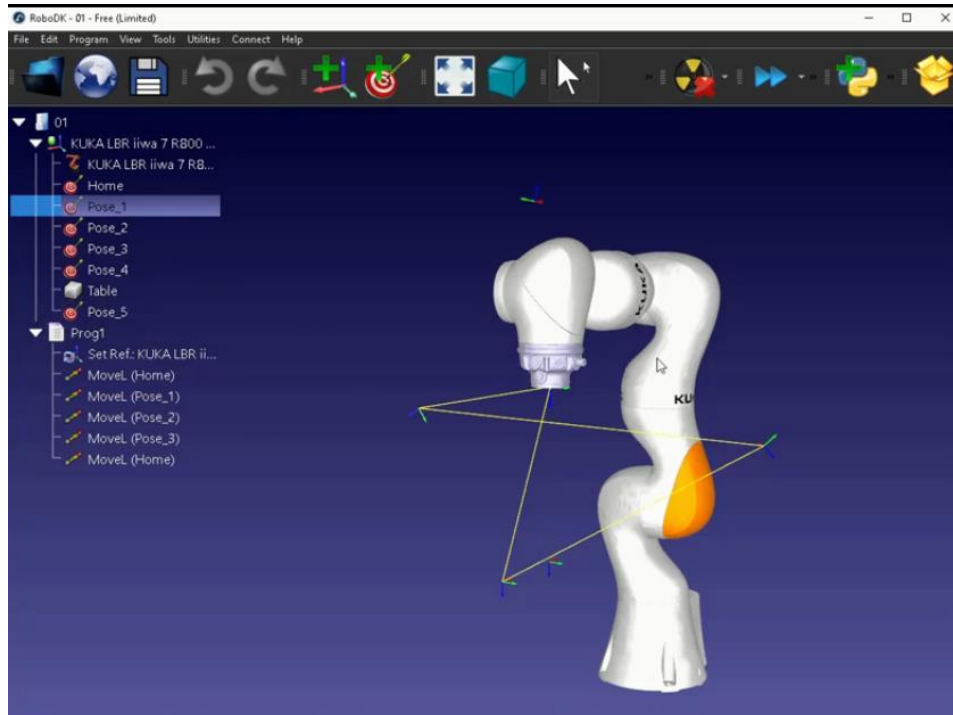


Figure 2.2.8 Added Python file to the RoboDK simulation to be synchronized with the robot controller using the Sunrise Workbench [27]

RoboDK also offers a variety of programming options to suit different user needs. Users can generate robot programs using intuitive drag-and-drop programming or use scripting in Python, C#, C++ [28]. The generated programs can be easily transferred to the robot controller or executed within the simulation environment.

2.3 Literature Review:

The objectives of several papers will be studied, considering their advantages and disadvantages will be addressed. Consequently, Design criteria for this research have been developed to address the weaknesses of each proposal and avoid its drawbacks.

In the work presented in [29], aim to propose an approach for extending the evaluation criteria specifically for scalable and modular lightweight robots, and investigate the suitability of current industrial lightweight robots for future versatile and flexible production systems. The restrictions of this article are limited empirical evidence and limited discussion on challenges and constraints associated with implementing the extended evaluation criteria, such as practical feasibility, cost implications, or compatibility with existing systems.

In these articles [30]–[32], a model for workspace monitoring and an interactive Augmented Reality (AR) User Interface (UI) have been proposed to enhance the safety and efficiency of human-robot collaboration (HRC) in industrial manufacturing settings. The paper specifically focuses on the third scenario of (ISO/TS), which involves monitoring and communicating the minimum protective distance between a robot and an operator in real-time. The limitations of these papers are recognition that the wearable AR, specifically the HoloLens setup, was not considered suitable for industrial manufacturing due to technical limitations and user feedback, and also acknowledges the need for future work and improvements, including experiments in a multi-machine work environment and enhancements based on user feedback.

Reference [33] focuses on the use of a multi-kinematic gripping system to handle different-sized cut-pieces of auxiliary materials, with the goal of reducing the cost of automation by using a single system for a variety of tasks. The system utilizes several robots with 6 or 7 degrees of freedom, and a genetic algorithm is employed for optimization. The restrictions of this article include the suboptimal absolute accuracy of the end-effectors due to the materials used and limited stiffness, which may necessitate manual adjustments. Additionally, velocity limitations exist in the transfer motions, potentially impacting the overall time needed for vacuum bagging.

In this article [34], a robot-based imaging system that combines photometric stereo and image stitching has been developed for high-resolution 3D surface imaging at a larger scale, and aims to evaluate the feasibility of high-resolution 3D surface imaging using

photometric stereo. However, the paper acknowledges limitations such as the limited encoder precision and accuracy of the robot arm, which requires compensation techniques for image stitching. In these papers [35], [36], a programming by demonstration (PbD) approach for robot programming using a haptic input device within an augmented reality environment has been explored, and also focuses on representing the characteristics and functionality of a robot's end effector through the haptic device to enable PbD. The limitations of these papers include the accuracy constraints imposed by the motion capture system, which restrict the system's suitability for high-precision applications. The lack of detailed validation metrics and extensive experimental evaluations also undermines the robustness of the proposed approach.

The purpose of this paper [37], is to address the challenges faced in programming and integrating collaborative robot systems, specifically focusing on the KUKA LBR iiwa robot, in industrial settings. However, the paper acknowledges certain limitations, such as the need for manual configuration and troubleshooting of programs, which can be time-consuming. In this research [38], presents an innovative concept for collaborative robot programming using digital twin and extended reality technologies. This paper acknowledges the limitations of current network technologies in meeting the requirements of the proposed approach, indicating the need for future advancements in 6G networks. However, it doesn't provide concrete solutions or strategies to overcome these limitations.

This research [39], tackles the issue of time delay in impedance admittance control of series elastic actuators (SEAs), which is crucial for safe human-robot collaboration in industrial settings. The restrictions of this research are Lack of validation for real-world testing to verify the applicability of the proposed method. In [40], a real-time framework that utilizes machine learning techniques has been presented to solve the analytical inverse kinematics problem for redundant manipulators, and find the optimal solution among the infinite solutions of the inverse kinematics problem. The restrictions of this article are that it focuses on solving the inverse kinematics problem specifically for kinematically redundant robots with analytical solutions, which restricts its applicability to other types of robots or kinematic configurations, and it lacks experimental validation using physical robots.

The key goal of this paper [41], is to present and compare two interfaces, SimulinkIIWA and KUKA Sunrise Toolbox (KST), for controlling KUKA iiwa robots from Simulink,

to demonstrate the ease of use and flexibility of the SimulinkIIWA interface, which allows graphical real-time control of iiwa robots without the need for writing any code. The limitations of this paper include the restriction of the SimulinkIIWA interface to Windows and Mac operating systems, the lack of a comprehensive discussion on the trade-offs and potential limitations of using User Datagram Protocol (UDP) as the communication protocol.

In this research [42], a path planning framework has been proposed for industrial robots that can handle complex manufacturing paths with process tolerances, process windows, constraints, and redundant degrees of freedom (DOF). The limitations of this research include the lack of detailed practical implementation information, such as specific experimental setups or implementation challenges. The purpose of this paper [43], is to presents an inclusive approach to enable lightweight robots (LWR) to change tasks within hybrid assembly systems quickly and with minimal setup times, and generating robot paths automatically based on digital product data. The constraints of this paper are that it does not provide detailed results of implementing the concept, making it challenging to assess the performance, efficiency, and effectiveness of this approach.

In this paper [44], a generic and comprehensive routine for optimizing robot poses and trajectories has been presented, considering factors such as kinematic sensitivity, the number of actuated joints, and the number of direction changes. The routine is experimentally validated using a KUKA LWR iiwa 14 for a pick and place process. But this research is limited to focuses specifically on vertically articulated robot arms and their pick and place processes. The routine's applicability to other types of robots or assembly scenarios is not extensively discussed.

The purpose of this research [45], is to present a path planning algorithm for hyper-redundant manipulators operating in confined spaces cluttered with obstacles. The algorithm aims to maximize the manoeuvring space by making the manipulator's links closely follow the walls of the environment, thus enabling it to navigate through tight spaces. But does not provide information about practical implementation considerations, such as sensor integration, real-time control, or hardware requirements.

The purpose of this paper [46], is to propose and develop a proof-of-concept assistive strategy using a collaborative robot in meat cutting tasks, by implementing an impedance control strategy that enables a KUKA LWR robot to provide assistive forces to a professional butcher while allowing the knife (tool) to move in all degrees of freedom.

One limitation of this paper is that primarily focuses on the development of the impedance control strategy and does not provide extensive evaluation or validation of its effectiveness in reducing musculoskeletal disorders. This article [47], focuses on the design and implementation of a software interface that enables voice control of a lightweight robot. It also discusses the evaluation of cloud services for robot control, and the examination of background noise's influence on process safety. The restrictions of this article are depended on challenges and limitations of speech recognition systems in real-world environments beyond background noise, and the recognition that performance criteria rapidly change, making long-term validity of system comparisons difficult to maintain.

In this research [48], a method for improving the accuracy of a KUKA iiwa LBR 7R 800 robot manipulator through kinematic calibration has been developed, and utilized the Atracsys fusionTrack 500 tracking system to record the manipulator positions, which enhances the accuracy of the calibration process. The research lacks detailed analysis or quantitative results to evaluate the effectiveness of the calibration method. This paper [49], presents a novel method that combines hand-guiding, redundancy, and obstacle avoidance at the end-effector (EEF) level for precise positioning in cluttered environments. it addresses the limitations of traditional hand-guiding solutions and using redundancy for navigating in-contact obstacles. The limitations of this paper are that not explore other aspects of human-robot interaction or address broader challenges in collaborative robotics, and lack of comparative analysis against existing methods.

The purpose of this article [50], is to introduce a control method that utilizes torque feedback from the joints of sensitive redundant manipulators to automatically adjust the self-motion manifold during contact with obstacles, by allowing users to control the robot at the end-effector level while the robot autonomously adapts to obstacles. This article does not delve into the potential challenges or limitations of implementing the proposed method in real-world industrial settings. In [51], a web-based virtual laboratory was designed and implemented using the KUKA LBR IIWA 14R820 robot, and emphasizes that the integration of the V-REP simulator, LabVIEW, and external libraries creates a comprehensive environment that allows for programming and controlling the robot. The paper acknowledges that there are differences between the simulated robot and the ideal model, leading to variations in the trajectories obtained. This indicates that the accuracy of the kinematic model may have limitations.

In the research presented in [52], a framework for identifying the physical parameters of a KUKA LBR iiwa 14 R820 has been introduced, which can be generalized to other serial manipulators, and extracting accurate dynamic parameters that are crucial for control, simulation, and interaction algorithms in robotic applications. But the bounds used in the optimization framework are not strictly accurate, which limits the possibility of accessing the true physical parameters solution. In this paper [53], the authors present a simulation setup using the Gazebo simulator, which allows them to analyse the robot's behaviour and identify weaknesses in the algorithm, software implementation, and robot hardware before performing real-world experiments. Simulation provides a cost-effective and safe way to test and refine the system. But they don't provide specific details or results of the validation process.

The purpose of this paper [54], is to design an intuitive and understandable interface for human-robot interaction, allowing users to program robot paths and visualize them. The system aims to enhance safety, improve human-robot interaction processes, increase the programming process speed. The limitation for this study mentions that the proposed system's accuracy depends on the devices used, suggesting that the system may not achieve high precision. In this thesis [55], the development of robotic applications for collaborative robots has been facilitated, specifically focusing on the KUKA iiwa robot, addresses the challenges associated with interacting with the KUKA iiwa robot, such as its complex architecture and non-user-friendly programming language, and development a hand-guided device, called the Multi-purpose Flange Adapter System, to simplify the control of the robot and explore its capabilities. This thesis is restricted to focusing only narrowly on the KUKA iiwa robot without addressing broader robotic systems or different robot models.

2.4 Conclusion:

This chapter has highlighted the main features and specifications of lightweight robots equipped with advanced sensors and intelligent capabilities. Especially KUKA LBR iiwa 7 R800, due to its collaborative nature, boasting seven degrees of freedom, integrated force/torque sensors, and advanced control capabilities. As a result, it has earned recognition as the premier choice for both industrial and research applications.

Through the literature review, multiple research papers were discovered that tackled various challenges and presented inventive solutions in the realm of collaborative robotics, particularly focusing on the KUKA LBR iiwa robot. These papers explored diverse subjects including evaluation criteria for lightweight robots, workspace monitoring, robot programming, path planning, impedance control, human-robot interaction, control interfaces, and calibration methods.

One notable contribution discussed in the literature review is that integrating lightweight robots, like the KUKA LBR iiwa, into assembly tasks can bring significant benefits in terms of improving efficiency and quality, emphasizing the potential of automation and human-robot collaboration in enhancing assembly processes. Furthermore, it emphasizes the importance of addressing the challenges associated with the integration of collaborative robots into manual assembly lines, particularly in terms of simplifying the programming process and enhancing flexibility for users.

Hence, this thesis aims to construct a platform interface using the Java language on the smartPAD via the Sunrise Workbench software. The purpose is to enable real-time control of KUKA iiwa robots, through the hand guiding feature, with the intention of simplifying the programming process and making it accessible to individuals without prior programming knowledge. The hand guiding motion feature is also used to achieve more accurate positioning within narrow spaces and crowded environments. Safety requirements, such as Automatic stop on collisions, impedance control, and Tool Centre Point (TCP) calibration methods, will be taken into consideration. Additionally, this research endeavours to provide a roadmap for future advancements in the integration of collaborative robots into assembly processes, fostering further improvements in this field.

CHAPTER 3: ROBOT SYSTEM DESIGN AND SETUP

This chapter introduces the system design for controlling the KUKA LBR iiwa 7 R800 robot. The design encompasses various aspects, including system requirements, the system block diagram, kinematics, and mathematical modelling of the robot. Additionally, it covers inverse kinematics, the dynamic model (with consideration for contact with obstacles), and discusses the system interfaces.

3.1 Introduction:

The importance of a well-designed system to effectively control the KUKA LBR iiwa 7 R800 robot cannot be overstated, as it is essential for developing and implementing advanced control strategies and applications. Such a system enables the robot to perform complex tasks with precision and safety in various real-world scenarios. The successful integration of the system interfaces and the mathematical models creates opportunities for further research and applications in robotics and automation, unlocking new possibilities for enhancing industrial processes, human-robot collaboration, and robotic assistance across diverse industries.

Additionally, the highlighted system interfaces play pivotal roles in operating the robot efficiently by providing control, communication interfaces, and power distribution. Therefore, careful selection of these interfaces is imperative to ensure seamless and reliable operation of the robot throughout its tasks.

3.2 System Requirements:

The project's system requirements, which involve designing a platform for controlling the robot, are detailed in Table 3.2.1:

Table 3.2.1 Project System Requirements

Tool Requirements	Type of Tool
Laptop or Desktop (PC)	Windows® / macOS®
KUKA smartPAD	smartPAD / smartPAD-2
Ethernet Cable	Ethernet interface KLI
KUKA LBR iiwa	7 R800 / 14 R820
SunriseWorkbench Software	OS 1.16 SI V3 / OS 1.11 SI V1
Robot Controller	KUKA Sunrise Cabinet
Gripper	Medien-Flansch Touch elektrisch
Supply Voltage	110V with 60Hz

3.3 System Block Diagram:

In this project, the platform interface is programmed with KUKA Sunrise Workbench, and a robot station is operated using the KUKA smartPAD control panel, comprising a robot controller, manipulator, and other devices [18]. Figure 3.3.1 illustrates the system block diagram of this project:



Figure 3.3.1 Project block diagram

- 1) Development computer with KUKA Sunrise.Workbench (connection via the KLI Ethernet of the robot controller).
- 2) KUKA smartPAD control panel.
- 3) KUKA Sunrise Cabinet robot controller.
- 4) KUKA LBR iiwa 7 R800 Manipulator.

3.4 System Kinematics and Mathematical Model:

This section provides details about the kinematic and dynamic modelling of the KUKA LBR iiwa 7 R800 robot and crucial mathematical insights into understanding and controlling the robot's movements and interactions with its environment.

3.4.1 D-H Parameters of The KUKA LBR iiwa 7 R800 Robot:

The dynamic formulation of any robotic system begins with determining the Denavit–Hartenberg (D-H) parameters, which helps to specify the precise position of each joint parameter in space by considering projections related to the previous joint coordinates. The KUKA LBR iiwa 7 R800 articulated manipulator involves seven D-H parameters (six joint parameters and one end effector). The first step in defining the D-H parameters for the robot is to identify its links and dimensions. The articulated manipulator, along with its reference frame, is depicted in Figure 3.4.1. To compute the coordinate transformations between them and derive the kinematics equations.

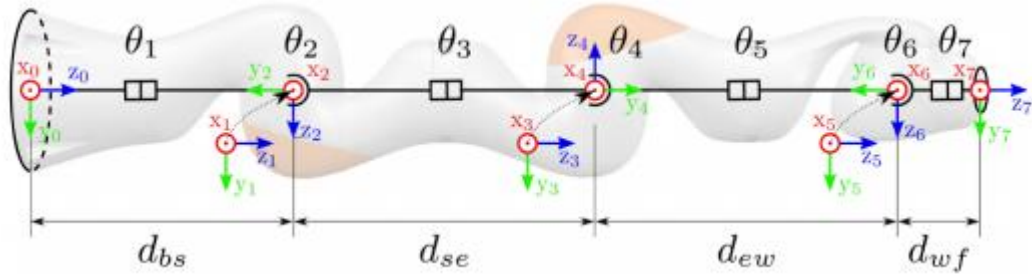


Figure 3.4.1 The coordinate reference frame for each joint parameter of the manipulator [56]

The D-H parameters for the manipulator shown in Figure 3.4.1 are listed in Table 3.4.1:

Table 3.4.1 DH parameters for KUKA LBR iiwa 14 R820 (7DOF)

Link (i)	Link Twist (α_{i-1}) [deg]	Link Length (a_{i-1}) [mm]	Link Offset (d_i) [mm]	Joint Angle (θ_i) [deg]
1	0°	0	$d_{bs} = 340$	θ_1
2	-90°	0	0	θ_2
3	90°	0	$d_{se} = 400$	θ_3
4	90°	0	0	θ_4
5	-90°	0	$d_{ew} = 400$	θ_5
6	-90°	0	0	θ_6
7	90°	0	$d_{wf} = 126 + Ef$	θ_7

- ❖ The Euclidean distance from the base coordinate frame 0 to the shoulder coordinate frame 2 is denoted as d_{bs} .
- ❖ The Euclidean distance from the base coordinate frame 2 to the shoulder coordinate frame 4 is denoted as d_{se} .
- ❖ The Euclidean distance from the base coordinate frame 4 to the shoulder coordinate frame 6 is denoted as d_{ew} .
- ❖ The Euclidean distance from the base coordinate frame 6 to the shoulder coordinate frame 7 is denoted as d_{wf} .
- ❖ Ef indicates the offset length between the KUKA last flange and the electrical end-effector mounted on top of the robot [48], as shown in Figure 3.4.2. This height is included in link 7 and is equal to 35 mm.

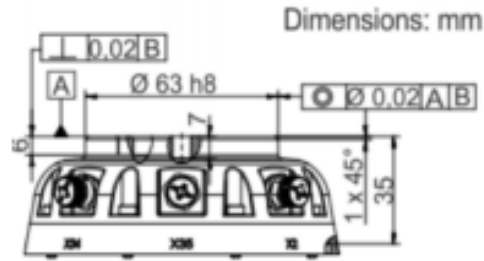


Figure 3.4.2 media flange electrical dimensions

3.4.2 Forward Kinematic for The KUKA LBR iiwa 7 R800 Robot:

Forward kinematics is employed to determine the position of the manipulator in space when the joint parameters are known for that position. In the case of an articulated manipulator, all seven joint angle values must be specified to calculate the end effector coordinates in space.

The overall forward kinematic of the robot is computed based on the previously derived D-H parameters using the homogeneous transformation matrix [48]. This matrix transforms the coordinate system $(i - 1)$ into the coordinate system i and is represented as follows:

$${}^{i-1}_iT = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1}d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

By incorporating the product of four fundamental transformations of DH parameters into the homogeneous transformation, the result becomes as follows:

➤ Frame 0 to frame 1 transformation matrix:

$${}^0_1T = \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 & 0 \\ s\theta_1 & c\theta_1 & 0 & 0 \\ 0 & 0 & 0 & d_{bs} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

➤ Frame 1 to frame 2 transformation matrix:

$${}^1_2T = \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s\theta_2 & -c\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

➤ Frame 2 to frame 3 transformation matrix:

$${}^2_3T = \begin{bmatrix} c\theta_3 & -s\theta_3 & 0 & 0 \\ 0 & 0 & -1 & -d_{se} \\ s\theta_3 & c\theta_3 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

➤ Frame 3 to frame 4 transformation matrix:

$${}^3_4T = \begin{bmatrix} c\theta_4 & -s\theta_4 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ s\theta_4 & c\theta_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

➤ Frame 4 to frame 5 transformation matrix:

$${}^4_5T = \begin{bmatrix} c\theta_5 & -s\theta_5 & 0 & 0 \\ 0 & 0 & 1 & d_{ew} \\ -s\theta_5 & -c\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6)$$

➤ Frame 5 to frame 6 transformation matrix:

$${}^5_6T = \begin{bmatrix} c\theta_6 & -s\theta_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s\theta_6 & -c\theta_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (7)$$

➤ Frame 6 to frame 7 transformation matrix:

$${}^6_7T = \begin{bmatrix} c\theta_7 & -s\theta_7 & 0 & 0 \\ 0 & 0 & -1 & -d_{wf} \\ s\theta_7 & c\theta_7 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (8)$$

The forward kinematic equation is determined by evaluating the transformation matrix for each frame individually and then multiplying all resulting matrices to determine the forward kinematics for the entire robot, as following below:

$${}^0_7T = {}^0_1T \times {}^1_2T \times {}^2_3T \times {}^3_4T \times {}^4_5T \times {}^5_6T \times {}^6_7T = \begin{bmatrix} R_{11} & R_{12} & R_{13} & P_X \\ R_{21} & R_{22} & R_{23} & P_Y \\ R_{31} & R_{32} & R_{33} & P_Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (9)$$

Where the rotation matrix is:

$$R_{11} = s7 * (s5 * (c4 * (s1 * s3 - c1 * c2 * c3) - c1 * s2 * s4) - c5 * (c3 * s1 + c1 * c2 * s3)) - c7 * (s6 * (s4 * (s1 * s3 - c1 * c2 * c3) + c1 * c4 * s2) + c6 * (c5 * (c4 * (s1 * s3 - c1 * c2 * c3) - c1 * s2 * s4) + s5 * (c3 * s1 + c1 * c2 * s3)))$$

$$R_{12} = s7 * (s6 * (s4 * (s1 * s3 - c1 * c2 * c3) + c1 * c4 * s2) + c6 * (c5 * (c4 * (s1 * s3 - c1 * c2 * c3) - c1 * s2 * s4) + s5 * (c3 * s1 + c1 * c2 * s3))) + c7 * (s5 * (c4 * (s1 * s3 - c1 * c2 * c3) - c1 * s2 * s4) - c5 * (c3 * s1 + c1 * c2 * s3))$$

$$R_{13} = c6 * (s4 * (s1 * s3 - c1 * c2 * c3) + c1 * c4 * s2) - s6 * (c5 * (c4 * (s1 * s3 - c1 * c2 * c3) - c1 * s2 * s4) + s5 * (c3 * s1 + c1 * c2 * s3))$$

$$\begin{aligned}
R_{21} &= c7 * (s6 * (s4 * (c1 * s3 + c2 * c3 * s1) - c4 * s1 * s2) + c6 * (c5 * (c4 * (c1 * s3 + c2 * c3 * s1) + s1 * s2 * s4) + s5 * (c1 * c3 - c2 * s1 * s3))) \\
&\quad - s7 * (s5 * (c4 * (c1 * s3 + c2 * c3 * s1) + s1 * s2 * s4) - c5 * (c1 * c3 - c2 * s1 * s3)) \\
R_{22} &= -c7 * (s5 * (c4 * (c1 * s3 + c2 * c3 * s1) + s1 * s2 * s4) - c5 * (c1 * c3 - c2 * s1 * s3)) - s7 * (s6 * (s4 * (c1 * s3 + c2 * c3 * s1) - c4 * s1 * s2) \\
&\quad + c6 * (c5 * (c4 * (c1 * s3 + c2 * c3 * s1) + s1 * s2 * s4) + s5 * (c1 * c3 - c2 * s1 * s3))) \\
R_{23} &= s6 * (c5 * (c4 * (c1 * s3 + c2 * c3 * s1) + s1 * s2 * s4) + s5 * (c1 * c3 - c2 * s1 * s3)) - c6 * (s4 * (c1 * s3 + c2 * c3 * s1) - c4 * s1 * s2) \\
R_{31} &= c7 * (c6 * (c5 * (c2 * s4 - c3 * c4 * s2) + s2 * s3 * s5) - s6 * (c2 * c4 + c3 * s2 * s4)) - s7 * (s5 * (c2 * s4 - c3 * c4 * s2) - c5 * s2 * s3) \\
R_{32} &= -c7 * (s5 * (c2 * s4 - c3 * c4 * s2) - c5 * s2 * s3) - s7 * (c6 * (c5 * (c2 * s4 - c3 * c4 * s2) + s2 * s3 * s5) - s6 * (c2 * c4 + c3 * s2 * s4)) \\
R_{33} &= s6 * (c5 * (c2 * s4 - c3 * c4 * s2) + s2 * s3 * s5) + c6 * (c2 * c4 + c3 * s2 * s4)
\end{aligned}$$

In addition to the position vector that representing the Cartesian coordinates of the end-effector relative to the base, as following below:

$$\begin{aligned}
P_X &= dew * (s4 * (s1 * s3 - c1 * c2 * c3) + c1 * c4 * s2) + dwf * (c6 * (s4 * (s1 * s3 - c1 * c2 * c3) + c1 * c4 * s2) - s6 * (c5 * (c4 * (s1 * s3 - c1 * c2 * c3) - c1 * s2 * s4) + s5 * (c3 * s1 + c1 * c2 * s3))) + c1 * dse * s2 \\
P_Y &= dse * s1 * s2 - dwf * (c6 * (s4 * (c1 * s3 + c2 * c3 * s1) - c4 * s1 * s2) - s6 * (c5 * (c4 * (c1 * s3 + c2 * c3 * s1) + s1 * s2 * s4) + s5 * (c1 * c3 - c2 * s1 * s3))) - dew * (s4 * (c1 * s3 + c2 * c3 * s1) - c4 * s1 * s2) \\
P_Z &= dbs + c2 * dse + dwf * (s6 * (c5 * (c2 * s4 - c3 * c4 * s2) + s2 * s3 * s5) + c6 * (c2 * c4 + c3 * s2 * s4)) + dew * (c2 * c4 + c3 * s2 * s4)
\end{aligned}$$

Where: $s_i: \sin \theta_i, c_i: \cos \theta_i$

3.4.3 Inverse Kinematic for The KUKA LBR iiwa 7 R800 Robot:

Inverse kinematics is the opposite of forward kinematics, as it uses the end effector position and orientation to calculate the joint parameters. By specifying the desired end effector position, algebraic methods are employed to solve the equations and calculate the joint angles.

The manipulator model in this study consists of seven joint angles, while the given end effector position and orientations can be specified by six parameters, resulting in one extra degree of freedom in the manipulator. This means that seven nonlinear equations need to be solved: four equations for the rotation matrix and three for the position vector. To find a solution, the target position (Pt) must lie within the robot's working range [57], as follows:

$$P_T = \begin{bmatrix} P_X \\ P_Y \\ P_Z \end{bmatrix} \quad (10)$$

The additional degree of freedom represents redundancy in the system. As a result, we can solve the inverse kinematics problem as two simpler problems [58]:

- 1) Arm link structure problem that provides inverse position kinematics:

From equations (9) and (10), the values of the joint angles θ_1 , θ_2 , θ_3 are obtained as follows:

$$\theta_1 = \text{Atan2}(P_Y, P_X) \quad (11)$$

Where $R_1 = \frac{P_X}{2c1} + \frac{P_Y}{2s1}$ from equation (11), and $R_2 = P_Z - d_{bs}$ Then,

$$\cos \theta_3 = \frac{R_1^2 + R_2^2 - d_{bs}^2 - d_{se}^2}{2d_{bs}d_{se}} = D \quad (12)$$

Since $\sin^2 \theta + \cos^2 \theta = 1$ Hence,

$$\theta_3 = \text{Atan2}(D, \pm \sqrt{1 - D^2}) \quad (13)$$

From equation (13), the result in θ_2 is:

$$\theta_2 = \text{Atan2}(R_1, -R_2) - \text{Atan2}(d_{bs} + d_{se}c3, d_{se}s3) \quad (14)$$

- 2) Wrist link structure problem that provides inverse orientation kinematics:

Where the rotation matrix ${}^0R = {}^0R {}^3R {}^6R$ Then,

$${}^3R = {}^0R^T {}^0R = {}^3R {}^4R {}^5R {}^6R \quad (15)$$

$${}^3_6R = \begin{bmatrix} s_4s_6 + c_4c_5c_6 & c_6s_4 - c_4c_5s_6 & -c_4s_5 \\ c_6s_5 & -s_5s_6 & c_5 \\ c_5c_6s_4 - c_4s_6 & -c_4c_6 - c_5s_4s_6 & -s_4s_5 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (16)$$

Hence, utilize the last column from equation 16 to solve for θ_4 , θ_5 , and θ_6 , as follows:

$$\theta_4 = \text{Atan2}(r_{33}, r_{13}) \quad (17)$$

$$\theta_5 = \text{Atan2}(\sqrt{r_{33}^2 + r_{13}^2}, r_{23}) \quad (18)$$

$$\theta_6 = \text{Atan2}(-r_{22}, r_{21}) \quad (19)$$

The angle θ_7 corresponds to the rotation around the Z-axis. When the Target Centre Axis P_{Tc} is parallel to the Y-axis of coordinate system 7, the projection of Target Centre onto the X-axis is zero [57], as follows:

$${}^7P_{Tc} = {}^7_0R {}^0P_{Tc} = {}^7_6R {}^6_5R {}^5_4R {}^4_3R {}^3_2R {}^2_1R {}^1_0R {}^0P_{Tc} \quad (20)$$

with ${}^7P_{Tc} = [0, {}^7P_{TcY}, {}^7P_{TcZ}, 0]^T$ and ${}^0P_{Tc} = [{}^0P_{TcX}, {}^0P_{TcY}, {}^0P_{TcZ}, 0]^T$ Then,

$${}^7_6R^{-1} {}^7P_{Tc} = {}^6_7R {}^7P_{Tc} = {}^6_5R {}^5_4R {}^4_3R {}^3_2R {}^2_1R {}^1_0R {}^0P_{Tc} \quad (21)$$

$$\begin{bmatrix} {}^7P_{TcY} \sin \theta_7 \\ {}^7P_{TcY} \cos \theta_7 \\ {}^7P_{TcZ} \\ 0 \end{bmatrix} = \begin{bmatrix} k \\ m \\ n \\ 0 \end{bmatrix} \quad (22)$$

From equation (22), the values of the joint angles θ_7 is obtained as follows:

$$\theta_7 = \text{Atan2}(k, m) \quad (23)$$

3.4.4 Jacobian Matrix for The End-Effector Velocities:

The Jacobian matrix in robotics establishes the relationship between joint velocities ($\dot{\theta}$) and end-effector velocities (V) of a robot manipulator. When the robot's joints move with specific velocities, it becomes essential to determine the corresponding velocity at which the end effector will move. The Jacobian provides this information, offering a link between joint velocities and end-effector velocities, as shown below [49]:

$${}^0V = {}^0J \cdot \dot{\theta} \quad (24)$$

Jacobian matrix serves as a crucial tool in this method for determining joint velocities and compensating for external forces and moments, thus enabling precise control and planning of robot movements and interactions with the environment [8]. Also, The Jacobian transpose maps Cartesian forces acting at the EEF into equivalent joint torques the relation is given as below:

$${}^0\tau = {}^0J^T \cdot {}^0f \quad (25)$$

3.4.5 Dynamic Model Due to Contact with Obstacles:

The dynamic model is crucial for implementing torque control-based methods, such as sensor less hand-guiding, where motor current measurements and friction models are used to determine the user's intention to move the end-effector (EEF) of the robot [50].

Furthermore, the dynamic model plays a role in calculating compensation torques and external forces. These torques are crucial for enabling the robot to interact with obstacles and perform tasks in structured environments [49]. The dynamic model's precision is essential for accurate calculations of torques due to joint dynamics, centrifugal effects, friction, and gravity, especially when only raw torque measurements at robot joints are available, this allows for intuitive and accurate control of the robot's motion.

The dynamic model can be found through Lagrange method or Newton-Euler algorithm. The Lagrangian Euler (LE) formulation is a standard and fundamental method used to derive the second-order dynamic equations of a robotic system. The Lagrangian equation is defined as the sum of the difference between the kinetic energy and potential energy of each joint parameter of the robot [52]. By considering the rigid link dynamics with the Lagrange method, the motion equations for the iiwa robot can be described as follows:

$$\tau_{\text{dyn}} = M(\theta)\ddot{\theta} + V(\theta, \dot{\theta})\dot{\theta} + G(\theta) + F_v\dot{\theta} + F_c\text{sign}(\dot{\theta}) \quad (26)$$

Where τ_{dyn} represents the required torque at each joint, with considering the external forces and moments.

$M(\theta)$ is the symmetric and positive-definite inertia matrix of links.

$V(\theta, \dot{\theta})$ is the centrifugal and Coriolis matrix of links.

$G(\theta)$ is the gravity vector of links.

F_v is the diagonal matrix of the viscous friction parameters.

F_c is the diagonal matrix of the Coulomb friction parameters.

After calculating the compensation torques and external forces, the extra torques due to contact forces with an obstacle (τ_c) must be computed to calculate the motion in the null space [50]. The process is as follows:

$$\tau_c = \tau_r - \tau_{\text{dyn}} \quad (27)$$

Where τ_r represents the raw torque measurements from the sensors at robot joints. This vector can be directly obtained if using a robot that provides such data, for example, the KUKA LBR iiwa 7 R800 industrial manipulator [49].

Alternatively, if only raw torque measurements at robot joints are available, the inverse dynamics equation of the robot shall be used, where the precision of the calculation relies mainly on the precision of the dynamical model used to describe the manipulator's dynamics [52]. Where the equation becomes as follows:

$$\ddot{\theta} = M(\theta)^{-1} \cdot (\tau_{\text{dyn}} - V(\theta, \dot{\theta})\dot{\theta} - G(\theta) - F_v\dot{\theta} - F_c\text{sign}(\dot{\theta})) \quad (28)$$

3.5 System Interface:

In this project, there are many system interfaces that are used. This section will present the details and properties of each interface separately, and it will also discuss the reasons for selecting them.

3.5.1 KUKA smartPAD Selection:

The smartPAD is the hand-held control panel designed for the industrial robot, encompassing all the essential operator control and display functions required for its operation, as shown in the Figure 3.5.1 and Figure 3.5.2. Moreover, it is equipped with a touch screen, known as the smartHMI, which can be conveniently operated using either a finger or a stylus, as demonstrated in Figure 3.5.3. Notably, there is no need for an external mouse or keyboard [18].

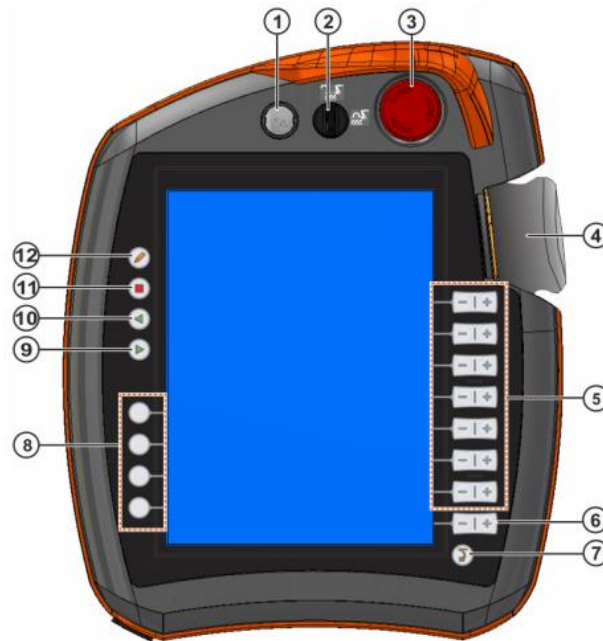


Figure 3.5.1 Front view of smartPAD

The description of Figure 3.5.1 is provided in Table 3.5.1:

Table 3.5.1 Description for the front view of the smartPAD

Number	Description
1	Button for disconnecting the smartPAD
2	Mode selector switch: used to change the operating mode
3	EMERGENCY STOP device: stops the robot in hazardous situations
4	6D mouse: no function
5	Jog keys: used to move the robot manually
6	Key for setting the override
7	Main menu key: The main menu key shows and hides the main menu on the smartHMI
8	User keys: the function of the user keys is freely programmable
9	Start key: the Start key is used to start a program
10	Start backwards key: no function
11	STOP key: the STOP key is used to stop a program that is running
12	Keyboard key: no function



Figure 3.5.2 Rear view of smartPAD

The description of Figure 3.5.2 is provided in Table 3.5.2:

Table 3.5.2 Description for the rear view of the smartPAD

Number	Description
1,3,5	<p>Enabling switches</p> <p>The enabling switches have 3 positions:</p> <ul style="list-style-type: none"> • Not pressed. • Centre position. • Fully pressed. <p>In operating modes T1, T2 and CRR, the manipulator can only be moved if one of the enabling switches is held in the central position.</p> <p>In the Automatic operating mode, the enabling switches have no function.</p>
2	Start key (green): used to start a program
4	USB connection: for FAT32-formatted USB sticks
6	Identification plate

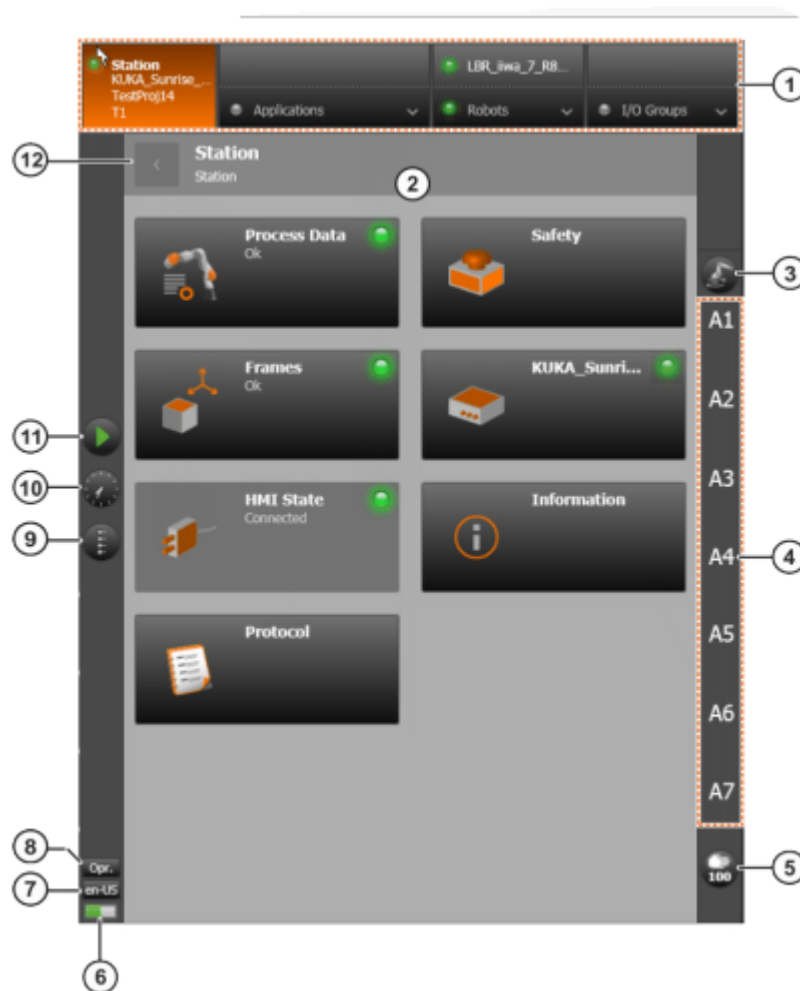


Figure 3.5.3 KUKA smartHMI user interface

The description of Figure 3.5.3 is provided in Table 3.5.3:

Table 3.5.3 Description for the smartHMI user interface

Number	Description
1	Navigation bar: main menu and status display.
2	Display area: display of the level selected in the navigation bar, here the Station level
3	Jogging options button: displays the current coordinate system for jogging with the jog keys.
4	Jog keys display: If axis-specific jogging is selected, the axis numbers are displayed here (A1, A2, etc.). If Cartesian jogging is selected, the coordinate system axes are displayed here (X, Y, Z, A, B, C). In the case of an LBR, the elbow angle (R) for executing a null space motion is additionally displayed.
5	Override button: Indicates the current override. Touching the button opens the Override window, in which the override can be set.
6	Life sign display: a steadily flashing life sign indicates that the smartHMI is active.
7	Language selection button: Indicates the currently set language.
8	User group button: Indicates the currently logged-on user group.
9	User key selection button: Touching the button opens the User key selection window.
10	Clock button: displays the system time. Touching the button displays the system time in digital format, together with the current date.
11	Jogging type button: displays the currently set mode of the Start key. Touching the button opens the Jogging type window, in which the mode can be changed.
12	Back button: Return to the previous view by touching this button.

In this user interface, as shown in Figure 3.5.4, seven sliders are presented, each corresponding to a Degree of Freedom (DOF) with reference to the joint limits and the actual value expressed in digits, along with torque measurements from the sensors at robot joints. Additionally, the cartesian position of the end effector with orientation values is displayed. Furthermore, real-time tracing of the path followed by the end-effector is also available.



Figure 3.5.4 A visual model of the seven sliders in KUKA smartPAD from [59]

3.5.2 Controller Selection:

The Sunrise Cabinet robot controller serves as the central brain of the robotic system, providing power distribution, communication interfaces, and precise control over the robot's movements and functionalities. It is equipped with cutting-edge technology, including a powerful quad-core processor, allowing it to handle complex computations and perform real-time control tasks with exceptional accuracy and speed [60]. In the Figure 3.5.5 provide Control unit overview:

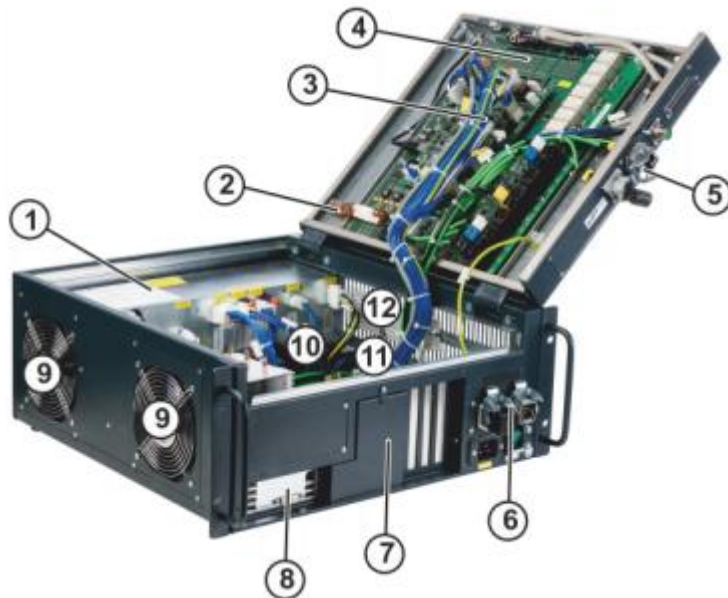


Figure 3.5.5 KUKA Sunrise Cabinet robot controller overview

The description of the components in Figure 3.5.5 are provided in Table 3.5.4:

Table 3.5.4 KUKA Sunrise Cabinet robot controller components

Number	Components
1	Low-voltage power supply unit
2	Memory card (EDS)
3	Power Management Board, Small Robot (PMB_SR)
4	Cabinet Interface Board, Small Robot (CIB_SR)
5	Interfaces
6	Main switch, interfaces
7	Network connections cover
8	PC interfaces
9	Fans
10	Batteries
11	Options
12	Brake resistor

The KUKA Sunrise Cabinet robot controller, equipped with its sophisticated Cabinet Interface Board (CIB_SR) and Power Management Board (PMB_SR), facilitates seamless communication among different components of the robot controller, as depicted in Figure 3.5.6. This feature ensures efficient data exchange and streamlined coordination of robot operations. Furthermore, the controller's capability to provide a continuous power supply through low-voltage power units and battery backup ensures a safe shutdown of the robot in the event of power disruptions [60].

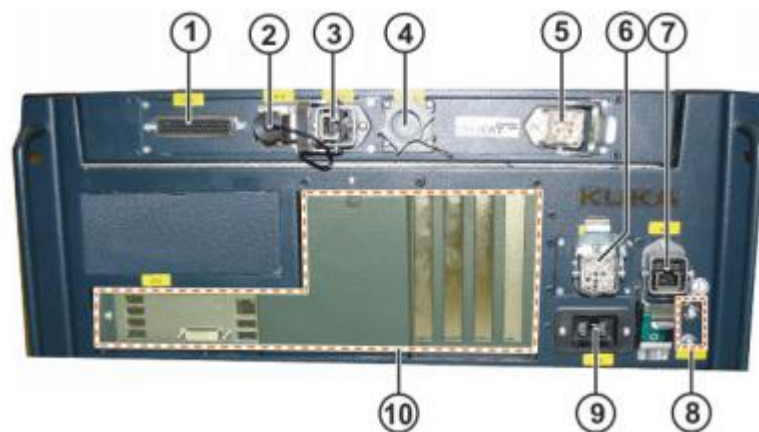


Figure 3.5.6 KUKA Sunrise Cabinet interfaces

The connection interfaces in Figure 3.5.6 are provided in Table 3.5.5:

Table 3.5.5 Connection interfaces for KUKA Sunrise Cabinet robot controller

Number	Interfaces
1	X11 interface
2	X19 smartPAD interface
3	X65 Extension interface
4	X69 Service interface
5	X650 Interface for media flange
6	X21 Manipulator interface
7	X66 KUKA Line Interface
8	PE connections
9	K1 Power supply connection
10	Control PC interfaces

3.5.3 Gripper Selection:

Collaborative grippers, also known as collaborative robotic grippers or cobot grippers, are advanced tools designed to work in tandem with human operators in a collaborative workspace [61]. Unlike traditional robotic grippers that require safety barriers to protect humans from potential hazards, collaborative grippers are equipped with safety features and sensors that allow them to safely interact with humans [62].

There are various types of collaborating grippers, each tailored to specific applications and environments. Some common types include: Vacuum Grippers, Electric Grippers, Magnetic Grippers, pneumatic Grippers, Soft Grippers, Adaptive Grippers, Dual Grippers and Geometric Grippers [62].

To utilize the robotic system effectively, particularly for rehabilitation tasks, careful consideration is essential when selecting the appropriate end-effector for the robot. In this project, the Collaborative Small Parts Gripper Co-act from SCHUNK Shop was chosen as the end-effector, as depicted in Figure 3.5.7. This gripper operates using electricity and

features a Media Flange Touch Electric [61]. The characteristics of the gripper are outlined in Table 3.5.6:



Figure 3.5.7 Medien-Flansch Touch Elektrisch Gripper

Table 3.5.6 Gripper Characteristics

Compatible robot	KUKA LBR iiwa 7/14
Robot flange	Media flange touch electric
LED strip light	Integrated
Grip Force, [N]	65- 230
Grip points	2
Payload, [kg]	0.7
Repeat accuracy [mm]	0.02
Size, [mm]	125.5 x 86.4 x 146.7 (X, Y, Z)
Weight, [kg]	1.15
Stroke per jaw [mm]	10
Integrated Force sensor	yes, inductive in two directions
Nominal voltage [V DC]	24
Max. current [A]	2
Max. ambient temperature [°C]	55

CHAPTER 4: IMPLEMENTATION OF UFPI AND DEVELOPMENT

This chapter outlines the creation of a user-friendly interface for controlling the KUKA LBR iiwa robot. It explains the control process, system flowchart, operational steps, encountered challenges (complex programming, safety, trajectory optimization, accuracy, integration), and their solutions. The objective is to simplify real-time robot control, improve programming, and ensure safe collaboration.

4.1 Introduction:

In the realm of collaborative robotics, the successful implementation of a user-friendly platform interface is a crucial stride toward enhancing the interaction between humans and robots. This chapter delves into the intricacies of the control process and system flowchart that underpin the development of the platform interface for the KUKA LBR iiwa 7 R800 robot. By employing the Java language on a smartPAD through the Sunrise Workbench software, the project seeks to create a seamless and efficient interface that empowers both novices and experts in robot control. The systematic progression through various configurations, programming decisions, and operational procedures is meticulously laid out, with a special focus on safety, precision, flexibility, and real-time control.

4.2 Control Process and System Flowchart:

This section presents a comprehensive understanding of the systematic approach employed to design, configure, and program the platform interface for the KUKA LBR iiwa 7 R800 robot. The control process will be elucidated step by step, emphasizing crucial configurations and programming choices aimed at guaranteeing the robot's efficient and secure operation. Additionally, a visual representation of the entire process will be provided via the System Flowchart, contributing significantly to the comprehension of the foundational mechanisms of the created interface.

4.2.1 Control Process and Safety Aspects:

In this project, a platform interface was developed using the Java language on a smartPAD through the Sunrise Workbench software. The following procedure comprehensively explain the programming process and the Java code that was written to create the UFPI interface. The complete code can be found in Appendix A.

Firstly, the robot's collision prevention is configured using two methods. The initial method is carried out through the Safety Configuration, where the permissible limit is defined, as depicted in Figure 4.2.1. If this limit is surpassed, the robot comes to a complete stop, requiring a program restart to start operation. Instructions for conducting Safety Configuration for this platform are outlined in Appendix B.

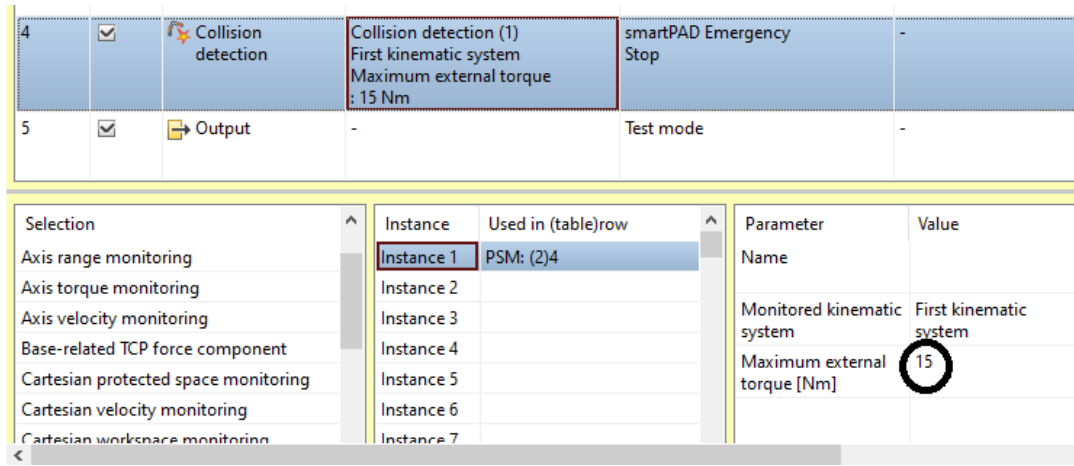


Figure 4.2.1 Collision detection in safety configuration

The second method is carried out through programming by specifying the maximum clash force and setting it to 7.5 Nm, Where the robot stops when the impact exceeds this limit, as illustrated in Figure 4.2.2:

```
public void setCollisionDetection() {
    BClash = true;
    JointTorqueCondition jt1 = new JointTorqueCondition(lbr_iwa_7_R800, JointEnum.J1, MinClashForce, MaxClashForce);
    JointTorqueCondition jt2 = new JointTorqueCondition(lbr_iwa_7_R800, JointEnum.J2, MinClashForce, MaxClashForce);
    JointTorqueCondition jt3 = new JointTorqueCondition(lbr_iwa_7_R800, JointEnum.J3, MinClashForce, MaxClashForce);
    JointTorqueCondition jt4 = new JointTorqueCondition(lbr_iwa_7_R800, JointEnum.J4, MinClashForce, MaxClashForce);
    JointTorqueCondition jt5 = new JointTorqueCondition(lbr_iwa_7_R800, JointEnum.J5, MinClashForce, MaxClashForce);
    JointTorqueCondition jt6 = new JointTorqueCondition(lbr_iwa_7_R800, JointEnum.J6, MinClashForce, MaxClashForce);
    JointTorqueCondition jt7 = new JointTorqueCondition(lbr_iwa_7_R800, JointEnum.J7, MinClashForce, MaxClashForce);
    ICondition collisionCondition = jt1.or(jt2, jt3, jt4, jt5, jt6, jt7);

    observing=getObserverManager().createConditionObserver(collisionCondition, NotificationType.OnEnable, new IAnyEdgeListener() {
        public void onAnyEdge(ConditionObserver conditionObserver, Date time, int missedEvents, boolean conditionValue) {
            if(conditionValue) {
                getLogger().info("force exceed");

                if(BClash) {
                    getLogger().info("Collision");
                    getApplicationControl().setApplicationOverride(0.0d);
                    mFlangeTouch.setLEDBlue(false);
                    getLogger().info("Impact the robot with an external force for only once");
                    ThreadUtil.millisSleep(1000);
                    count = 0;
                }
            }
        }
    });
}
```

Figure 4.2.2 Collision detection by coding

Afterward, the Hand Guiding Motion feature is activated with the addition of a velocity limit to achieve more accurate movement, especially within narrow spaces and crowded environments, as shown in Figure 4.2.3:

```
private HandGuidingMotion Hand;

@Override
public void initialize() {
    kuka_Sunrise_Cabinet_1 = getController("KUKA_Sunrise_Cabinet_1");
    lbr_iiwa_7_R800 = (LBR) getDevice(kuka_Sunrise_Cabinet_1,
        "LBR_iiwa_7_R800_1");
    mFlangeTouch = new MediaFlangeIOGroup(kuka_Sunrise_Cabinet_1);
    gripper = new HRCEGP40(mFlangeTouch);
    this.setCollisionDetection();
    Hand = new HandGuidingMotion();
}

@Override
public void run() {
    Hand.setJointVelocityLimit(1.5); //rads/s
    Hand.setCartVelocityLimit(50); //cm/s
    sampleIntervall=0;
}
```

Figure 4.2.3 Hand Guiding Motion feature

After that, the Impedance Mode feature is also activated, with low stiffness added on all axes to achieve more flexibility, as shown in Figure 4.2.4:

```
if (!ServoMotion.validateForImpedanceMode(lbr_iiwa_7_R800))
{
    getLogger().info("Validation of torque model failed - correct your mass property settings");
    getLogger().info("Servo motion will be available for position controlled mode only, until validation is performed");
}

getLogger().info("Hold position in impedance control mode");
CartesianImpedanceControlMode impedanceControlMode = new CartesianImpedanceControlMode();
impedanceControlMode.parametrize(CartDOF.X).setStiffness(stiffnessX);
impedanceControlMode.parametrize(CartDOF.Y).setStiffness(stiffnessY);
impedanceControlMode.parametrize(CartDOF.Z).setStiffness(stiffnessZ);
impedanceControlMode.parametrize(CartDOF.ROT).setStiffness(50);
```

Figure 4.2.4 Impedance Mode feature

Then, the program enters the while loop to start recording the user-desired movements, as shown in Figure 4.2.5:

```

int ret =0;
while(ret != 4)
{
    ret = getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION,InformationText
    switch (ret)
    {
    case 0:
        TypeMO[sampleIntervall] = "PTP";
        RecordMotion();
        break;
    case 1:
        TypeMO[sampleIntervall] = "MovePTP_With_Impedance";
        RecordMotion();
        break;
    case 2:
        TypeMO[sampleIntervall] = "LIN";
        RecordMotion();
        break;
    case 3:
        TypeMO[sampleIntervall] = "CIRC";
        RecordMotionCIRC();
        break;
    case 4:
        getLogger().info("Move away from the robot and reduce speed if it is high.");
        break;
    }
}

```

Figure 4.2.5 While loop to start recording the user-desired movements

After the user saves the movements, the program enters a while loop to execute the recorded movements, as shown in Figure 4.2.6. It continues within this loop, repeating the movements as desired by the user.

```

while(run != 0)
{
    mFlangeTouch.setLEDBlue(true);
    lbr_iiwa_7_R800.move(ptpToHomePosition);
    gripper.setOpen();
    ThreadUtil.millisSleep(1000);

    for(i=0;i<sampleIntervall;i++)
    {
        if(TypeMO[i] == "PTP"){
            getLogger().info("Go to point number " +(1+i) + " by using PTP Motion.");
            lbr_iiwa_7_R800.move(ptp(myPrintFrame[i])).setJointVelocityRel(y).setJointAccelerationRel(x));
        }
        else if (TypeMO[i] == "MovePTP_With_Impedance") {
            getLogger().info("Go to point number " +(1+i) + " by using Impedance Motion Mode.");
            observing.disable();
            runSmartServoLINMotion(impedanceControlMode);
            observing.enable();
        }
        else if (TypeMO[i] == "LIN") {
            getLogger().info("Go to point number " +(1+i) + " by using LIN Motion.");
            lbr_iiwa_7_R800.move(lin(myPrintFrame[i])).setJointVelocityRel(y).setJointAccelerationRel(x);
        }
        else{
            getLogger().info("Go to point number " +(1+i) + " by using CIRC Motion.");
            lbr_iiwa_7_R800.move(circ(auxiliaryFrame[i], myPrintFrame[i])).setJointVelocityRel(y).setJointAccelerationRel(x);
        }
    }

    if(myIndex[i] != i){
        if(gripper.getStatus()[0]){
            ThreadUtil.millisSleep(500);
            gripper.setClose();
            getLogger().info("Gripper is close.");
            ThreadUtil.millisSleep(500);
        }
    }
}

```

Figure 4.2.6 While loop to execute the recorded movements

In the background tasks, the User key is programmed to control the state of the gripper and the blue LED, as depicted in Figure 4.2.7.

```

IUserKeyBar gripperBar = getApplicationUI().createUserKeyBar("Gripper");
IUserKeyListener listener1 = new IUserKeyListener() {
    @Override
    public void onKeyEvent(IUserKey key, UserKeyEvent event) {
        if (event == UserKeyEvent.FirstKeyDown) {
            gripper.setClose();
            value = "Close";
        }
        else if (event == UserKeyEvent.SecondKeyDown) {
            gripper.setOpen();
            value = "Open";
        }
        key.setText(UserKeyAlignment.Middle, value);
    }
};

final IUserKey GripperKey1 = gripperBar.addDoubleUserKey(0, listener1, false);
GripperKey1.setText(UserKeyAlignment.TopMiddle, "grip");
GripperKey1.setText(UserKeyAlignment.Middle, value);
GripperKey1.setText(UserKeyAlignment.BottomMiddle, "release");

IUserKeyListener listener2 = new IUserKeyListener() {
    @Override
    public void onKeyEvent(IUserKey key, UserKeyEvent event) {
        if (event == UserKeyEvent.FirstKeyDown) {
            if(gripper.getStatus()[0]){
                gripper.setClose();
                value = "Close";
            }
            else{
                gripper.setOpen();
                value = "Open";
            }
        }
        else if (event == UserKeyEvent.SecondKeyDown) {
            if(mFlangeTouch.getLEDBLue()== true){
                mFlangeTouch.setLEDBLue(false);
                KeyLED = UserKeyLED.Red;
            }
            else{
                mFlangeTouch.setLEDBLue(true);
                KeyLED = UserKeyLED.Green;
            }
        }
        GripperKey1.setText(UserKeyAlignment.Middle,value);
        key.setText(UserKeyAlignment.Middle, "Omar");
        key.setLED(UserKeyAlignment.BottomRight, KeyLED,UserKeyLEDSIZE.Normal);
    }
};

IUserKey GripperKey2 = gripperBar.addDoubleUserKey(2, listener2, true);
GripperKey2.setText(UserKeyAlignment.TopMiddle, "obverse");
GripperKey2.setText(UserKeyAlignment.Middle,"Omar");
GripperKey2.setText(UserKeyAlignment.BottomLeft, "LED");
GripperKey2.setLED(UserKeyAlignment.BottomRight, KeyLED,UserKeyLEDSIZE.Normal);
grripperBar.publish();
}

```

Figure 4.2.7 User key programming

The complete code can be found in Appendix C.

4.2.2 System Flowchart:

The System Flowchart provides a visual representation for the flow of processes within the system. It illustrates the sequence of steps implemented to achieve a user-friendly platform interface, as shown in Figure 4.2.8:

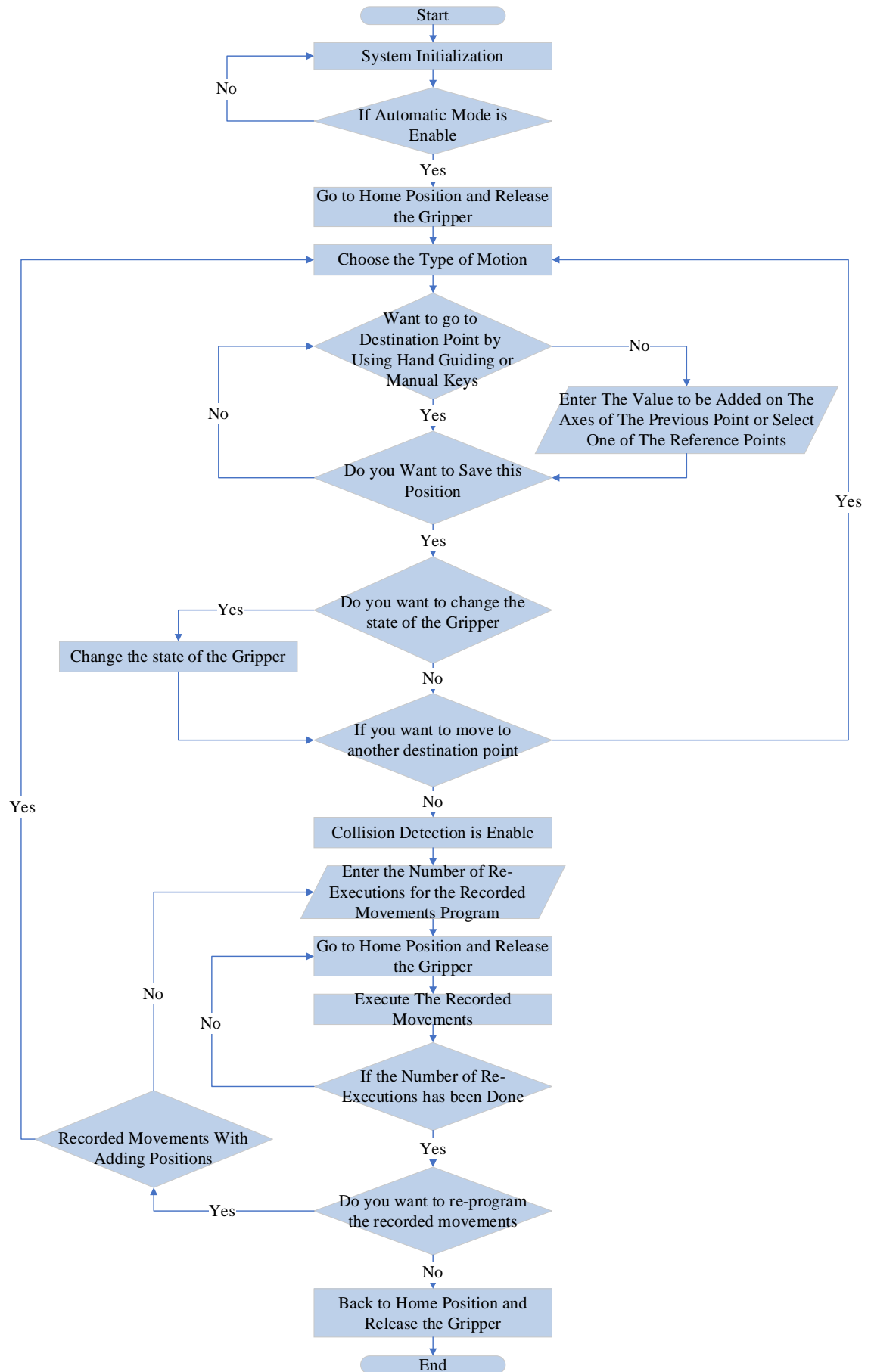


Figure 4.2.8 System Flowchart

4.3 Operation Manual:

When the programmed platform on the smartPAD is in operation, an informational comment appears indicating the need to enable automatic mode and then execute this program, as depicted in Figure 4.3.1:

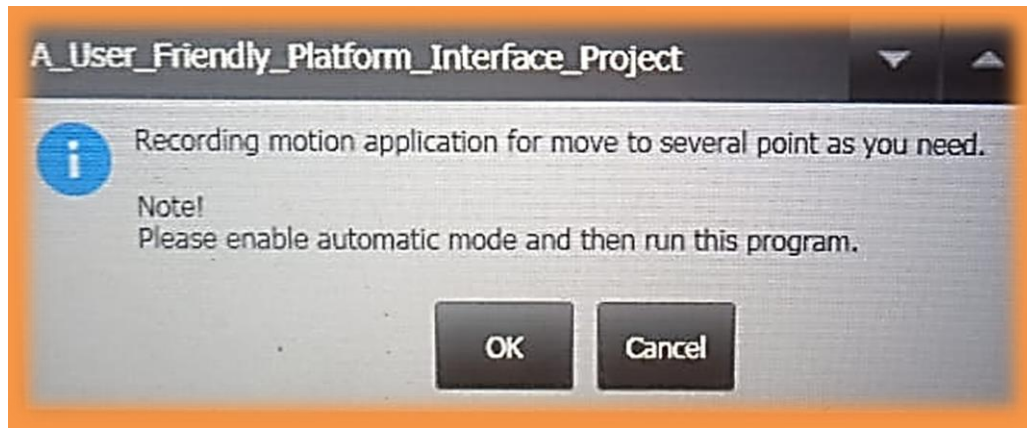


Figure 4.3.1 Informational comment indicating to enable automatic mode

Subsequently, a question command appears, inquiring whether the user intends to initiate movement. If the response is yes, further clarification is sought regarding the type of movement, as shown in Figure 4.3.2:

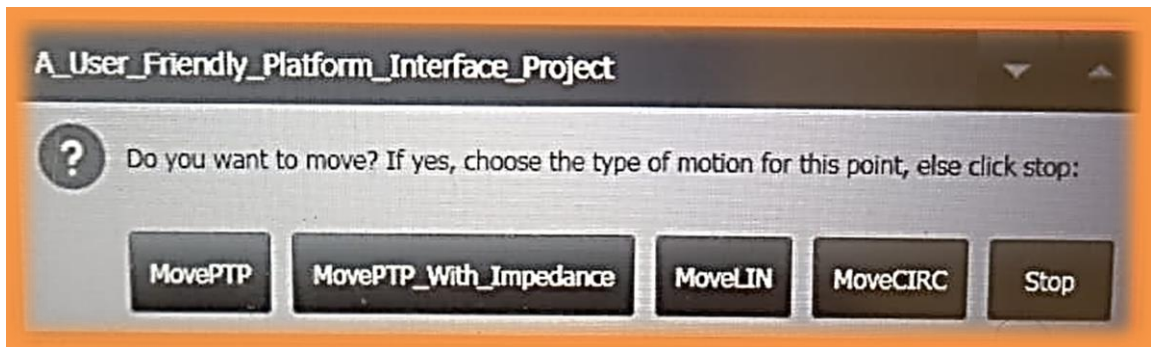


Figure 4.3.2 Question command indicating the type of motion

Once the initial point is chosen, a question command will appear, asking the user whether they wish to utilize the Hand Guiding Motion feature, Manual Keys, input a value along one of the axes of the preceding point until they transition to the subsequent point, or selecting one of the previously designated points as reference points. This process is illustrated in Figure 4.3.3:

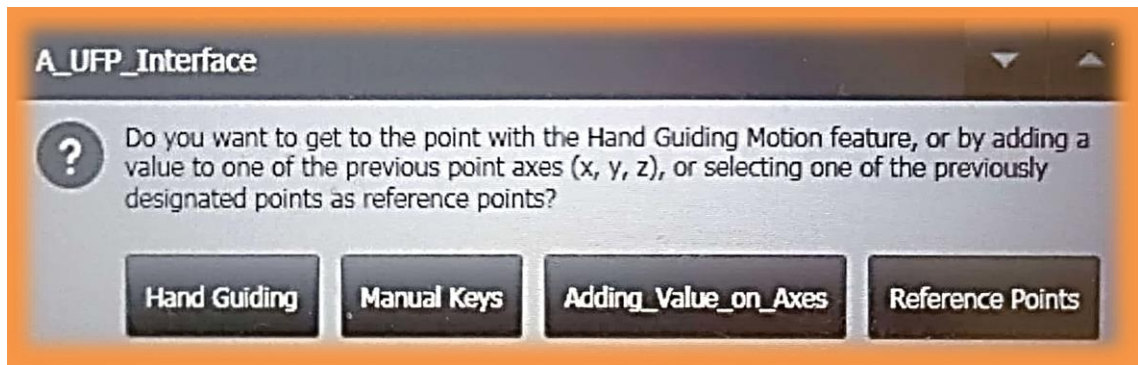


Figure 4.3.3 Question command asking about the approach to reach the destination point

To control the robot using the Manual Keys (Jog keys), you need in the first to transition from automatic mode to T1 mode. Subsequently, move the robot to your desired location, after that switch back to automatic mode, press the 'OK' button, and then click the Start key to run the interface and complete the program. As shown in the Figure 4.3.4:

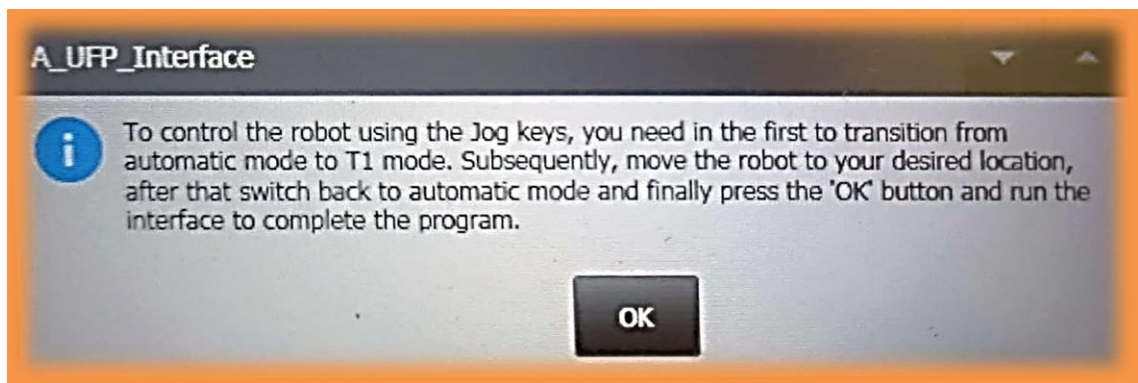


Figure 4.3.4 Informational comment explaining how to use the Jog Keys

If the choice to add a value to any of the axes of the previous point is opted for, with the intention of transitioning to the next point, the Question command will activate to determine the specific axis onto which the value is intended to be added. This procedure is exemplified in Figure 4.3.5:

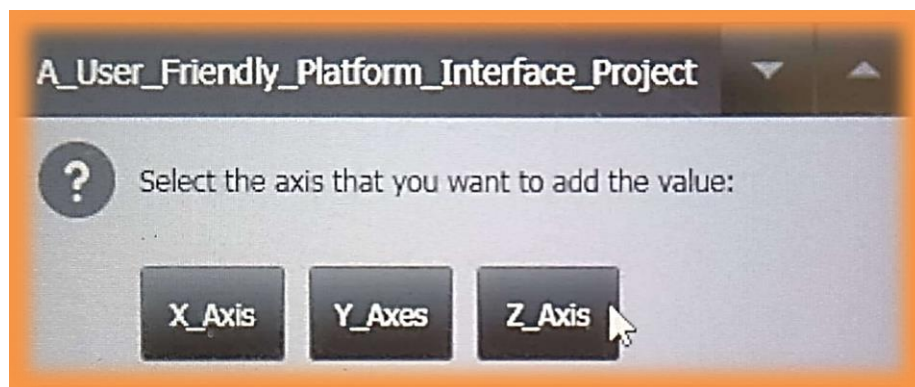


Figure 4.3.5 Question command to determine the specific axis

Once an axis is chosen, the question command surfaces to know the amount of displacement for its addition to the chosen axis, as depicted in Figure 4.3.6. But it must be noted that the resultant axis value post-addition must fall within the robot's operational workspace. To establish the positive orientation of the End Effector (EEF) axis, the right-hand rule is employed, as illustrated in Figure 4.3.7:

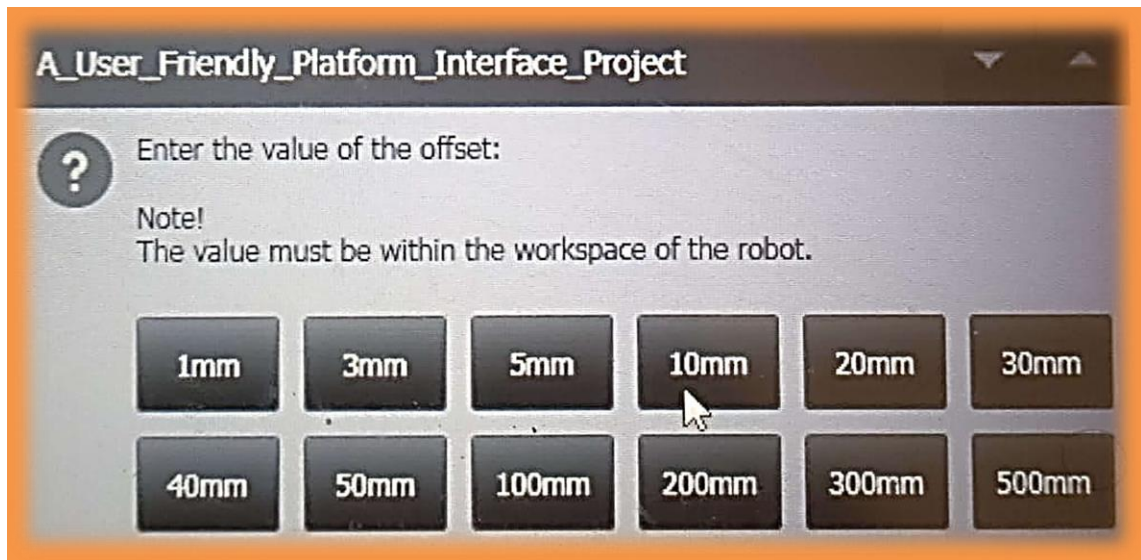


Figure 4.3.6 Question command to know the amount of offset

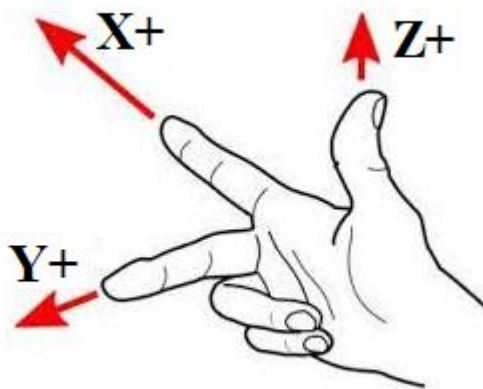


Figure 4.3.7 Right-Hand rule to know the direction of the coordinates for the EEF

Therefore, once the magnitude of the required axis offset is defined, the question command emerges to indicate the sign of the offset value, as depicted in Figure 4.3.8:

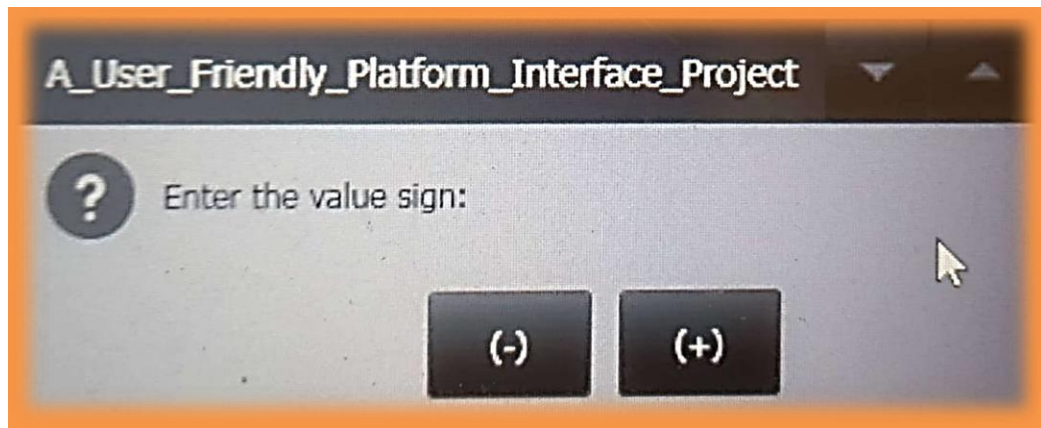


Figure 4.3.8 Question command to indicate the sign of the offset value

If the option to select one of the previously designated reference points is chosen, aiming to proceed to it as the next point, the Question command will activate to ascertain the number of the reference point. It is important to note that entering a number for a reference point that hasn't been created before will cause the robot to move to the last saved reference point. This process is illustrated in Figure 4.3.9:

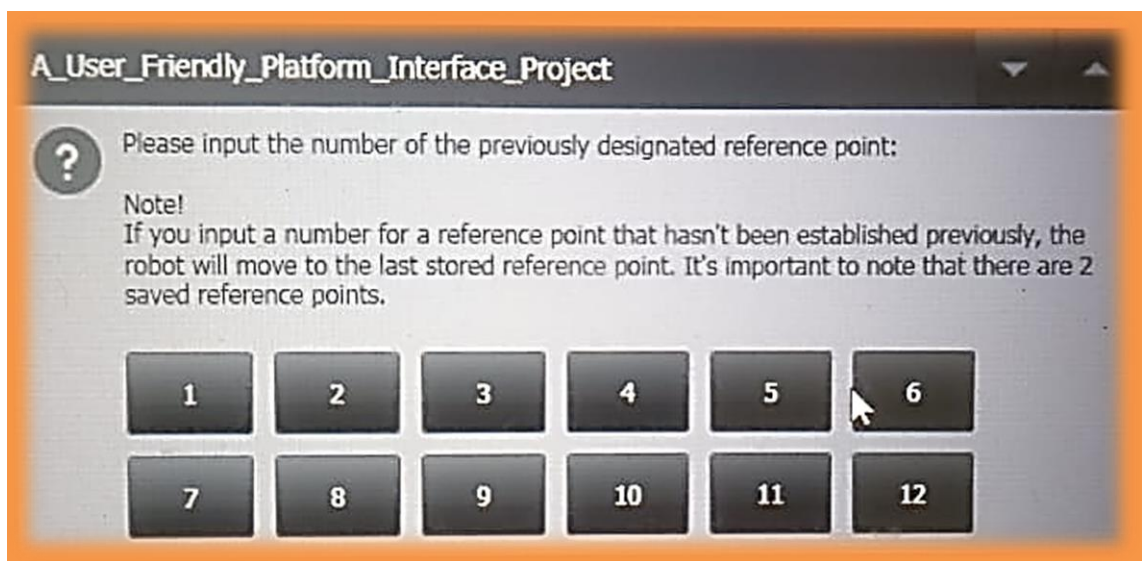


Figure 4.3.9 Question command to ascertain the number of the previously designated reference point

After selecting the movement type and the method to reach the destination point, a question command appears to seek user confirmation on whether to save the new position or not, depicted in Figure 4.3.10:

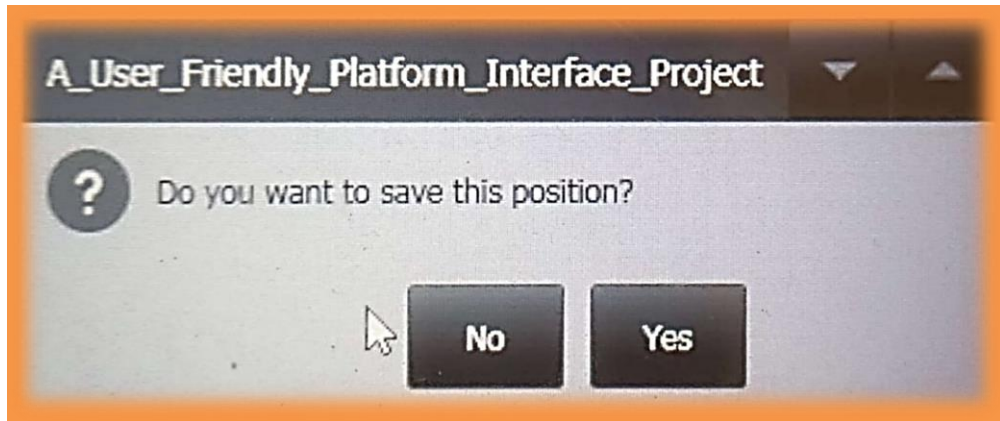


Figure 4.3.10 Question command to save the new position

Afterward, it displays a question command whether the user intends to change the state of the Gripper or not, as depicted in Figure 4.3.11:

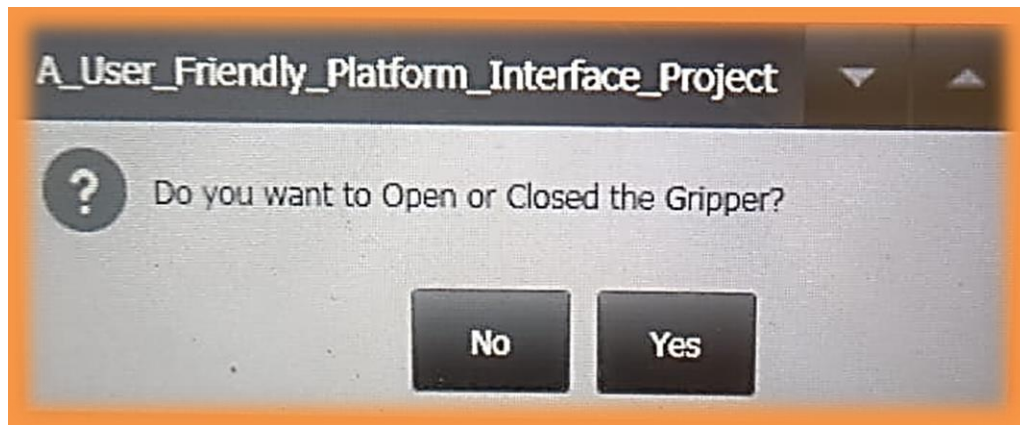


Figure 4.3.11 Question command to change the state of the Gripper

When selecting the circular motion type, In the first must be proceed to the auxiliary point and then proceed to the destination point. It's essential to note that the destination point must be lies along the circular path with the auxiliary point; otherwise, an error will occur during execution. The option to Movement with Impedance Mode feature can also be selected.

Before proceeding to select the next point, the Question command appears to determine whether the user wants to save the recorded point as a reference point for return use again or not, as shown in the Figure 4.3.12. It should be noted that the ability to save reference points is restricted to a maximum of 12 points, because There is a limit of twelve buttons per Modal Dialog. This presents one of the challenges with the smartPAD: it does not allow direct value input; instead, values must be selected from a dialog.

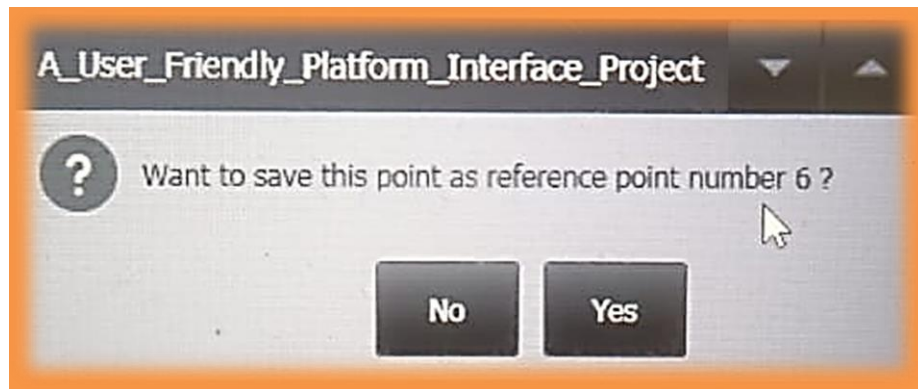


Figure 4.3.12 Question command to save the recorded point as a reference point

After saving the movements program, the question command appears to determine the desired number of re-executions for the recorded movements, as illustrated in Figure 4.3.13:

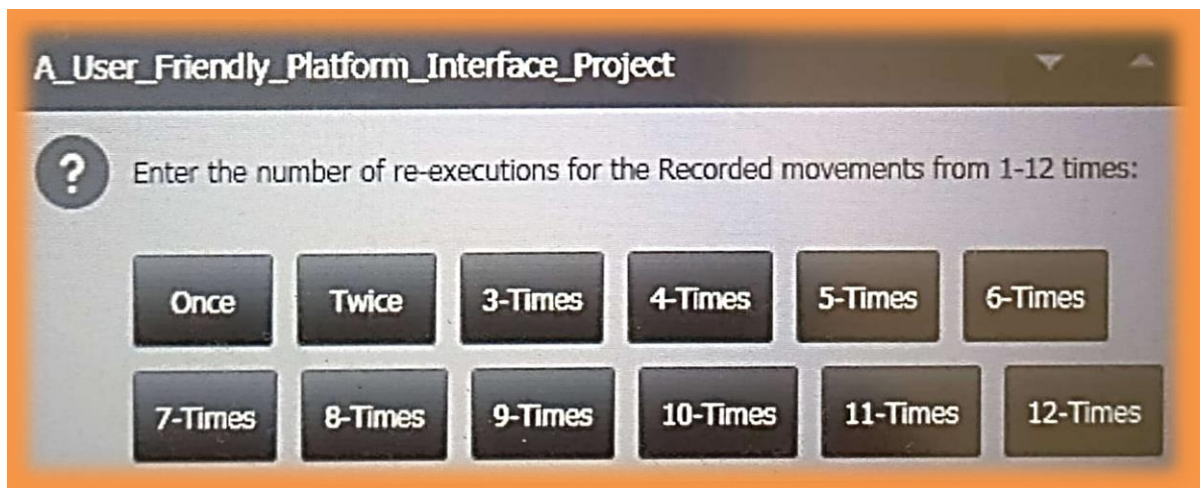


Figure 4.3.13 Question command to determine the number of re-executions

Finally, a question command appears to allow the user to choose whether he want to re-program the recorded movements with or without adding positions, or terminate the program, as shown in Figure 4.3.14:

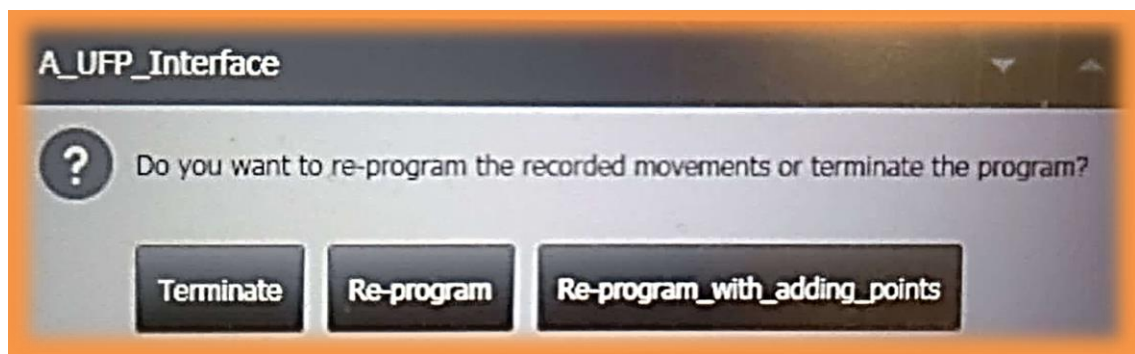


Figure 4.3.14 Question command to re-program the recorded movements or terminate the program

4.4 Problems and Solutions:

While advancing user-friendly control interface for non-programmers, this project encountered various challenges and problems that were overcome in order to achieve successful completion. These problems included:

- 1) **Complex Robot Programming:** The complexity of robot programming, especially with using the Java language, posed a challenge.
- 2) **Human-Robot Safety:** Ensuring safe interaction between human and robot during collaboration was a concern.
- 3) **Trajectory Optimization:** Optimizing robot trajectories for efficient movement was a difficult challenge.
- 4) **Enhancing Accuracy and Calibration:** Improving the robot's precision and calibration, including the tools and base frames, proved to be a big difficulty.
- 5) **System Integration:** Integrating with other systems and meeting specific application requirements was complex.
- 6) **Data Entry:** Data entry presented a problem due to the inability to input values and data in real time through the SmartPAD's touch screen or keyboard.

To tackle these challenges, the project aimed to provide solutions for these problems as follows:

- 1) To address complex programming, a user-friendly Java-based interface was developed by using the smartPAD and Sunrise Workbench software.
- 2) Safety was ensured by implementing features like automatic collision stop, and impedance control.
- 3) Trajectory optimization was achieved through real-time control and hand guiding features.
- 4) Accuracy and calibration were improved using advanced techniques in the hand guiding motion to enhance robot performance in narrow spaces, and TCP calibration.
- 5) Integration challenges were overcome by developing a comprehensive interface compatible with various systems.
- 6) The data and value entry issue were resolved by offering predetermined selections and granting users the flexibility to make choices according to the most selected values.

CHAPTER 5: RESEARCH PLAN AND CONCLUSIONS

This chapter provides a comprehensive overview of the research plan and its outcomes. It outlines the chosen research methodology, describing how the research problem was approached and the objectives were achieved. The chapter offers a detailed description of the developed system, and it highlights the advancements in human-robot collaboration, safety, and productivity. The chapter also presents future research directions, to further expand the impact and applicability of the developed system.

5.1 Research Methodology:

This research adopts a descriptive and exploratory research design to address the identified research problem and accomplish the specified objectives. The primary focus is on developing a user-friendly interface for real-time control of the KUKA LBR iiwa 7 R800 robot, with a particular emphasis on simplifying programming, enhancing flexibility, ensuring safety, and improving calibration processes.

The methodology involves qualitative data collection, primarily focusing on the development of a Java-based platform interface for controlling the robot. This data undergoes comprehensive analysis, involving a detailed examination of the developed Java code, the logical flow of operations, and the effectiveness of the interface in achieving the intended research objectives.

The validity and reliability of the research are upheld through meticulous documentation of the development process, code logic, and operational procedures. Rigorous testing of the developed interface is conducted to verify its consistency, repeatability, and alignment with the research goals. It is important to acknowledge the limitations of the research, which include the focus on the specific KUKA LBR iiwa 7 R800 robot model and its integration into assembly tasks.

Ethical considerations are paramount throughout the research, with a strong emphasis on prioritizing the safety of human participants and ensuring the secure operation of the robot during the testing and implementation phases. Personal or sensitive data is not involved in the research process.

5.2 System Description:

The system description encompasses various components, interfaces, and functionalities that contribute to the seamless real-time control and operation of the robot within different contexts. These components include the following:

1. **Hardware Components:** The core hardware components include the KUKA LBR iiwa 7 R800 robot, boasting seven degrees of freedom and integrated force/torque sensors for precise and flexible movements. The smartPAD control panel, equipped with a touch screen and intuitive user interface, facilitates direct interaction with the robot. The Sunrise Cabinet robot controller acts as the central processing hub, overseeing power distribution, communication interfaces, and overall robot operations.
2. **Software Components:** The system relies on the Sunrise Workbench software, offering a programming environment for configuring robot behavior and control logic. The Java programming language is employed to craft a user-friendly platform interface on the smartPAD, enabling real-time control and communication with the robot. Proprietary software ensures smooth communication between the smartPAD and the robot controller, facilitating command execution and movement coordination.
3. **Communication Interfaces:** Effective communication is established through wireless connectivity between the smartPAD and the robot controller, enabling seamless data exchange and command execution. The robot controller interfaces with actuators, sensors, and power distribution units to translate user commands into precise robotic movements.
4. **System Functionality:** At its core, the system revolves around the user-friendly platform interface created using the Java language on the smartPAD. This interface empowers users with varying levels of expertise to control the robot in real time. Touch-based interactions on the smartPAD initiate, guide, and monitor the robot's movements. Safety features, such as collision detection and impedance control, ensure secure human-robot collaboration, while trajectory optimization enhances the efficiency and precision of robot tasks.
5. **Integration and Compatibility:** Designed to seamlessly integrate with existing production systems, the system can be incorporated into networked environments, enabling efficient data exchange and coordination. Its compatibility with the KUKA LBR iiwa 7 R800 model ensures a smooth integration process to meet evolving industry needs.

5.3 Contribution and Significance:

In terms of contributions, this research advances the field of collaborative robotics by providing a tangible user-friendly interface that facilitates real-time control of the KUKA LBR iiwa 7 R800 robot. The interface addresses significant challenges related to complex programming, safety considerations, trajectory optimization, accuracy enhancement, and system integration. Its development holds implications for enhancing human-robot collaboration and automating manual assembly processes, aligning with the broader goals of efficient and safe industrial operations.

The significance of this research lies in its potential to extend the capabilities of collaborative robots in industrial settings, particularly in the context of manual assembly tasks. The user-friendly interface, created through the Java language on the smartPAD, opens avenues for wider adoption of robotic technology among individuals without prior programming knowledge. Additionally, it underscores the importance of safety measures and calibration in ensuring successful and productive human-robot collaboration. The subsequent chapter will present the implementation results, discuss challenges and their resolutions, and assess user feedback to gauge the interface's usability and effectiveness.

5.4 Conclusions and Future Work:

This research embarked on the development of a UFPI interface for real-time control of the KUKA LBR iiwa robot, addressing challenges related to complex programming, safety, trajectory optimization, accuracy, and system integration. The designed Java-based platform interface on the smartPAD, in conjunction with the Sunrise Workbench software, succeeded in providing an accessible and efficient means for users to interact with the robot.

The research demonstrated the significance of human-robot collaboration in enhancing industrial processes, particularly in manual assembly tasks. The integration of safety features, such as collision detection and impedance control, ensured secure interaction between humans and the robot. Hand guiding has played a significant role in enhancing the capability to achieve more precise positioning within confined spaces and crowded environments. Trajectory optimization and real-time control improved the robot's efficiency and precision, contributing to enhanced productivity.

The results indicated that the developed interface touch-based streamlined the programming process, making it approachable for individuals without prior programming knowledge, and enabled users to guide and monitor the robot's movements in real time. The modular architecture of the system facilitated integration into existing production setups, opening avenues for broader industrial adoption.

While this research achieved significant milestones in enhancing the control and collaboration of the KUKA LBR iiwa robot, there are several ways for future exploration and improvement remain, such as:

- ✚ Advanced Control Algorithms: Future research could delve into the development and implementation of advanced control algorithms, such as machine learning and adaptive control, to further enhance the robot's capabilities and adaptability to changing environments.
- ✚ Multi-Robot Collaboration: Investigating multi-robot collaboration scenarios could lead to improved efficiency and coordination in complex manufacturing processes, where multiple robots work together seamlessly.
- ✚ Enhanced Safety Mechanisms: Further refinement of safety mechanisms, including predictive collision avoidance and real-time risk assessment, could elevate the level of human-robot collaboration and safety in industrial environments.
- ✚ Integration with IoT and Industry 4.0: Exploring the integration of the developed system with Internet of Things (IoT) technologies and Industry 4.0 principles could lead to more comprehensive and interconnected manufacturing processes.
- ✚ User Experience Optimization: Future work could focus on improving the user interface to make interactions even more user-friendly, and inclusion haptic feedback with augmented reality elements.
- ✚ Using other software for integrating them with the platform such as V-REP, RoboDK, and MATLAB to further elevate the capabilities and applicability of the developed system.

REFERENCES:

- [1] “iiwa feel you,” Germany, 2016. Accessed: Jul. 16, 2023. [Online]. Available: https://www.kuka.com/-/media/kuka-downloads/imported/8350ff3ca11642998dbdc81dcc2ed44c/db_lbr_iiwa_en.pdf?rev=85b58d22fba84da0bde8968da8827a14&hash=4F4A8F2DC3FD3E2A39851820374F61E0
- [2] Y. R. Stürz, L. M. Affolter, and R. S. Smith, “Parameter Identification of the KUKA LBR iiwa Robot Including Constraints on Physical Feasibility,” in *IFAC-PapersOnLine*, Elsevier B.V., Jul. 2017, pp. 6863–6868. doi: 10.1016/j.ifacol.2017.08.1208.
- [3] “LBR iiwa | KUKA AG.” <https://www.kuka.com/en-de/products/robot-systems/industrial-robots/lbr-iiwa> (accessed Jul. 16, 2023).
- [4] D. Calandra, A. Cannavò, and F. Lamberti, “Improving AR-powered remote assistance: a new approach aimed to foster operator’s autonomy and optimize the use of skilled resources,” *International Journal of Advanced Manufacturing Technology*, vol. 114, no. 9–10, pp. 3147–3164, Jun. 2021, doi: 10.1007/s00170-021-06871-4.
- [5] Z. Gao, T. Wanyama, I. Singh, A. Gadhri, and R. Schmidt, “From industry 4.0 to robotics 4.0 - A conceptual framework for collaborative and intelligent robotic systems,” in *Procedia Manufacturing*, Elsevier B.V., 2020, pp. 591–599. doi: 10.1016/j.promfg.2020.03.085.
- [6] I. Granata, M. Faccio, and M. Calzavara, “Energy expenditure and makespan multi-objective optimization for cobots systems design,” *Procedia Comput Sci*, vol. 217, pp. 126–135, 2023, doi: 10.1016/j.procs.2022.12.208.
- [7] S. Vaidya, P. Ambad, and S. Bhosle, “Industry 4.0 - A Glimpse,” in *Procedia Manufacturing*, Elsevier B.V., 2018, pp. 233–238. doi: 10.1016/j.promfg.2018.02.034.
- [8] Z. Fu, E. Spyarakos-Papastavridis, Y. H. Lin, and J. S. Dai, “Analytical Expressions of Serial Manipulator Jacobians and their High-Order Derivatives based on Lie Theory*,” in *Proceedings - IEEE International Conference on Robotics and Automation*, Institute of Electrical and Electronics Engineers Inc., May 2020, pp. 7095–7100. doi: 10.1109/ICRA40945.2020.9197131.
- [9] A. Jain, H. Singh, R. A. Bobby, S. K. Saha, S. Kumar, and S. D. Roy, “Repeatability measurement and kinematic identification of LBR iiwa 7 R800 using monocular camera,” in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Jul. 2019. doi: 10.1145/3352593.3352617.
- [10] “LBR iiwa 7 R800 Specification KUKA Roboter GmbH,” 2022. Accessed: Jul. 05, 2023. [Online]. Available: https://www.kuka.com/-/media/kuka-downloads/imported/8350ff3ca11642998dbdc81dcc2ed44c/0000246832_en.pdf?rev=17b8a807a70745d4a16627fae9886121&hash=FBD6DB831EED428F307E6C5A6B79FA08

- [11] V. Chawda and G. Niemeyer, "Toward Torque Control of a KUKA LBR IIWA for Physical Human-Robot Interaction," in *IEEE/RSJ*, 2017. doi: 10.0/Linux-x86_64.
- [12] U. Dombrowski, T. Stefanak, and A. Reimer, "Simulation of human-robot collaboration by means of power and force limiting," in *Procedia Manufacturing*, Elsevier B.V., 2018, pp. 134–141. doi: 10.1016/j.promfg.2018.10.028.
- [13] A. Zanella, A. Cisi, M. Costantino, M. Di Pardo, G. Pasquettaz, and G. Vivo, "Criteria Definition for the Identification of HRC Use Cases in Automotive Manufacturing," *Procedia Manuf*, vol. 11, pp. 372–379, 2017, doi: 10.1016/j.promfg.2017.07.120.
- [14] S. Mokaram et al., "A ROS-integrated API for the KUKA LBR iiwa collaborative robot," in *IFAC-PapersOnLine*, Elsevier B.V., Jul. 2017, pp. 15859–15864. doi: 10.1016/j.ifacol.2017.08.2331.
- [15] J. James, G. Clarke, R. Mathew, B. Mulkeen, and N. Papakostas, "On Reducing the Localisation Error of Modern Mobile Robotic Platforms," in *Procedia CIRP*, Elsevier B.V., 2022, pp. 168–173. doi: 10.1016/j.procir.2022.09.067.
- [16] A. K. Ramasubramanian and N. Papakostas, "Operator - Mobile robot collaboration for synchronized part movement," in *Procedia CIRP*, Elsevier B.V., 2020, pp. 217–223. doi: 10.1016/j.procir.2020.05.228.
- [17] "Software Option KUKA Sunrise.ApplicationFramework 1.2 For KUKA Sunrise.OS 1.16 For KUKA Sunrise.Workbench 1.16," 2018. [Online]. Available: www.kuka.com
- [18] "System Software KUKA Sunrise.OS 1.16 KUKA Sunrise.Workbench 1.16 Operating and Programming Instructions for System Integrators," 2019. [Online]. Available: www.kuka.com
- [19] "Kuka LBR iiwa Programming Learn Module | UNSW Making." <https://www.making.unsw.edu.au/learn/kuka-lbr-iiwa-programming-learn-module/> (accessed Jul. 07, 2023).
- [20] M. Safeea and P. Neto, "KUKA Sunrise Toolbox: Interfacing Collaborative Robots with MATLAB," *IEEE Robot Autom Mag*, vol. 26, no. 1, pp. 91–96, Mar. 2019, doi: 10.1109/MRA.2018.2877776.
- [21] M. Safeea and P. Neto, "User's manual: KUKA Sunrise Toolbox," Portugal, Sep. 2018. Accessed: Jul. 07, 2023. [Online]. Available: <https://www.researchgate.net/publication/325402564>
- [22] "GitHub - Modi1987/KST-advanced-examples: Advanced projects on controlling KUKA iiwa from MATLAB using KUKA Sunrise Toolbox (KST)." <https://github.com/Modi1987/KST-advanced-examples> (accessed Jul. 09, 2023).
- [23] "Detailed: Controlling IIWA from Vrep using KST, part 4 - YouTube." <https://www.youtube.com/watch?v=eFIAX2gsz1g&list=PLz558OYgHuZdVzTaB79iM8Y8u6EjFe0d8&index=38> (accessed Jul. 09, 2023).

- [24] “Using A Simulator For Programming In Robotic,” 2023. Accessed: Jul. 10, 2023. [Online]. Available: <https://www.simprotic.org/wp-content/uploads/2023/06/SIMPROTIC-EN-version-3.0-2.pdf>
- [25] A. Atanas and G. Atanas, “Comparative Analysis of Robodk and Robot Operating System For Solving Diagnostics Tasks in Off-Line Programming,” in *International Conference Automatics and Informatics*, Atanas Atanassov and Atanas Garbev, Eds., Bulgaria, Varna: International Conference Automatics and Informatics, Oct. 2020, pp. 1–3.
- [26] “Basic Guide - RoboDK Documentation.” <https://robodk.com/doc/en/Basic-Guide.html#Start> (accessed Jul. 10, 2023).
- [27] “GitHub - Modi1987/RoboDKiiwaInterface: Drivers for controlling KUKA iiwa from RoboDK.” <https://github.com/Modi1987/RoboDKiiwaInterface> (accessed Jul. 09, 2023).
- [28] T. Al-Geddawy, “A digital twin creation method for an opensource low-cost changeable learning factory,” in *Procedia Manufacturing*, Elsevier B.V., 2020, pp. 1799–1805. doi: 10.1016/j.promfg.2020.10.250.
- [29] T. Rossmeissl, E. Groß, L. Zarco, T. Schlegel, J. Siegert, and T. Bauernhansl, “Approach for extending evaluation criteria for scalable and modular industrial robots,” in *Procedia CIRP*, Elsevier B.V., 2019, pp. 1022–1027. doi: 10.1016/j.procir.2019.03.245.
- [30] A. Hietanen, R. Pieters, M. Lanz, J. Latokartano, and J. K. Kämäräinen, “AR-based interaction for human-robot collaborative manufacturing,” *Robot Comput Integr Manuf*, vol. 63, Jun. 2020, doi: 10.1016/j.rcim.2019.101891.
- [31] S. Hjorth and D. Chrysostomou, “Human–robot collaboration in industrial environments: A literature review on non-destructive disassembly,” *Robotics and Computer-Integrated Manufacturing*, vol. 73. Elsevier Ltd, Feb. 01, 2022. doi: 10.1016/j.rcim.2021.102208.
- [32] F. Semeraro, A. Griffiths, and A. Cangelosi, “Human–robot collaboration and machine learning: A systematic review of recent research,” *Robotics and Computer-Integrated Manufacturing*, vol. 79. Elsevier Ltd, Feb. 01, 2022. doi: 10.1016/j.rcim.2022.102432.
- [33] M. Vistein, J. Faber, C. Schmidt-Eisenlohr, and D. Reiter, “Automated handling of auxiliary materials using a multi-kinematic gripping system,” in *Procedia Manufacturing*, Elsevier B.V., 2019, pp. 1276–1283. doi: 10.1016/j.promfg.2020.01.220.
- [34] K. Chen, J. Wen, J. Wu, and Z. Ji, “Automated robot-based large-scale 3D surface imaging,” in *Procedia Computer Science*, Elsevier B.V., 2020, pp. 2949–2958. doi: 10.1016/j.procs.2020.09.208.
- [35] F. Müller, C. Deuerlein, and M. Koch, “Cyber-physical-system for representing a robot end effector,” in *Procedia CIRP*, Elsevier B.V., 2021, pp. 307–312. doi: 10.1016/j.procir.2021.05.071.

- [36] M. Ikeda, N. Chitturi, M. Ganglbauer, and A. Pichler, “Knowledge based accuracy improvement in programming by demonstration of point based processes,” in *Procedia Manufacturing*, Elsevier B.V., 2021, pp. 16–23. doi: 10.1016/j.promfg.2021.10.004.
- [37] V. V. Nair, D. Kuhn, and V. Hummel, “Development of an easy teaching and simulation solution for an autonomous mobile robot system,” in *Procedia Manufacturing*, Elsevier B.V., 2019, pp. 270–276. doi: 10.1016/j.promfg.2019.03.043.
- [38] D. Calandra, F. G. Praticcò, A. Cannavò, C. Casetti, and F. Lamberti, “Digital twin- and extended reality-based telepresence for collaborative robot programming in the 6G perspective,” *Digital Communications and Networks*, Oct. 2022, doi: 10.1016/j.dcan.2022.10.007.
- [39] H. Zhong, P. Li, L. Gao, and X. Li, “Low-delay admittance control of hydraulic series elastic actuator for safe human-robot collaboration,” in *Procedia Manufacturing*, Elsevier B.V., 2020, pp. 147–153. doi: 10.1016/j.promfg.2020.05.031.
- [40] M. N. Vu, F. Beck, M. Schwegel, C. Hartl-Nesic, A. Nguyen, and A. Kugi, “Machine learning-based framework for optimally solving the analytical inverse kinematics for redundant manipulators,” *Mechatronics*, vol. 91, p. 102970, May 2023, doi: 10.1016/j.mechatronics.2023.102970.
- [41] M. Safeea, P. Neto, and R. Béarée, “Model-based hardware in the loop control of collaborative robots,” in *Procedia Manufacturing*, Elsevier B.V., 2020, pp. 133–139. doi: 10.1016/j.promfg.2020.10.020.
- [42] T. Weingartshofer, B. Bischof, M. Meiringer, C. Hartl-Nesic, and A. Kugi, “Optimization-based path planning framework for industrial manufacturing processes with complex continuous paths,” *Robot Comput Integr Manuf*, vol. 82, Aug. 2023, doi: 10.1016/j.rcim.2022.102516.
- [43] M. Linsinger, J. Stecken, J. Kutschinski, and B. Kuhlenkötter, “Situational task change of lightweight robots in hybrid assembly systems,” in *Procedia CIRP*, Elsevier B.V., 2019, pp. 81–86. doi: 10.1016/j.procir.2019.03.015.
- [44] P. Gümbel, X. He, and K. Dröder, “Precision optimized pose and trajectory planning for vertically articulated robot arms,” in *Procedia CIRP*, Elsevier B.V., 2022, pp. 185–190. doi: 10.1016/j.procir.2022.02.176.
- [45] Y. Bulut and E. S. Conkur, “A real-time path-planning algorithm with extremely tight maneuvering capabilities for hyper-redundant manipulators,” *Engineering Science and Technology, an International Journal*, vol. 24, no. 1, pp. 247–258, Feb. 2021, doi: 10.1016/j.jestch.2020.07.002.
- [46] H. Maithani, J. A. C. Ramon, L. Lequievre, Y. Mezouar, and M. Alric, “Exoscarne: Assistive strategies for an industrial meat cutting system based on physical human-robot interaction,” *Applied Sciences (Switzerland)*, vol. 11, no. 9, May 2021, doi: 10.3390/app11093907.

- [47] C. Deuerlein, M. Langer, J. Seßner, P. Heß, and J. Franke, “Human-robot-interaction using cloud-based speech recognition systems,” in *Procedia CIRP*, Elsevier B.V., 2020, pp. 130–135. doi: 10.1016/j.procir.2020.05.214.
- [48] J. Chimento and R. Secoli, “Kinematic Calibration of a Seven Revolute Joints Serial Manipulator,” Master Degree, Polytechnic University, London, 2019.
- [49] M. Safeea, P. Neto, and R. Bearee, “Precise hand-guiding of redundant manipulators with null space control for in-contact obstacle navigation,” in *IEEE Industrial Electronics Society*, M. Safeea, P. Neto, and R. Bearee, Eds., Lisbon, Portugal: IEEE Industrial Electronics Society, Oct. 2019, pp. 14–17.
- [50] M. Safeea, P. Neto, and R. Béarée, “Task execution combined with in-contact obstacle navigation by exploiting torque feedback of sensitive robots,” in *Procedia Manufacturing*, Elsevier B.V., 2020, pp. 187–192. doi: 10.1016/j.promfg.2020.10.027.
- [51] S. J. Martinez, J. G. Guarnizo, and J. Avendano, “Design and implementation of a remote virtual laboratory by internet applied to kuka lbr iiwa 14 r820,” in *Lecture Notes in Electrical Engineering*, Springer, 2021, pp. 442–452. doi: 10.1007/978-3-030-53021-1_45.
- [52] T. Xu et al., “Dynamic Identification of the KUKA LBR iiwa Robot with Retrieval of Physical Parameters Using Global Optimization,” *IEEE Access*, vol. 8, pp. 108018–108031, 2020, doi: 10.1109/ACCESS.2020.3000997.
- [53] A. Shafikov, A. Sagitov, H. Li, N. Schiefermeier-Mach, and E. Magid, “Robotic palpation modeling for kuka lbr iiwa using gazebo simulator,” in *Proceedings of International Conference on Artificial Life and Robotics*, ALife Robotics Corporation Ltd, 2020, pp. 436–439. doi: 10.5954/ICAROB.2020.OS18-3.
- [54] M. Ostanin, R. Yagfarov, and A. Klimchik, “Interactive robots control using mixed reality,” in *IFAC-PapersOnLine*, Elsevier B.V., Sep. 2019, pp. 695–700. doi: 10.1016/j.ifacol.2019.11.307.
- [55] M. L. Pinto, “Developing Collaborative Applications Using the Kuka Iiwa - ProQuest,” Master degree, Instituto Politecnico de Leiria, Portugal, 2021. Accessed: Jul. 11, 2023. [Online]. Available: <https://www.proquest.com/docview/2724235186?pq-origsite=gscholar&fromopenview=true>
- [56] C. Faria, F. Ferreira, W. Erllhagen, S. Monteiro, and E. Bicho, “Position-based kinematics for 7-DoF serial manipulators with global configuration control, joint limit and singularity avoidance,” *Mech Mach Theory*, vol. 121, pp. 317–334, Mar. 2018, doi: 10.1016/J.MECHMACHTHEORY.2017.10.025.
- [57] S. Doliwa, “Calculation of the inverse kinematics for the KUKA LBR iiwa R800 (7 DOF),” *Zenodo*, Oct. 2020, doi: 10.5281/zenodo.4063575.

- [58] G. Singh and V. K. Banga, “Kinematics and trajectory planning analysis based on hybrid optimization algorithms for an industrial robotic manipulators,” *Soft comput*, vol. 26, no. 21, pp. 11339–11372, Nov. 2022, doi: 10.1007/s00500-022-07423-y.
- [59] L. Gammieri, M. Schumann, L. Pelliccia, G. Di Gironimo, and P. Klimant, “Coupling of a Redundant Manipulator with a Virtual Reality Environment to Enhance Human-robot Cooperation,” in *Procedia CIRP*, Elsevier B.V., 2017, pp. 618–623. doi: 10.1016/j.procir.2016.06.056.
- [60] “KUKA Sunrise Cabinet Assembly Instructions,” Germany, Aug. 2015. Accessed: Jul. 27, 2023. [Online]. Available: https://www.oir.caltech.edu/twiki_oir/pub/Palomar/ZTF/KUKARoboticArmMaterial/MA_KUKA_Sunrise_Cabinet_en.pdf
- [61] “SCHUNK Hand in hand for tomorrow.” <https://schunk.com/de/en/gripping-systems/parallel-gripper/co-act-egp-c/co-act-egp-c-64-n-n-ktoc/p/0000000000001321172> (accessed Aug. 03, 2023).
- [62] P. Tucan, G. Carbone, and D. Pîslă, “A STUDY REGARDING USE OF KUKA LBR IIWA 7 R800 ROBOTIC ARM IN PERFORMING POST-STROKE REHABILITATION TASKS,” *INSTITUTULUI POLITEHNIC DIN IAȘI*, vol. 64, no. 68, Jun. 2018.

APPENDIX A

The following program represents the complete Java code that was written to construct the User-Friendly Platform (UFPI) interface:

```
public class A_UFPI_Interface extends RoboticsAPIApplication {
    private Controller kuka_Sunrise_Cabinet_1;
    private ISmartServoLINRuntime impedanceMode = null;
    private LBR lbr_iiwa_7_R800;
    private static final int stiffnessZ = 400; //in [N/m]
    private static final int stiffnessY = 400; //in [N/m]
    private static final int stiffnessX = 400; //in [N/m]
    private static final double[] MAX_TRANSLATION_VELOCITY = { 150,
150, 150 };
    private MediaFlangeIOGroup mFlangeTouch;
    private HRCEGP40 gripper;
    private final static String informationText =
        "Recording motion application for move to several
point as you need." + "\n" +
        "\n" + "Note!" + "\n" + "Please enable
automatic mode and then run this program.";

    private final static String informationTextax =
        "Because you select a circular motion type; So at
the first, go to the auxiliary point: ";

    private final static String informationTextde =
        "Now go to the destination point:" + "\n" +
        "\n" + "Note!" + "\n" + "The point must
be on a circular path, otherwise an error will occur when executing.";

    private final static String InformationText =
        "Do you want to move? If yes, choose the type of
motion for this point, else click stop: ";

    private final static String InformationTextg =
        "Do you want to Open or Closed the Gripper? ";

    private final static String Informationtext =
        "Do you want to re-program the recorded movements or
terminate the program? ";

    private final static String informationtext =
        "Do you want to save this position? ";

    private final static String InformationTextRe =
        "Enter the number of re-executions for the Recorded
movements from 1-12 times: ";

    private final static String InformationTextch =
        "Do you want to get to the point with the Hand
Guiding Motion feature, or by adding a value to one of the " +
        "previous point axes (x, y, z), or
selecting one of the previously designated points as reference
points? ";

    private final static String InformationTextch2 =
        "To control the robot using the Jog keys, you need
in the first to transition from automatic mode to T1 mode. " +
```

"Subsequently, move the robot to your desired location, after that switch back to automatic mode and finally press the 'OK' button and run the interface to complete the program.";

```
final static String InformationTextad =
    "Select the axis that you want to add the value:";

final static String InformationTextva =
    "Enter the value of the offset:"+"\n" +
    "\n" + "Note!"+" \n" + "The value must
be within the workspace of the robot.";
```

```
final static String InformationTextsi =
    "Enter the value sign:";

private double x = 0.05;
private double y = 0.5;
private int i = 0;
private static int sampleIntervall=0;
private JointPosition myJoints;
private int[] myIndex = new int[2000000];
private int[] mySave = new int[2000000];
private String[] mycr = new String[2000000];
private static int n = 0;
private static int cr = 0;
private static String[] TypeMO = new String[2000000];
private Frame[] myPrintFrame = new Frame[2000000];
private Frame[] auxiliaryFrame = new Frame[2000000];
private ConditionObserver observing;
private boolean BClash;
private double MaxClashForce = 7.5;
private double MinClashForce = -7.5;
private int count = 0;
private HandGuidingMotion Hand;
JointPosition Home = new JointPosition(0,0,0,-
Math.toRadians(90),0,Math.toRadians(90),0);
```

```
static String InformationTextPo =
    "Please input the number of the previously
designated reference point:"+"\n" +
    "\n" + "Note!"+" \n" + "If you input a
number for a reference point that hasn't been established previously,
the robot will move " +
    "to the last stored reference point.
It's important to note that there are "+ n + " saved reference
points.";
```

```
static String informationtextsa =
    "Want to save this point as reference point number
"+ (n+1) + " ?";
```

```
@Override
public void initialize() {
    kuka_Sunrise_Cabinet_1 =
getController("KUKA_Sunrise_Cabinet_1");
    lbr_iiwa_7_R800 = (LBR) getDevice(kuka_Sunrise_Cabinet_1,
        "LBR_iiwa_7_R800_1");
    mFlangeTouch = new
MediaFlangeIOGroup(kuka_Sunrise_Cabinet_1);
    gripper = new HRCEGP40(mFlangeTouch);
    this.setCollisionDetection();
```



```

        Hand = new HandGuidingMotion();
    }

    public void setCollisionDetection(){
        BClash = true;
        JointTorqueCondition jt1 = new
JointTorqueCondition(lbr_iiwa_7_R800, JointEnum.J1, MinClashForce, MaxCla
shForce);
        JointTorqueCondition jt2 = new
JointTorqueCondition(lbr_iiwa_7_R800, JointEnum.J2, MinClashForce, MaxCla
shForce);
        JointTorqueCondition jt3 = new
JointTorqueCondition(lbr_iiwa_7_R800, JointEnum.J3, MinClashForce, MaxCla
shForce);
        JointTorqueCondition jt4 = new
JointTorqueCondition(lbr_iiwa_7_R800, JointEnum.J4, MinClashForce, MaxCla
shForce);
        JointTorqueCondition jt5 = new
JointTorqueCondition(lbr_iiwa_7_R800, JointEnum.J5, MinClashForce, MaxCla
shForce);
        JointTorqueCondition jt6 = new
JointTorqueCondition(lbr_iiwa_7_R800, JointEnum.J6, MinClashForce, MaxCla
shForce);
        JointTorqueCondition jt7 = new
JointTorqueCondition(lbr_iiwa_7_R800, JointEnum.J7, MinClashForce, MaxCla
shForce);
        ICondition collisionCondition = jt1.or(jt2, jt3, jt4, jt5,
jt6, jt7);

        observing=getObserverManager().createConditionObserver(collision
Condition, NotificationType.OnEnable, new IAnyEdgeListener()
        {
            public void onAnyEdge (ConditionObserver
conditionObserver, Date time, int missedEvents, boolean
conditionValue){
                if(conditionValue){
                    getLogger().info("force exceed");

                    if(BClash){

getLogger().info("Collision");

getApplicationControl().setApplicationOverride(0.0d);

mFlangeTouch.setLEDBLue(false);
getLogger().info("Impact the
robot with an external force for only once");
ThreadUtil.sleep(1000);
count = 0;
BClash = !BClash;
                    }
                    else if(count == 1){

mFlangeTouch.setLEDBLue(true);
getLogger().info("Wait a
little bit");
ThreadUtil.sleep(1000);

getApplicationControl().setApplicationOverride(1.0d);
BClash = !BClash;
                }
            }
        }
    }

```

```

        }
        count++;}}
    } );

    @Override
    public void run() {
        Hand.setJointVelocityLimit(1.5); //rads/s
        Hand.setCartVelocityLimit(50); //cm/s
        sampleIntervall=0;
        n = 0;
        cr = 0;

        CartesianImpedanceControlMode impedanceControlMode = new
        CartesianImpedanceControlMode();

        impedanceControlMode.parametrize(CartDOF.X).setStiffness(stiffne
ssX);

        impedanceControlMode.parametrize(CartDOF.Y).setStiffness(stiffne
ssY);

        impedanceControlMode.parametrize(CartDOF.Z).setStiffness(stiffne
ssZ);

        impedanceControlMode.parametrize(CartDOF.ROT).setStiffness(50);
//in[Nm/rad].

        int isCancel =
getApplicationUI().displayModalDialog(ApplicationDialogType.INFORMATIO
N, informationText, "OK", "Cancel");
        if (isCancel == 1)
        {
            return;
        }

        PTP ptpToHomePosition = ptp(Home);

        ptpToHomePosition.setJointVelocityRel(y).setJointAccelerationRel
(x);

        lbr_iiwa_7_R800.move(ptpToHomePosition);
        gripper.setOpen();

        if
(!ServoMotion.validateForImpedanceMode(lbr_iiwa_7_R800))
        {
            getLogger().
            .info("Validation of torque model failed - correct
your mass property settings");
            getLogger().
            .info("Servo motion will be available for position
controlled mode only, until validation is performed");
        }

        int run =2;
        while(run == 2)
        {
            if (i > 0)
            {
                getLogger().info("Enter the positions that to
be added:");
            }
        }
    }
}

```

```

        int ret =0;
        while(ret != 4)
        {
            ret =
getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION, I
nformationText, "MovePTP", "MovePTP_With_Impedance", "MoveLIN", "MoveCIRC"
, "Stop");

            switch (ret)
            {
                case 0:
                    TypeMO[sampleIntervall] = "PTP";
                    RecordMotion();
                    break;
                case 1:
                    TypeMO[sampleIntervall] =
"MovePTP_With_Impedance";
                    RecordMotion();
                    break;
                case 2:
                    TypeMO[sampleIntervall] = "LIN";
                    RecordMotion();
                    break;
                case 3:
                    TypeMO[sampleIntervall] = "CIRC";
                    RecordMotionCIRC();
                    break;
                case 4:
                    getLogger().info("Move away from the robot and
reduce speed if it is high.");
                    break;
            }
        }
        observing.enable();
        run =1;
        while(run != 0 && run != 2)
        {
            int Re
=getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION,
InformationTextRe, "Once", "Twice", "3-Times", "4-Times", "5-Times", "6-
Times", "7-Times", "8-Times", "9-Times", "10-Times", "11-Times", "12-
Times");

            for(int c=0; c <= Re; c++)
            {
                mFlangeTouch.setLEDBLue(true);
                getLogger().info("Go to Home Position");
                getLogger().info("Execution No. " + (1+c));
                lbr_iiwa_7_R800.move(ptpToHomePosition);
                gripper.setOpen();
                ThreadUtil.millisSleep(1000);

                for(i=0; i<sampleIntervall; i++)
                {
                    if(TypeMO[i] == "PTP"){
                        getLogger().info("Go to point number " + (1+i)
+ " by using PTP Motion.");

                        lbr_iiwa_7_R800.move(ptp(myPrintFrame[i]).setJointVelocityRel(y)
.setJointAccelerationRel(x));
                    }
                    else if (TypeMO[i] == "MovePTP_With_Impedance") {

```

```

        getLogger().info("Go to point number " + (1+i)
+ " by using Impedance Motion Mode.");
        observing.disable();
        runSmartServoLINMotion(impedanceControlMode);

    }
    else if (TypeMO[i] == "LIN") {
        getLogger().info("Go to point number " + (1+i)
+ " by using LIN Motion.");

        lbr_iowa_7_R800.move(lin(myPrintFrame[i]).setJointVelocityRel(y)
.setJointAccelerationRel(x));
    }
    else{
        getLogger().info("Go to point number " + (1+i)
+ " by using CIRC Motion.");
        lbr_iowa_7_R800.move(circ(auxiliaryFrame[i],
myPrintFrame[i]).setJointVelocityRel(y).setJointAccelerationRel(x));
    }

    if(myIndex[i] != i) {
        if(gripper.getStatus()[0]) {
            ThreadUtil.sleep(500);
            gripper.setClose();
            getLogger().info("Gripper is close.");
            ThreadUtil.sleep(500);
        }
        else{
            ThreadUtil.sleep(500);
            gripper.setOpen();
            getLogger().info("Gripper is open.");
            ThreadUtil.sleep(500);
        }
    }
    observing.enable();
}

getLogger().info("Finished");
mFlangeTouch.setLEDBLue(false);
run =
getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION, I
nformationText, "Terminate", "Re-program", "Re-
program_with_adding_points");
}
}
mFlangeTouch.setLEDBLue(true);
getLogger().info("Back to Home Position");
lbr_iowa_7_R800.move(ptpToHomePosition);
gripper.setOpen();
for(int c=0; c<=10; c++)
{
    mFlangeTouch.setLEDBLue(true);
    ThreadUtil.sleep(250);
    mFlangeTouch.setLEDBLue(false);
    ThreadUtil.sleep(250);
}

private void RecordMotion() {
    getLogger().info("Go to point number "
+ (1+sampleIntervall) + " :");

```

```

ApproachToReach();

mFlangeTouch.setLEDBLue(true);
myIndex[sampleIntervall] = sampleIntervall;
int grip =
getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION, I
nformationTextg, "No", "Yes");
switch (grip)
{
case 0:
    break;
case 1:
    myIndex[sampleIntervall] = sampleIntervall+1;
    if(gripper.getStatus()[0]){
        gripper.setClose();
        getLogger().info("Gripper is close.");
    }
    else{
        gripper.setOpen();
        getLogger().info("Gripper is open.");
    }
    break;
}
mFlangeTouch.setLEDBLue(false);

myJoints = lbr_iiwa_7_R800.getCurrentJointPosition();

myPrintFrame[sampleIntervall] =
lbr_iiwa_7_R800.getForwardKinematic(myJoints).copyWithRedundancy();
sampleIntervall++;
}

private void RecordMotionCIRC() {
    getLogger().info("Go to point number "
+(1+sampleIntervall) + " :");
    // getLogger().info("Because you select a circular motion
type; So at the first, go to the auxiliary point:");

    getApplicationUI().displayModalDialog(ApplicationDialogType.INFO
RMATION, informationTextax, "OK");
    cr =1;
    ApproachToReach();

    myJoints = lbr_iiwa_7_R800.getCurrentJointPosition();

    auxiliaryFrame[sampleIntervall] =
lbr_iiwa_7_R800.getForwardKinematic(myJoints).copyWithRedundancy();
    cr = 2;
    // getLogger().info("Now go to the destination point:");

    getApplicationUI().displayModalDialog(ApplicationDialogType.INFO
RMATION, informationTextde, "OK");
    ApproachToReach();

    cr = 0;
    mFlangeTouch.setLEDBLue(true);
    myIndex[sampleIntervall] = sampleIntervall;
    int grip =
getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION, I
nformationTextg, "No", "Yes");
    switch (grip)

```

```

{
    case 0:
        break;
    case 1:
        myIndex[sampleIntervall] = sampleIntervall+1;
        if(gripper.getStatus()[0]){
            gripper.setClose();
            getLogger().info("Gripper is close.");
        }
        else{
            gripper.setOpen();
            getLogger().info("Gripper is open.");
        }
        break;
}
mFlangeTouch.setLEDBLue(false);

myJoints = lbr_iiwa_7_R800.getCurrentJointPosition();

myPrintFrame[sampleIntervall] =
lbr_iiwa_7_R800.getForwardKinematic(myJoints).copyWithRedundancy();
sampleIntervall++;
}

protected void runSmartServoLINMotion(final IMotionControlMode
controlMode)
{
    getLogger().info("Hold position in impedance control
mode");
    AbstractFrame initialPosition =
lbr_iiwa_7_R800.getCurrentCartesianPosition(lbr_iiwa_7_R800.getFlange(
));
    // Create a new smart servo linear motion
    SmartServoLIN SmartMotion = new
SmartServoLIN(initialPosition);

    SmartMotion.setMaxTranslationVelocity(MAX_TRANSLATION_VELOCITY);
    SmartMotion.setMinimumTrajectoryExecutionTime(20e-3);

    //getLogger().info("Starting the SmartServoLIN in " +
controlMode);

    lbr_iiwa_7_R800.moveAsync(SmartMotion.setMode(controlMode));

    //getLogger().info("Get the runtime of the SmartServoLIN
motion");
    impedanceMode = SmartMotion.getRuntime();

    ThreadUtil.sleep(500);
    // Update the smart servo LIN runtime
    //
theSmartServoLINRuntime.updateWithRealtimeSystem();
    // Set new destination
    impedanceMode.setDestination(myPrintFrame[i]);
    ThreadUtil.sleep(6000);
    impedanceMode.setDestination(myPrintFrame[i]);
    getLogger().info("Turn off the impedance control
mode");

    impedanceMode.stopMotion();
}

```



```

        int ad
=getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION,
InformationTextad,"X_Axis","Y_Axes","Z_Axis");
        if (cr == 2){
            myPrintFrame[sampleIntervall] =
auxiliaryFrame[sampleIntervall].copyWithRedundancy();
        }
        else{
            myPrintFrame[sampleIntervall] =
myPrintFrame[sampleIntervall-1].copyWithRedundancy();
        }
        int va
=getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION,
InformationTextva,"1mm","3mm","5mm","10mm","20mm","30mm","40mm","50mm"
,"100mm","200mm","300mm","500mm");
        switch (va)
        {
        case 0:
            va=1;
            break;
        case 1:
            va=3;
            break;
        case 2:
            va=5;
            break;
        case 3:
            va=10;
            break;
        case 4:
            va=20;
            break;
        case 5:
            va=30;
            break;
        case 6:
            va=40;
            break;
        case 7:
            va=50;
            break;
        case 8:
            va=100;
            break;
        case 9:
            va=200;
            break;
        case 10:
            va=300;
            break;
        case 11:
            va=500;
            break;
        }
        int si
=getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION,
InformationTextsi,"(-)","(+)");
        switch (si)
        {
        case 0:
            va=-va;

```



```

        break;
    case 1:
        break;
    }
    switch (ad)
    {
    case 0:
        getLogger().info("Go to this point
by adding value on X Axis.");

        myPrintFrame[sampleIntervall].setX(myPrintFrame[sampleIntervall]
.getX()+ va);

        lbr_iiwa_7_R800.move(lin(myPrintFrame[sampleIntervall])).setJoint
VelocityRel(y).setJointAccelerationRel(x));
        break;
    case 1:
        getLogger().info("Go to this point
by adding value on Y Axis.");

        myPrintFrame[sampleIntervall].setY(myPrintFrame[sampleIntervall]
.getY()+ va);

        lbr_iiwa_7_R800.move(lin(myPrintFrame[sampleIntervall])).setJoint
VelocityRel(y).setJointAccelerationRel(x));
        break;
    case 2:
        getLogger().info("Go to this point
by adding value on Z Axis.");

        myPrintFrame[sampleIntervall].setZ(myPrintFrame[sampleIntervall]
.getZ()+ va);

        lbr_iiwa_7_R800.move(lin(myPrintFrame[sampleIntervall])).setJoint
VelocityRel(y).setJointAccelerationRel(x));
        break;
    }

    break;
    case 3:
        int Po
=getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION,
InformationTextPo,"1","2","3","4","5","6","7","8","9","10","11","12");
        if(Po >= n){
            Po = n;
        }
        getLogger().info("Go to this point as
point number "+(1+Po)+ " that previously saved.");
        if (mycr[Po] == "auxiliary"){

            lbr_iiwa_7_R800.move(lin(auxiliaryFrame[mySave[Po]]).setJointVel
ocityRel(y).setJointAccelerationRel(x));
        }
        else{

            lbr_iiwa_7_R800.move(lin(myPrintFrame[mySave[Po]]).setJointVeloc
ityRel(y).setJointAccelerationRel(x));
        }
        break;
    }
}

```

```

        sav =
getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION, i
nformationtext, "No", "Yes");
    }
}
else{
    int sav1 =0;
    while(sav1 != 1)
    {
        int ch2
=getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION, i
nformationTextch, "Hand Guiding", "Manual Keys");
        switch (ch2)
        {
            case 0:
                getLogger().info("Go to this
point by using hand guiding.");
                lbr_iiwa_7_R800.move(Hand);
                break;
            case 1:
                getLogger().info("Go to this
point by using manual keys.");

                getApplicationUI().displayModalDialog(ApplicationDialogType.INFO
RMATION, InformationTextch2, "OK");

                myJoints =
lbr_iiwa_7_R800.getCurrentJointPosition();

                myPrintFrame[sampleIntervall] =
lbr_iiwa_7_R800.getForwardKinematic(myJoints).copyWithRedundancy();

                lbr_iiwa_7_R800.move(lin(myPrintFrame[sampleIntervall])).setJoint
VelocityRel(y).setJointAccelerationRel(x);
                break;
        }
        sav1 =
getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION, i
nformationtext, "No", "Yes");
    }
}
if (n < 12 && ch != 3){
    int sav2 =
getApplicationUI().displayModalDialog(ApplicationDialogType.QUESTION, i
nformationtextsa, "No", "Yes");
    if (sav2 == 1)
    {
        if (cr == 1){
            mycr[n] = "auxiliary";
        }
        else{
            mycr[n] = "Print";
        }
        mySave[n] = sampleIntervall;
        n++;
    }
}
}
}

```

APPENDIX B

The following steps outline the process of setting up a Kuka LBR iiwa device and performing Safety Configuration for this platform:

1. Open Kuka sunrise workbench.
2. Create new project, as shown in Figure B.1:

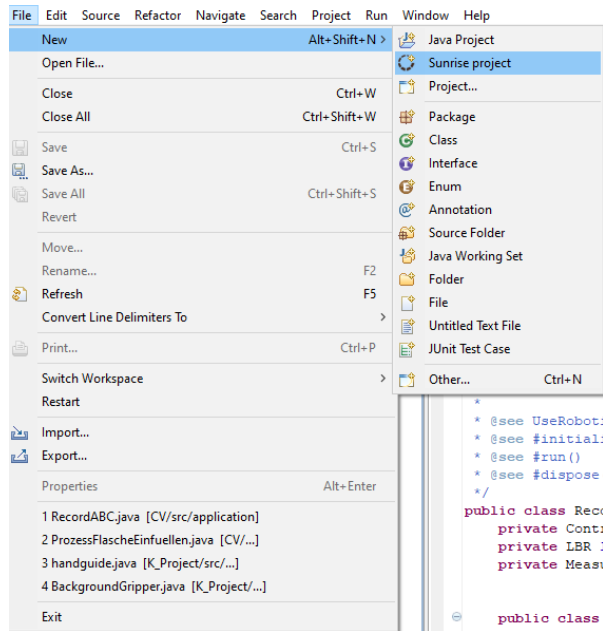


Figure B.1 Create new project

3. Assign IP-address of the robot and then click next, as shown in Figure B.2:

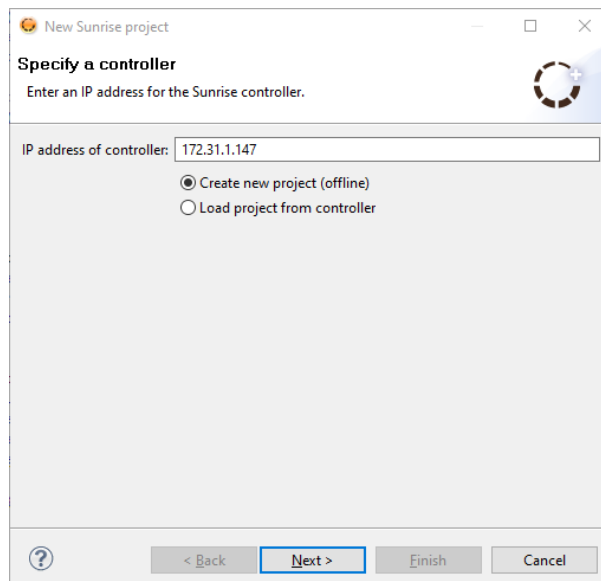


Figure B.2 Assign IP-address of the robot

4. Select LBR iiwa 7 R800, as shown in Figure B.3:

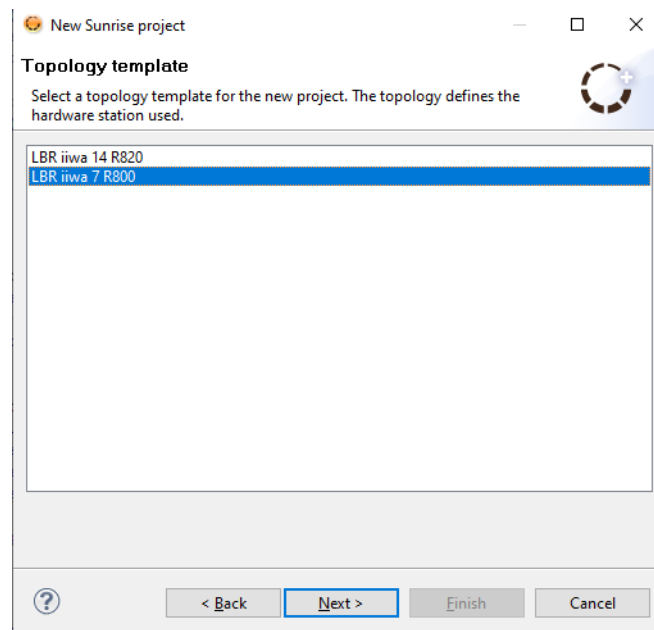


Figure B.3 Select LBR iiwa 7 R800

5. Select Electric gripper, as shown in Figure B.4:

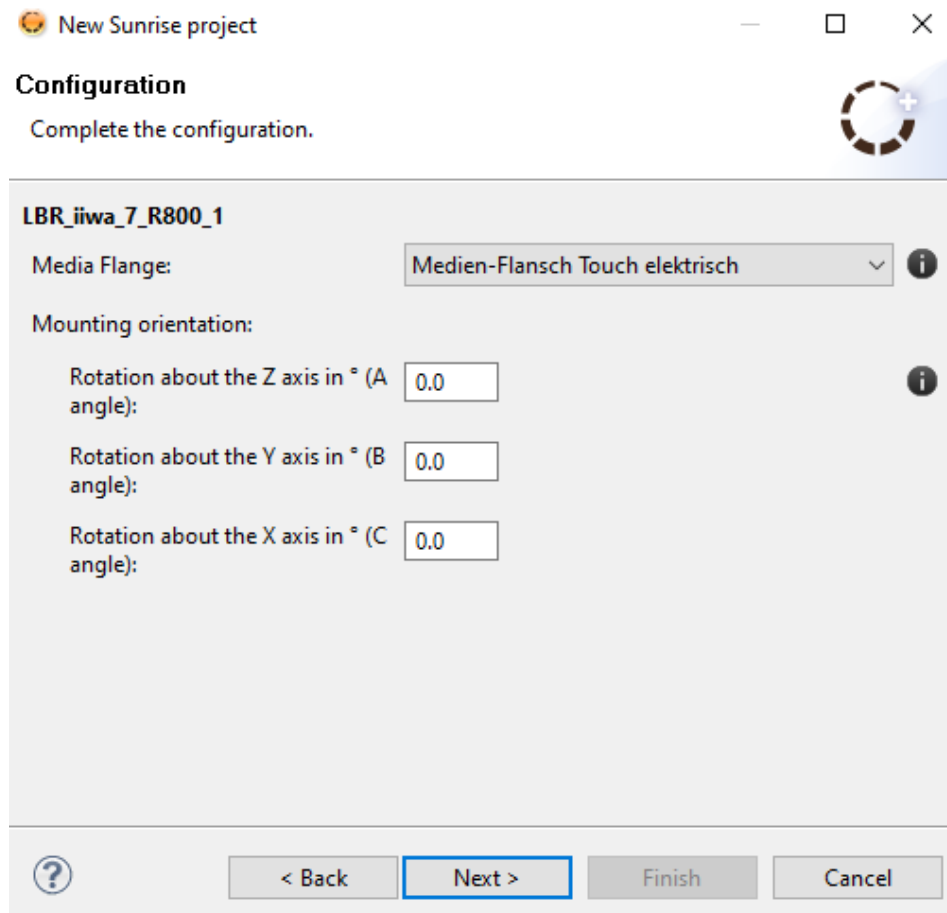


Figure B.4 Select Medien-Flansch Touch elektrisch

6. Click on Safety configuration, as shown in Figure B.5:

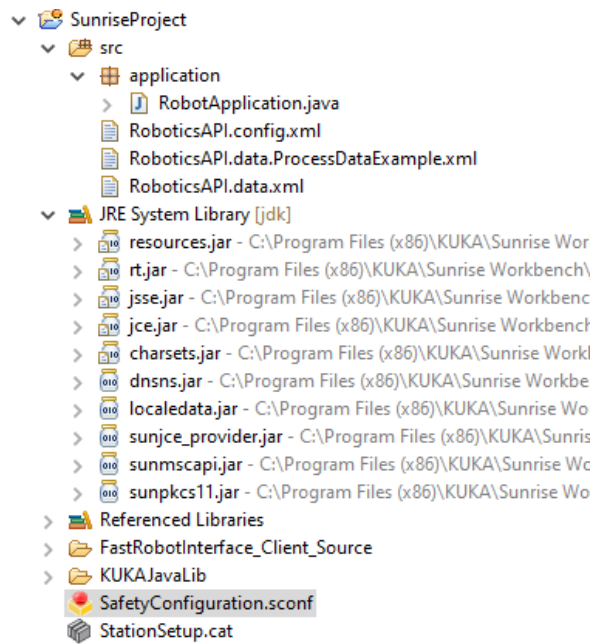


Figure B.5 Safety configuration

7. Setup customer PSM safety parameter and KUKA PSM, as shown in Figure B.6 and Figure B.7 respectively:

Configurable customer safety configuration (6/100)						
Row	Active	Category	AMF 1	AMF 2	AMF 3	Reaction
1	<input checked="" type="checkbox"/>	External EMERGENCY STOP	Input signal (1) Input CIB_SR.1	-	-	Stop 1 (path-maintaining)
2	<input checked="" type="checkbox"/>	Operator safety	Input signal (2) Input CIB_SR.2	Time delay (1) Delay time : 0ms	-	Stop 1 (path-maintaining)
3	<input checked="" type="checkbox"/>	Safety stop	Input signal (3) Input CIB_SR.3	-	-	Stop 1 (path-maintaining)
4	<input checked="" type="checkbox"/>	Collision detection	Collision detection (2) First kinematic system Maximum external torque : 15 Nm	-	-	Stop 1
5	<input checked="" type="checkbox"/>	Operator safety	Automatic mode	Torque referencing (1) First kinematic system	Position referencing (1) First kinematic system	Output CIB_SR.12
6	<input checked="" type="checkbox"/>	Operator safety	High-velocity mode	Torque referencing (2) First kinematic system	Position referencing (2) First kinematic system	Output CIB_SR.13

Selection	Instance	Used in (table)row	Parameter	Value	Info
	-		-		No cell selected for which information is available.

Tool selection table | KUKA PSM | Customer PSM | 61%

Figure B.6 Customer PSM safety parameter

Non-configurable KUKA safety configuration							(3/100)			
Row	Active	Category	AMF 1	AMF 2	AMF 3	Reaction				
1		Local EMERGENCY STOP	-	smartPAD Emergency Stop	-	Stop 1 (path-maintaining)				
2		Enabling device	Hand guiding device enabling inactive	Test mode	smartPAD enabling switch inactive	Stop 1 (path-maintaining)				
3		Velocity monitoring	Hand guiding device enabling active	Cartesian velocity monitoring (100) First kinematic system	-	Stop 1 (path-maintaining)				

Selection	Instance	Used in (table)row	Parameter	Value	Info
	-		-		No cell selected for which information is available.

Figure B.7 KUKA PSM safety parameter

- Click on StationSetup, as shown in Figure B.8:

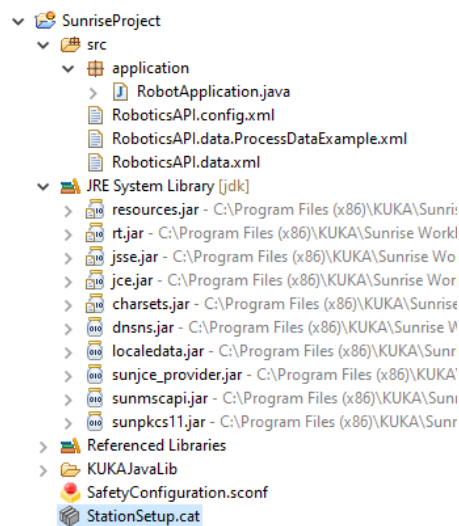


Figure B.8 StationSetup

9. Select software tab and check this follow element, as shown in Figure B.9:





















Elements	Install	Mode	Selected version	Currently installed version
▼  KUKA_Sunrise_Cabinet_1 (Version: 1)				
 Direct Servo Motion Example Applications	<input checked="" type="checkbox"/>	Automatic	<input type="radio"/> [not selected]	[not installed]
 Direct Servo Motion Extension	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 1.15.0.Con1-16-0_4	[not installed]
 Fast Robot Interface Extension Example Applications	<input checked="" type="checkbox"/>	Automatic	<input type="radio"/> [not selected]	[not installed]
 Fast Robot Interface Extension	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 1.15.1.Con1-16-0_4	[not installed]
 Smart Servo Motion Example Applications	<input checked="" type="checkbox"/>	Automatic	<input type="radio"/> [not selected]	[not installed]
 Smart Servo Motion Extension	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 1.16.0.Con1-16-0_4	[not installed]
 Smart Servo Linear Motion Example Applications	<input checked="" type="checkbox"/>	Automatic	<input type="radio"/> [not selected]	[not installed]
 Smart Servo Linear Motion Extension	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 1.15.0.Con1-16-0_4	[not installed]
 FSoE Slave support	<input checked="" type="checkbox"/>	Automatic	<input type="radio"/> 1	[not installed]
 KUKA Sunrise.GripperToolbox	<input type="checkbox"/>	Automatic	<input type="radio"/> [not selected]	[not installed]
 Handguiding Support	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 1	[not installed]
 Ikarus AntiVirus	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 2.7.30	[not installed]
 KUKA Sunrise.PROFINET® D	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 1.3.1	[not installed]
 KUKA Sunrise.PROFINET® M/S	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 1.3.1	[not installed]
 Robotics API	<input checked="" type="checkbox"/>	Manual	<input checked="" type="radio"/> 1.0.0	[not installed]
 Human Robot Collaboration	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 1	[not installed]
 smartHMI Language Pack	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 1.10.2	[not installed]
 KUKA Sunrise.StatusController	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 1.1.0	[not installed]
 Sunrise Data Recording	<input checked="" type="checkbox"/>	Automatic	<input checked="" type="radio"/> 1	[not installed]

Figure B.9 Software Tab

APPENDIX C

The provided Java code below illustrates the program that utilized for configuring the User keys within the background tasks:

```

public class BackgroundGripper extends RoboticsAPICyclicBackgroundTask {

    @Inject
    private LBR lbr_iiwa_7_R800;

    private UserKeyLED KeyLED = UserKeyLED.Yellow;
    private Controller kuka_Sunrise_Cabinet_1;
    private MediaFlangeIOGroup mFlangeTouch;
    private HRCEGP40 gripper;
    private String value = " ";
    JointPosition Home = new JointPosition(0,0,0,-Math.toRadians(90),0,0,0);
    PTP ptpToHomePosition = ptp(Home);

    public void initialize() {
        kuka_Sunrise_Cabinet_1 = getController("KUKA_Sunrise_Cabinet_1");
        initializeCyclic(0, 50, TimeUnit.MILLISECONDS,
            CycleBehavior.BestEffort);

        mFlangeTouch = new MediaFlangeIOGroup(kuka_Sunrise_Cabinet_1);
        gripper = new HRCEGP40(mFlangeTouch);
        ptpToHomePosition.setJointVelocityRel(0.25);

        IUserKeyBar gripperBar = getApplicationUI().createUserKeyBar("Gripper");
        IUserKeyListener listener1 = new IUserKeyListener() {
            @Override
            public void onKeyEvent(IUserKey key, UserKeyEvent event) {
                if (event == UserKeyEvent.FirstKeyDown) {
                    gripper.setClose();
                    value = "Close";
                }
                else if (event == UserKeyEvent.SecondKeyDown) {
                    gripper.setOpen();
                    value = "Open";
                }
                key.setText(UserKeyAlignment.Middle, value);
            }
        };

        final IUserKey GripperKey1 = gripperBar.addDoubleUserKey(0, listener1,
false);

        GripperKey1.setText(UserKeyAlignment.TopMiddle, "grip");
        GripperKey1.setText(UserKeyAlignment.Middle, value);
        GripperKey1.setText(UserKeyAlignment.BottomMiddle, "release");

        IUserKeyListener listener2 = new IUserKeyListener() {
            @Override
            public void onKeyEvent(IUserKey key,
UserKeyEvent event) {
                if (event ==
UserKeyEvent.FirstKeyDown) {

```



```

        if(gripper.getStatus()[0]){
            gripper.setClose();
            value = "Close";
        }
        else{
            gripper.setOpen();
            value = "Open";
        }
    }
    else if (event == UserKeyEvent.SecondKeyDown) {

        if(mFlangeTouch.getLEDBlue()== true){

            mFlangeTouch.setLEDBlue(false);

            UserKeyLED.Red;

            KeyLED =

        }
        else{

            mFlangeTouch.setLEDBlue(true);

            UserKeyLED.Green;

            KeyLED =

        }
    }

    GripperKey1.setText(UserKeyAlignment.Middle,value);
    key.setText(UserKeyAlignment.Middle, "Omar");

    key.setLED(UserKeyAlignment.BottomRight, KeyLED,UserKeyLEDSize.Normal);
    }
    };

    IUserKey GripperKey2 =
    gripperBar.addDoubleUserKey(2, listener2, true);
    GripperKey2.setText(UserKeyAlignment.TopMiddle,
    "obverse");

    GripperKey2.setText(UserKeyAlignment.Middle,"Omar");
    GripperKey2.setText(UserKeyAlignment.BottomLeft,
    "LED");

    GripperKey2.setLED(UserKeyAlignment.BottomRight,
    KeyLED,UserKeyLEDSize.Normal);

    gripperBar.publish();
}

@Override
public void runCyclic() {
    if(mFlangeTouch.getUserButton()){
        getLogger().info("Move to home position");
        lbr_iiwa_7_R800.move(ptpToHomePosition);
    }
}
}

```

APPENDIX D

The Figure D.1 displays an image of the KUKA LBR iiwa 7 R800 robot, which served as the subject of study and implementation for this project:



Figure D.1 KUKA LBR IIWA 7 R800 robot in the university laboratory

ZUSAMMENFASSUNG

Diese Forschungsarbeit präsentiert eine umfassende Studie, die sich auf die Gestaltung und Umsetzung einer benutzerfreundlichen Plattform-Schnittstelle (UFPI) für die Echtzeitsteuerung des KUKA LBR iiwa 7 R800-Roboters konzentriert. Ziel der Studie ist es, die Interaktion und Zusammenarbeit zwischen Menschen und Robotern in industriellen Umgebungen zu verbessern, insbesondere bei manuellen Montageaufgaben. Die Forschung legt den Schwerpunkt auf die Vereinfachung komplexer Programmierung, die Gewährleistung von Sicherheit, die Optimierung von Trajektorien, die Verbesserung der Genauigkeit und die Erleichterung nahtloser Systemintegration. Diese Studie verwendet einen umfassenden Ansatz bei der Sammlung und Analyse qualitativer Daten. Basierend auf diesen Daten entwickelt die Forschung eine auf Java basierende Schnittstelle auf dem smartPAD-Bedienfeld unter Verwendung der Sunrise Workbench-Software, die eine intuitive Echtzeitsteuerung und Überwachung der Bewegungen des Roboters ermöglicht. Die Forschung unterstreicht die Bedeutung dieser Weiterentwicklung zur Förderung effizienter und sicherer industrieller Abläufe und skizziert gleichzeitig zukünftige Forschungsrichtungen sowie das Potenzial für eine breitere industrielle Anwendung.