Jacek Dąbrowski

Interface Design for Physical Human-Robot Interaction using Sensorless Force Control Method

School of Electrical Engineering

Espoo, 24.10.2013

Project supervisor:

Prof. Ville Kyrki

Instructor:

PhD. Polychronis Kondaxakis

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
FINAL PROJECT

Author: Jacek Dąbrowski

Title: Interface Design for Physical Human-Robot Interaction using sensorless control method

Date: 22.10.2013 Language: English Number of pages: 6 + 73

Degree programme: Electrical Engineering

Project supervisor: Prof. Ville Kyrki

Advisor: PhD. Polychronis Kondaxakis

The rapid increase in the usage of robots has made interaction between a human and a robot a crucial field of research. Physical human–robot interaction constitutes a relevant and growing research area. Nowadays robots are used in almost all areas of life, such as in households, for education and in medicine. Therefore, many research studies are being conducted on ergonomic human–robot interfaces enabling people to communicate, collaborate and to teach a robot through physical interaction.

This thesis is focused on developing a physical human-robot interface by means of which the user is able to control a walking humanoid by exerting force. Through physical contact with the robot arm, a human can influence the direction and velocity of the robot walk. In other words, the user leads the humanoid by the hand, and the robot compensates this external force by following the user.

The developed interface offers a method of sensorless force control. Instead of the traditional approach using force/torque measurement, the fact that a DC motor's torque is proportional to the armature current was applied. Two different control algorithms were implemented and compared. Consequently, a usability test was conducted for different interfaces to find the one which was the most ergonomic.

Keywords: Force control, Human-robot interaction, Interface design, Robotics, System identification.

Content

Abbreviations and Symbols

Symbols

F	Force producing rotation
r	Lever-arm
sgn	Sign function
t	Time
T	Torque generated by the motor
W	The Wilcoxon test statistic
x	Cross product
$Z_M(s)$	Laplace transfer function

Abbreviations

ADC	Analog-To-Digital
API	Application Programming Interface
BDD	Block Definition Diagram
CPU	Central Processing Unit
DC	Direct current
DCM	Device Control Manager
DOF	Degrees of freedom
EMF	Electromotive force
FOH	The First-Order Hold
HRI	Human-Robot Interaction
IP	Internet Protocol
LbD	Learning by Demonstration
MRE	Magnetic Rotary Encoders
OS	Operating System
PEM	Prediction Error Minimization
PI	Proportional-Integral
PID	Proportional-Integral-Derivative
RAM	Random Access Memory
pHRI	physical Human-Robot Interaction
rpm	Revolutions per minute
SDK	Software Development Kit
SISO	Single Input Single Output
SysML	System Modeling Language
TCP	Transmission Control Protocol

UML	Unified Modeling Language
USB	Universal Serial Bus
Wi-Fi	Wireless Fidelity
3D	Three dimensional space

1. Introduction

This section presents the background of the conducted research as well as the objectives and restrictions of the investigation. In addition, the structure of the thesis is described and a brief description of each chapter is provided.

1.1 Background

Robots are no longer limited to working in well-defined, static environments. Nowadays robots are utilised in almost all areas of life, such as in household, educational or medical applications. This means that this new generation of robots has to be able to efficiently handle interactions with humans or other robots. In general, both the subject and problem are formulated as human-robot interaction.

Today's rapid increase in robotic applications demands a more in-depth study in the field of human-robot interaction. Therefore, physical human-robot interaction has become a relevant and growing research area, with investigations being conducted on ergonomic human-robot interfaces enabling people to communicate, collaborate and teach robotic artefacts by physical interaction. This imposes new challenges to create interfaces that will be characterised by efficiency, accuracy or easiness of usage. Also, appropriate methods are necessary in order to verify and evaluate the above-mentioned features.

This thesis is focused on developing a physical human-robot interface by means of which a user will be able to control a walking humanoid by exerting force on its arm. Through physical contact with the robot's arm a human operator will control the direction and velocity of the robot's walk. In other words, the user will lead the humanoid by the hand, and the robot will compensate external force stimuli by following that user.

Moreover, the developed interface is based on sensorless force-control. Instead of traditional approaches applying force/torque sensors, we have adopted a simplified approach where a DC motor's torque is assumed to be proportional to its armature current.

Several different control algorithms will be implemented and compared. Consequently, we will conduct usability tests for different interfaces.

1.2 Objectives and restrictions

The first objective of the thesis is to develop a general interface for physical human-robot interaction. Such an interface has to allow the user to control the walk of a humanoid by

exerting force on the robot's arm and to lead the humanoid in a particular direction. In other words, by pressing external force on a robot's arm the robot ought to compensate this human influence by following the human. This means that the user should implicitly affect the direction and velocity of the robot walk. The more force is applied to a humanoid's arm, the faster the robot's walking speed should be.

The second objective is to study how to detect external force without force/torque sensors. In most cases the force control is carried out by means of such sensors. Nevertheless, in many robots there are no force/torque sensors and there is still a demand to react on external forces. A good example of such a case is NAO, a humanoid platform which is currently very popular within academia. We utilise this platform for the scopes of this work.

Information deriving from other physical quantities has to be used in order to indirectly retrieve information regarding force. In this thesis the main aim is to use information coming from the motor's current.

Another target of the research is to extend the general interface by implementing different types of force controllers. Consequently, various interfaces should be evaluated in terms of their performances.

The last objective is to design and conduct usability tests for the developed interfaces. As was mentioned above, it is very important for the user to create a functional and effective interface. Therefore, this thesis presents alternative interface solutions based on and evaluated according to their performance through a number of usability tests.

In order to control robot behavior by a human, the manipulated arm is placed in a well-defined initial position. Interaction between the human and robot will not be considered for an arm position that differs from the initial position.

This thesis demonstrates physical HRI by means of force control. The lack of an adequate measuring instrument resulted in the fact that it will not be demonstrated how particular values of external force affect the motor's current measurements. Nonetheless, the proposed solution could be extended towards this investigation.

1.3 Structure of the thesis

This master's thesis has been divided and structured into several sections according to its contents. Each of the chapters is briefly described below.

Section 2 is an introduction to human-robot interaction and presents a literature review on this subject as well as on current trends. Also, this section covers a presentation and comparison of

existing force controllers, their applications and potential problems that might occur during implementation.

Section 3 shows a description of the platform that is used in the implementation part. The NAO humanoid robot is still a new and innovative implementation with growing popularity among academia.

Section 4 contains an overview of the designed interfaces for physical HRI. The general structure for the proposed interface is shown and a brief description of each element is presented. Moreover, it explains how interaction between a human and a robot within the system takes place. Moreover, it proposes an overall software model which provides the functionality for the interface.

Section 5 presents the detailed structure and implementation of the developed interface by means of the NAO platform. It explains the methodology of controlling the NAO's joints and how information from the DC motor can be turned into force control. In addition, it shows the system identification for which a generic interface has been designed. Finally, different controller implementation performances are illustrated.

Section 6 presents the usability test for the designed interfaces. In addition to a concise literature review, the chapter presents suitable usability tests which were initially designed and then used to test the available interfaces. Based on the results, a discussion is presented to clarify and evaluate the obtained outcomes.

Section 7 provides conclusions along with a summary and an analysis of the work. Also, extensions to available controller interfaces are presented as well as their relative application areas. This part sums up and verifies how the determined objectives were carried out.

2. Human–robot interaction

This chapter contains an introduction and literature review of human–robot interaction. It also illustrates the role of force control in the context of pHRI (physical human-robot interaction), which is related to the thesis objectives.

2.1 Introduction to HRI

Originally, robots were mainly involved to execute well-defined repeated tasks, such as manufacturing tasks. Since the time robots have been engaged to work in the human environment, there has been an increasing necessity to make robots more interactive for less structured scenarios. Currently, robots are utilised in almost all areas of life. In addition to service [1] and industry applications, robots are applied in education as teaching assistants [2] or in rehabilitation for post-stroke patients [3]. Introducing robots into human society has formed a new discipline which is human–robot interaction.

HRI refers to studying interaction between a human and a robot. As a multidisciplinary area it involves fields such as engineering, social science, communication and psychology. HRI answers the following questions: How might a human interact with a robot; how should a system for HRI be designed; and what kinds of criteria should be evaluated for HRI [4]. Moreover, scientists are conducting many investigations towards the sociological aspect of HRI. A good example of this is the fact that researchers are trying to find out how robots should be developed in order to communicate in an intuitive and transparent way with people [5]. Therefore, research on designing ergonomic, functional and effective interfaces is necessary [6].

On one hand, it is desired that robots would be used everywhere where a human could be exposed to some dangerous situations, for instance, during a terrorist attack [3]. On the other hand, it is important to secure collaboration between a human and a robot in order to prevent any scenario where a human could be hurt [6]. However, it is impossible to form ready-made models and algorithms for all scenarios of the HRI. Thus, HRI tends to apply artificial intelligence in robotics. The intelligent robot makes a decision with a certain degree of uncertainty. Moreover, in order to increase perception and interaction with the human environment, methodologies based on various and multiple sensors are applied. Mapping the human senses, such as sight, hearing or touch into a robot might create the conditions for a more natural interaction.

One of the most popular methods of interaction between a human and a robot is based on vision. In traditional approaches, i.e. machine vision by means of a video camera, varied algorithms are used to image recognition, e.g. face detection [7]. Lately, methodology using an infra-red projected light sensor has been employed, such as the Kinect device where a

robot using the sensor creates a 3D model based on a retrieved image [8]. In this way it can perceive and recognise human gestures and be arranged to do some forms of action. It has to be remembered that in the traditional picture as well as in the 3D model a robot system tries to fit the captured images into a predefined pattern. Significant dilemmas in those approaches are building a precise recognition algorithm and the quality of the retrieved data, which depends on factors such as ambient colour or illumination.

Another example of the methodology used in human–robot interaction is collaboration by means of speech recognition. Just as in the visual methods, the audio signals are initially filtered and then proceed to the recognition algorithm. New research is currently investigating the impact of human verbal and non-verbal traits on interaction with the robot [9]. Every human speaks and gestures in a slightly different way and those facts may also be used in HRI.

In order to supplement the above-mentioned methods, HRI is extended by physical human–robot interaction. pHRI is discussed here more extensively because of its relevance to the content of this thesis. Today pHRI is greatly desired in many applications, but due to its complexity it still remains a largely unexplored area. However, current work results on pHRI are being widely employed in applications such as in rehabilitation, object manipulation or service robotics. In many cases robots are used to enhance physical force as well.

One of the most recent research studies in pHRI has been an investigation into cooperative dance [10]. With the exception of appropriate balancing, the robot has to make steps correctly and according to pressed force by a human partner. To make such decisions the robot has to be equipped with accurate sensors and advanced control algorithms.

According to the anticipated application, every robot has to fulfill specific criteria, such as reaction time, accuracy or safeness. As was mentioned at the beginning of this section, safety is an especially crucial aspect of pHRI [4, 6]. Thus, diverse sensors are utilized, such as force/torque or tactile [11], which enable to determine the localisation of a contact point. These allow the robot to become more capable in terms of acting on manipulated objects as well as more perceptive on the force that is exerted by a human. This is currently a very important topic and many studies are being conducted towards tactile sensors, e.g. there are research studies involving flexible robot skin for pHRI [12].

Pressure sensitive skin can be placed on geometrically complex robot surfaces. Thus, a robot equipped with that type of a skin can precisely assess the contact point with a human and avoid collision [13]. Besides, this method significantly enhances the opportunity for interaction between a human and a robot.

A different but growing aspect involving HRI is robot Learning by Demonstration (LbD). This subject is especially interesting due to the fact that the robot does not have to be manually programmed. Thus, LbD constitutes a huge advantage to those users who do not have programming skills. In addition, it significantly speeds up the process of acquiring new

capabilities by the robot. LbD consists of two phases: the first stage is called observation, which is based on teaching the robot new behaviour by demonstrating a task. After that the reconstruction phase is conducted during which the robot executes a previously taught action.

LbD can be found in many applications, such as in a household or a workshop, as usually tasks in those places are repeatable and structured. Nonetheless, performance might differ due to various environments. A ready-made program cannot be applied without modification, which constitutes a drawback for unskilled users. Thus LbD overcame such an obstacle, e.g. in service robotics many investigations regarding LbD and autonomous navigation are being conducted [14].

Learning by Demonstration has been applied in physical human–robot interaction as well. In this case the robot is taught by physical contact with a human who kinesthetically displaces particular robot joints into desired positions. As was mentioned before, at the beginning the robot learns the movement by recording either the external force that was exerted by the human or the joint displacement. Later, when the reconstruction phase takes place the robot replays the movements with a certain force/torque or velocity.

On the whole, due to increasing popularity of learning by physical demonstration, criteria are examined for evaluating LbD, such as usability tests [15]. In this paper the investigators analysed the usefulness of LbD with regard to diverse force controllers regulating the robot during the learning phase.

2.2 Force control

As was mentioned in the previous subsection, force control is an important aspect in physical interaction between a human and a robot. There are many algorithms regulating the contact force between human and robot as well as between robot and environment. Most of them are based on a dynamic model of the robot together with information from force/torque sensors [16, 17].

Each controller features a different complexity and performance. From the user’s perspective different controller performance can be considered as various interface usability [15]. Thus, some controllers have to be reviewed in the context of this thesis.

In [16, 17] a traditional impedance controller as well as a position-force controller are implemented. Admittance control is briefly shown in [18]. For simplification, the above-mentioned methodologies are presented below with the assumption that the controlled object is a mass-spring-damper system. The relation between exerted force placed on that object and the occurring displacement can be described by the following Laplace transfer function [16]:

$$Z_M(s) = Ms + B + \frac{K}{s} \tag{2.1}$$

where Z is the object impedance, M is the mass, B is the damping parameter, K is the spring constant.

The goal of the impedance controller is to regulate the assumed mechanical impedance of the inspected object, such as a manipulator. Such a manipulator can respond to the exerted force according to the adjusted parameter of the mechanical impedance - Z_M. For instance, force exerted on the manipulator may displace it but, at the same time, the manipulator opposes the external force with some resistance. That resistance can be tailored by parameters K, B, and M. Thus, the change of manipulator dynamics can be done by means of adjusting the mass - M. Figure 2.1 presents the structure of impedance regulation, where:

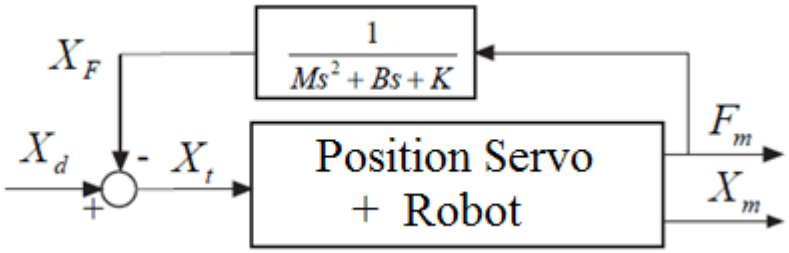


Figure 2.1: Impedance controller [16]

F_m denotes exerted force, X_m signifies position displacement, X_d represents set position and X_t symbolises control signal.

In [19] the impedance control method is applied to man–robot cooperation. On the whole, the authors utilised two regulators for the designed system. During the experiment part the manipulator is manually maneuvered by the human, who exerts force on the robot. Based on information from the sensors, that force is utilised by the first impedance controller to execute the proper motions. The other controller regulates the force exerted by the robot to its working environment.

The position-force controller can be considered to be a proportional-integral regulator (PI) where the damping parameter is identified with a proportional term - Figure 2.2 . By using the integral term the steady-state error is eliminated [18].

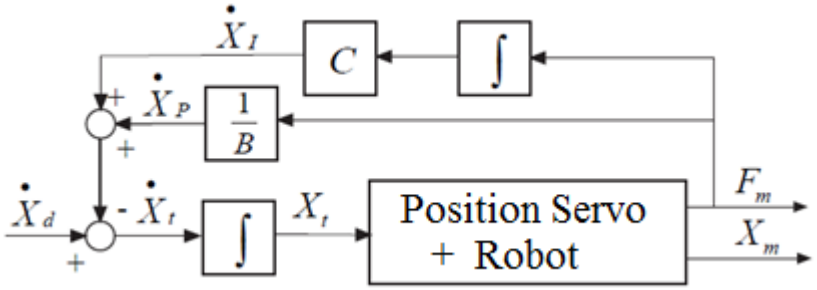


Figure 2.2: Position-force controller [16]

A different approach than the methods specified above is discussed in [20]. It is presented based on the LbD in industrial operations. Manual maneuvering of the robot to teach it activities such as painting and lacquering has currently been mastered. However, there are still ambitions to enhance human–robot interaction, for instance, by making manipulation of a robot similar to operating a spray gun. Therefore, the authors proposed the usage of a virtual tool together with an admittance controller. Implementation of that solution gives the user the impression of operating a robot with the dynamics of a virtual tool. To regulate such dynamics the above-mentioned admittance controller is integrated – Figure 2.3. In response to the exerting force/torque by the human, it adequately displaces the robot joints.

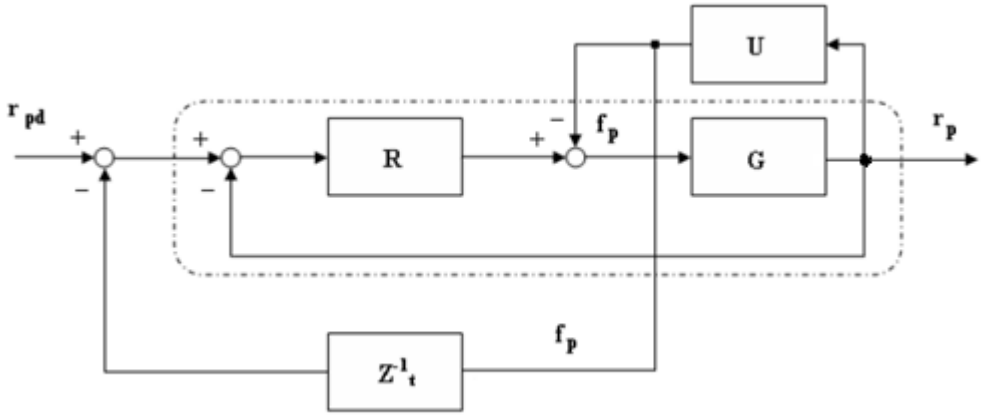


Figure 2.3: The position-controlled system with the admittance controller [20]

The dotted box illustrates the position control of the industrial robot. Outside of this there is an impedance controller which impacts on the input of the position regulator - R. The dynamics of the human and robot is accordingly presented by blocks U and G. As is illustrated in Fig. 2.3, the admittance controller is affected by human force exerted on the robot. In addition, the linear block Z_t^{-1} corresponds to the admittance filter. Hence, the new position of the manipulator might be formulated as:

$$r_p = r_{pd} - Z_t^{-1} f_p \tag{2.2}$$

where Z_t^{-1} is the admittance filter which relies on the damping factor and the mass of the virtual tool. f_p denotes the force exerted by a human, r_{pd} is reference trajectory and r_p signifies new position.

Another approach involving a virtual tool and force control can be seen in [15]. Their usage in LbD is studied as well. In addition, the authors discuss the technical issues regarding gravity compensation and singularity management. On the whole, different force controllers are proposed to those that were presented earlier, e.g. a proportional controller and its extension, the virtual-tool controller.

In the first case the controller makes the proportional regulation of the end-effector velocity to the force/torque exerted by a human. The structure of this controller is presented in Fig. 2.4

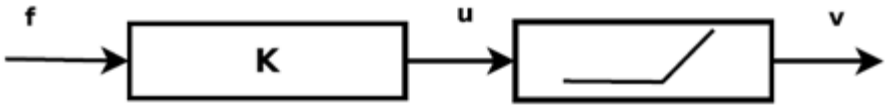


Figure 2.4: Simple force controller basic diagram [15]

The force introduced to the proportional block, K, is reduced by the gravity component. In effect, it is only the quantity acted by the human. After amplifying that signal it is examined at the threshold block. The value of the threshold is calculated with respect to the norm of the exerted force or the torque vector.
The lack of a feedback loop with an additional correction block is an essential drawback in this control method. The rapid fluctuation of the exerted force may result in a steep change of the robot end-effector velocity.

The second controller from [15] utilises the virtual tool and it provides smooth and natural movements. Moreover, it requires less force to displace the robot joints with the same velocity than in the case of the proportional controller. Like in [20], it is based on the assumption that the robot end-effector is modeled as a virtual point and its acceleration is controlled. Consequently, the force applied to the virtual mass influences the robot acceleration.

Until now, force control methods were discussed with the assumption of knowing the force/torque value. In many cases, as in [21], there is no access to such sensors. Nonetheless, there are still needs to control the force/torque exerted on the robot. Therefore, some methods can overcome that obstacle.

One sensorless force estimation method is investigated in [22]. The paper presents a way of determining the force/torque applied at the end of the robot effector by means of the motor current and shaft position.
A further study regarding this method was conducted in the context of grasping [23]. For research purposes a humanoid equipped with current sensors as well as encoders was used. In

the experiment the robot hands aim to squeeze a given object which has a certain stiffness. As a result, the reaction force opposes the force generated by the robot joints.

The authors discuss the common relationship between the produced force, the motor's current and the robot arm's distance. This information is then reliable in order to carry out the design of the force controller. Thus, the authors propose that the force controllers work by means of the motor's current and the distance between the arms.

The first controller in [23] works by using information from the DC motor. Supposing that the DC motor produces a torque that is proportional to its armature current, it is formulated by the following control rule

$$u(t) = \max\{d_{min,}\, u(t-1) - \lambda<\tau_{max}(y_c(t))^* \delta\} \quad (2.3)$$

where:

y_c is measured motor's current, d_{min} denotes minimum distance between the arm, τ_{max} is threshold for the motor's current, u presents requested distance between the arm , λ signifies characteristic function, δ symbolises the step of the descending distance between the arms in each iteration.

The paper also proposes another controller that relies only on the arms' distance, so it does not contain information about the force/torque produced by the joints. Therefore, it is not discussed here.

2.3 Conclusion

In this section different methods of interaction between a human and a robot have been presented in the context of a literature review. Also, pHRI was more extensively discussed due to its significant relation to the practical content of the thesis. To sum up, HRI and pHRI might be considered from two different points of view.

Firstly, the quality of interaction is evaluated by human users. From an analysis of the available literature, this subject seems poorly addressed. In [4, 5] criteria are presented as to what HRI is supposed to fulfill, but there are no proposals how to conduct those evaluations. In contrast, in [15] the usability of different force controllers based on the user's feelings is evaluated. Nevertheless, there was no investigation towards evaluating interaction on the basis of the interface perspective. Therefore, this subject will be studied in the thesis.

Secondly, most of the force control methods presented are based on information from a force/torque sensor. In addition, [22, 23] show techniques how to deal with force control if such sensors are not available. In both papers, use of the motor's current is proposed in order

to estimate the force/torque. However, these solutions are appropriate when it is necessary to estimate the force which is exerted by the robot rather than on the robot, which remains a dilemma in this thesis.

3. The NAO humanoid robot

In this section the research platform, i.e. the NAO humanoid robot, is presented - Figure 3.1. The chapter is divided into four subsections. Section 3.1 briefly introduces the reader to the robot and its capabilities. It is worth noting due to the innovativeness of the NAO platform. Section 3.2 describes the hardware of the NAO robot in more detail, including its mechanical architecture. In addition, a description of the actuators, due to their essence in system identification as well as force controller design, is presented. Section 3.3 is devoted to a software architecture description. Apart from the software framework, the section aims to characterise the possibilities of developing software for the robot by using various tools. Section 3.4 describes the accessory applications that enable the user to control the robot's behaviour in an ergonomic way.

Figure 3.1: Aldebaran-Robotics NAO humanoid [21]

In general, the above-mentioned information regarding hardware and software architecture constitutes a short summary of the functionality of the NAO platform, which is necessary and sufficient in order to conduct the tasks described in the other chapters of the thesis. A comprehensive description of the NAO platform capabilities can be found in the technical documentation [21].

3.1. Background

NAO is an autonomous biped robot which was first released in 2004 by the French company Aldebaran-Robotics. Due to its capabilities, design and low price, compared to its counterparts, the NAO has become a solution for those who need access to a robot offering high performance and simplicity of handling. Also, for these reasons the NAO has been selected as a standard platform in RoboCup, an international robot soccer competition [24].

3.2. Hardware

The NAO robot's specification, as presented in this subsection, involves version NAO V4, and this platform was used during the experimental part. The essential parameters are given below in the Table 3.1.

Body	
Height (m)	0.57
Weight (kg)	4.5
Degrees of freedom (DOF): 25	
Head	2 DOF
Arms	5 DOF X2
Pelvis	1 DOF
Leg	5 DOF X 2
Hands	1 DOF X 2
Masses [g]	
Chest	1217.1
Head	401
Upper Arm	163
Lower Arm	87
Thigh	533
Tibia	423
Foot	158
Total	4346.1

Table 3.1: Technical parameters of the NAO humanoid [24]

As shown in the table, the NAO has 25 degrees of freedom (DOF), in which 11 of them are performed by the lower part of the body and the rest are achieved using the upper part. This guarantees that movement of the robot looks like human motions. In addition, each of the robot's joints is actuated by a high quality brush DC motor. In general, in the NAO robot three types of DC motor can be distinguished, which are presented in Table 3.2. For each motor type there are two speed reduction ratios. This fact imposes that in the robot six different types of actuators are used to rotate the joints.

	Motor type 1	**Motor type 2**	**Motor type 3**
Manufacturer	Portescap	Portescap	Portescap
Model	22NT82213P	17N88208E	16GT83210E
No load speed	8 300 rpm ±10%	8 400 rpm ±12%	10 700 rpm ±10%
Stall torque	68 mNm ±8%	9.4 mNm ±8%	14.3 mNm ±8%
Nominal torque	16.1 mNm	4.9 mNm	6.2 mNm

Table 3.2: Motors in the NAO - V. 4.0 [21]

In order to control the humanoid's joints accurately, the robot is equipped with MREs (Magnetic Rotary Encoders) using the Hall effect. By and large, the MRE is a small electronics chip detecting the angular displacement of a magnet. The chip as well as the magnet are placed either on a motor or on the robot's joint.
There are two configurations for attaching the MRE. First of all, for bigger motors, such as those located in the lower part of the robot, two encoders are used. One is installed on the motor and the other is placed on the corresponding joint. In the second configuration, where there are smaller motors, i.e. in the upper part of the robot body, one encoder is attached to the robot joint. In the first case the solution is much more precise. On the whole, the sensors feature 12-bit precision, i.e. 4096 values correspond to 0.1° precision [21]. The output signal of the encoder is the absolute value, and based on that information all of the motors are controlled using the position feedback, which is presented in Figure 3.2. For configurations with two encoders, position control is performed by means of the MREs being placed on the motor.

The block diagram below presents the positional feedback control, where the controller regulates a motor's power by adjusting the PWM duty cycle. The K_s parameter is used as a scaling (0–100%) factor to the applied PWM duty cycle. Thanks to this parameter the user is allowed to master the stiffness of each of the robot's joints. [25]. By decreasing the stiffness the motor's torque is reduced accordingly, and when the stiffness of the motor is set at 100% then the motor's torque is maximal.

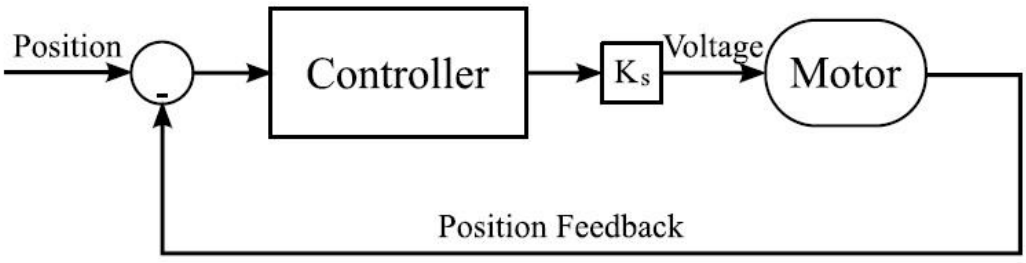


Figure 3.2: Block diagram of the motor's positional feedback control [25]

For computation, the NAO is equipped with ATOM Z530 1.6 GHz and 1 GB of main memory, which are embedded in the motherboard located in the robot head. The CPUs are used to support the operating system as well as the user's applications. In addition to the

central unit, in the robot torso an ARM-7 microcontroller is placed which is responsible for sending information to the actuators.
Generally, several types of system buses can be distinguished. For instance, RS-485 is utilised to communicate between ARM-7 and the actuators' microcontroller – Microchip 16 bit dsPICS. Other examples are CPU and ARM-7, which interact with each other using a USB.

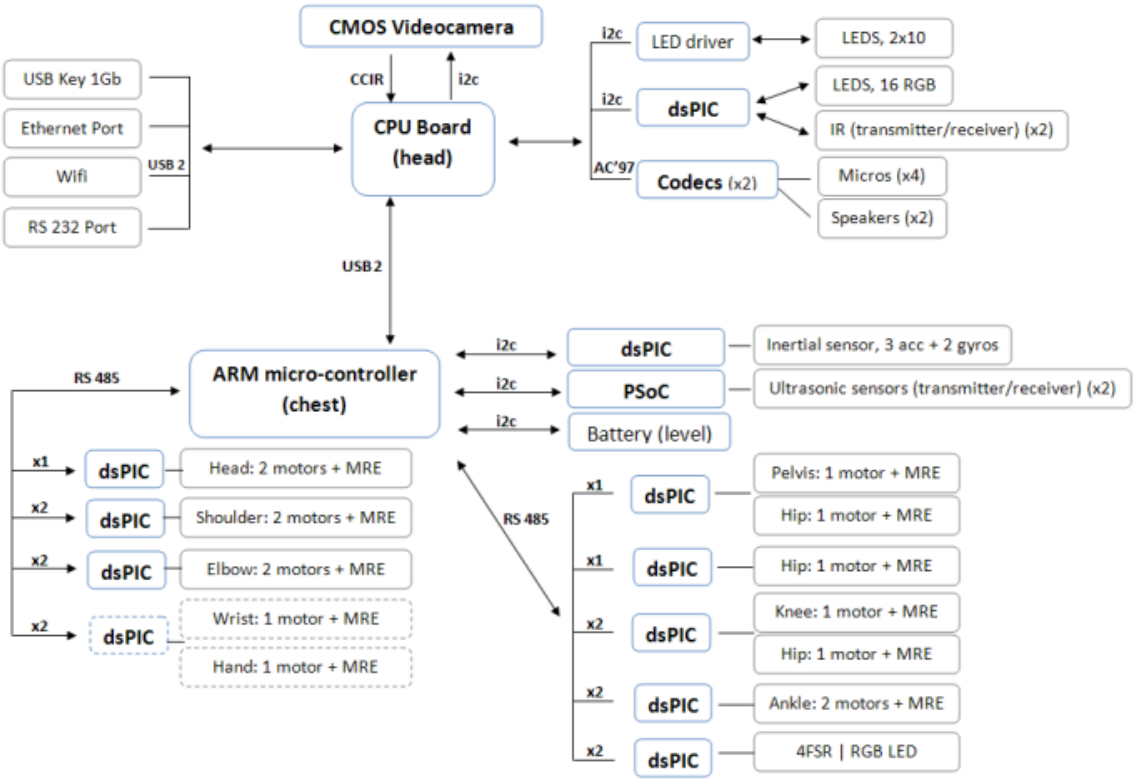


Figure 3.3: Electronic architecture of the NAO robot [21]

As is presented in Figure 3.3, most of the actuators are directly controlled by means of the dsPICs. In addition to regulation function, the microcontroller provides access to interial, ultrasonic and battery level sensors. The sensors and actuators used in the practical part of the project are described below in detail.

JointHardness actuator determines the stiffness of a particular joint in a range of from 0 - 100%. The actuator is denoted as the K_s parameter in the Fig. 3.2. Thus, by adjusting the value of JointHardness, the actuator regulates the power delivered to the motor.

Current sensor measures the value in Amperes for the battery and a motor. For each motor board there is a current sensor in the shape of a resistor placed between the H-bridge and the ground potential. The current flowing through a motor is then calculated based on the voltage drop over the resistor. The value of the voltage is measured by an ADC microcontroller. Nevertheless, unless the bridge is powered by means of PWM, the default value of the motor's current is zero. In addition, each motor board limits the current in order to protect

itself, the motor and the mechanical parts of the robot. Thus, if the value of the current exceeds a limit, the PWM duty cycle will be decreased to reduce current.

Temperature sensor – the value of the temperature is returned in Celsius units by two devices: motor board and battery. In the case of the motor board, the temperature is simulated based on a motor's current. Moreover, the board imposes a limit regarding temperature, so its value also influences the motor's current limit.

Apart from the described sensors and actuators, the NAO has the following sensors presented in Table 3.3:

Type	**Quantity**
Loudspeakers	2
Microphones	4
Video camera	2
Infra-Red sensor	2
Force Sensitive Resistors	8
Accelerometer	1
Gyrometer	1
Sonars	2
Magnetic Rotary Encoders	34
Contact and tactile sensors	12

Table 3.3: Sensors of the NAO - V. 4.0 [21]

3.3. Software

NAOqi is a framework utilised to build applications and to program the NAO robot. It defines the structure of an application as well as the way an application works. Moreover, the NAOqi framework is responsible for delivering libraries which are necessary in order to perform a particular task. So, in that sense, a software developer builds up and adjusts the components of an application according to project requirements.

The NAOqi framework is a cross-platform framework because it facilitates development and compilation of software for various system platforms, such as Windows, Linux or Mac OS. Simultaneously, the framework allows to cross-compile an application to the robot's operation system. Another feature of the NAOqi is the fact that it is cross-language, providing support for various programming languages. To obtain full access to robot functionality, it is recommended to use C++ or Python.

As was mentioned before, the NAOqi architecture offers an application programming interface on the basis of libraries. A library consists of one or more software modules, and each module can be considered as a software class with submitted methods. Additionally, the API determines how system components interact with each other. Besides, the NAOqi allows to extend the API by attaching own modules and their methods. Such a module can later be utilised as an original module delivered through the manufacturer of the framework. More information about this is explained in the further part of this subsection.

In the NAOqi architecture three elements of the system are crucial in order to develop own software for the robot, i.e. the module, the broker and the proxy. Therefore, they are characterised below.

The module is an element of the NAOqi architecture that might be considered as a class with its interfaces, i.e. methods. Every module is placed in a certain library according to the functionality it facilitates. At the moment of NAOqi booting process, a file of available libraries (autoload.ini)is loaded. Then, instances of modules are initiated.

The modules are grouped into local and remote modules according to their location and mutual connection. The remote module is a compiled, executable file and might be run outside the robot, for instance, on a user's workstation. In effect, such a module provides easier debugging but, on the other hand, it carries some limitations, such as slower execution of the program, which is caused by the network connection between the remote module's broker and the main broker run on the robot. A remote module needs its own broker in order to communicate with the others, which are situated in the range of the main broker. A broker is an executable program, i.e. a process which is, among others, in charge of network communication. More about the broker is explained in the next part of the subsection.

Another example is the local module compiled as a dynamic library. It enables to execute a program much faster than in the case of the remote module. However, a local module can only be run and utilised on the robot where the module was attached. Local modules are run in the same process and they communicate with each other by a common broker. Thus, they can share the memory and methods for themselves without using the network connection. This is a faster way of interaction between modules. In the NAOqi architecture there are standard modules which are grouped under their destiny. Some of them, i.e. ALMemory, ALMotion and The Device Control Manager are employed in the practical part of this thesis work, therefore, they are briefly presented below:

ALMemory provides access to memory where information is stored from sensors and events. By using this module the robot memory may be used to transfer data between the modules as well as for mutual communication within the processes. In the robot memory, data are represented as variables and access to them is secured thanks to a mutual exclusion algorithm.

ALMotion supports control of robot movements by regulating a particular joint or a whole limb. Besides, ALMotion delivers ready-made functions which allow to perform complex actions and postures, such as standing up or walking.

The Device Control Manager is in charge of communication between the electronic devices of the robot. It constitutes a conjunctive layer between low level software from the hardware and a high level application based on modules - Figure 3.4. All commands are sent to the actuators by means of DCM. Also, information regarding sensors is delivered to ALMemory by the DCM module.

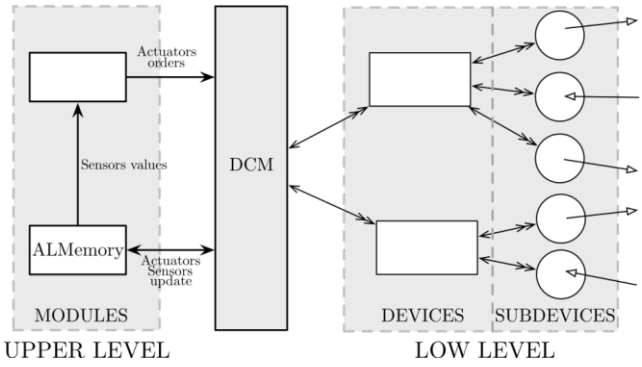


Figure 3.4: The Device Control Manager as a conjunctive layer [26]

In addition to the framework, the NAOqi is the name of the main broker, i.e. the process running on the NAO which controls the robot's behaviour. During the start-up of a broker, the autoload.ini is loaded, a file with information regarding which library should be submitted. In the earlier part of this subsection it was mentioned that each of the libraries is a collection of modules and that the modules are published by access to their methods. This access is visible to other modules located in the range of the common broker as well as to the other brokers or proxy ones which are connected via the network. Methods offered by the remote and local modules are attached to the broker. They create a tree structure, as presented in Figure 3.5.

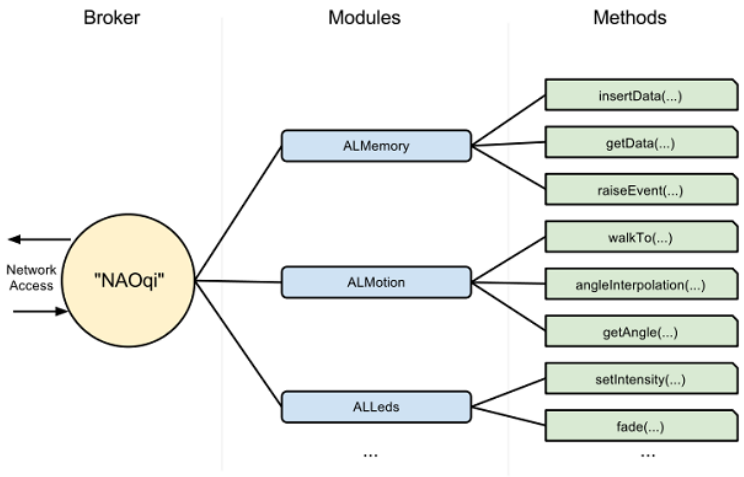


Figure 3.5: The broker and its modules [21]

One of the broker functions is to provide access to the modules and their methods. Another specialty is to deliver and manage a network connection. Due to this it is feasible to call in module methods from outside the process. Thus, a broker reveals module interfaces to the rest of the architecture.

In the NAOqi architecture it is possible to create several brokers offering their services. Nonetheless, in order to share own modules the brokers have to connect with the main broker running on the NAO. In this way a tree structure of brokers can be differentiated in the parent and child relationship as presented in Figure 3.6.

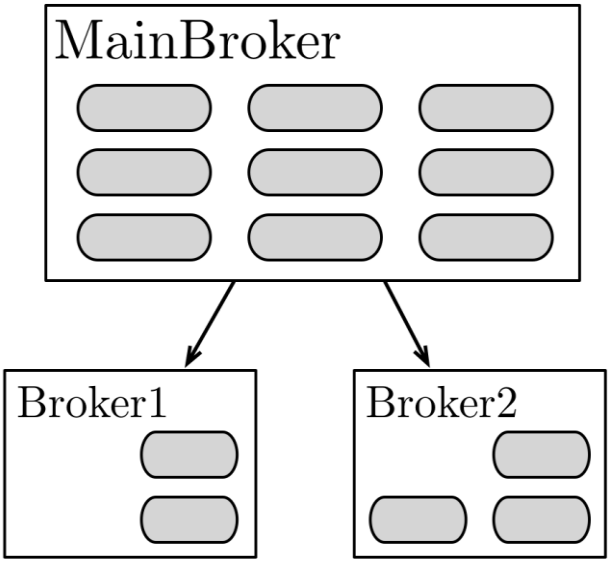


Figure 3.6: Broker tree structure [26]

The proxy allows to utilise modules located in a broker. It connects to a particular broker and retrieves information about a desired module. The proxy is an object which behaves like a module which represents, i.e. a proxy offers the same methods which are delivered by a certain module. There are two options for creating a proxy. First of all, when the proxy and its module are located within the same broker, this is called a local call. The other is to connect with a module situated in another broker. For this purpose the module's name as well as IP address and the port of the broker which the proxy wants to employ all have to be determined.

In general, the following types of connection may be distinguished between the objects as described above:

- Connection within brokers – it allows mutual data transmission, therefore, it is also thought of as a symmetric link between the objects.
- Broker and proxy connection – it constitutes an asymmetric link between the objects. A proxy might communicate with a particular broker and represent all of its modules. In contrast to this, the broker forms a passive attitude and it just publishes its own services.

3.4. Development tools

In this subsection the development tools allowing rapid programming and testing one's own project for the NAO robot are described.

Choregraphe is a programming environment that enables one to develop software for the NAO. The user may develop robot behaviour by using a graphical programming language. Moreover, libraries of standard functions are available where each function is represented as a box. From this set of boxes the user can build his/her own program by dragging a box and dropping it into the workspace. Through linking the boxes in the workspace, more complex robot behaviour can be constructed.

Choregraphe supports sequential, parallel and event-based software. Moreover, it is worth mentioning that the application offers an online and offline work mode. Thus, it is possible to test one's own code on a real robot or on the simulator. The main window of the Choregraphe is illustrated in the picture below.

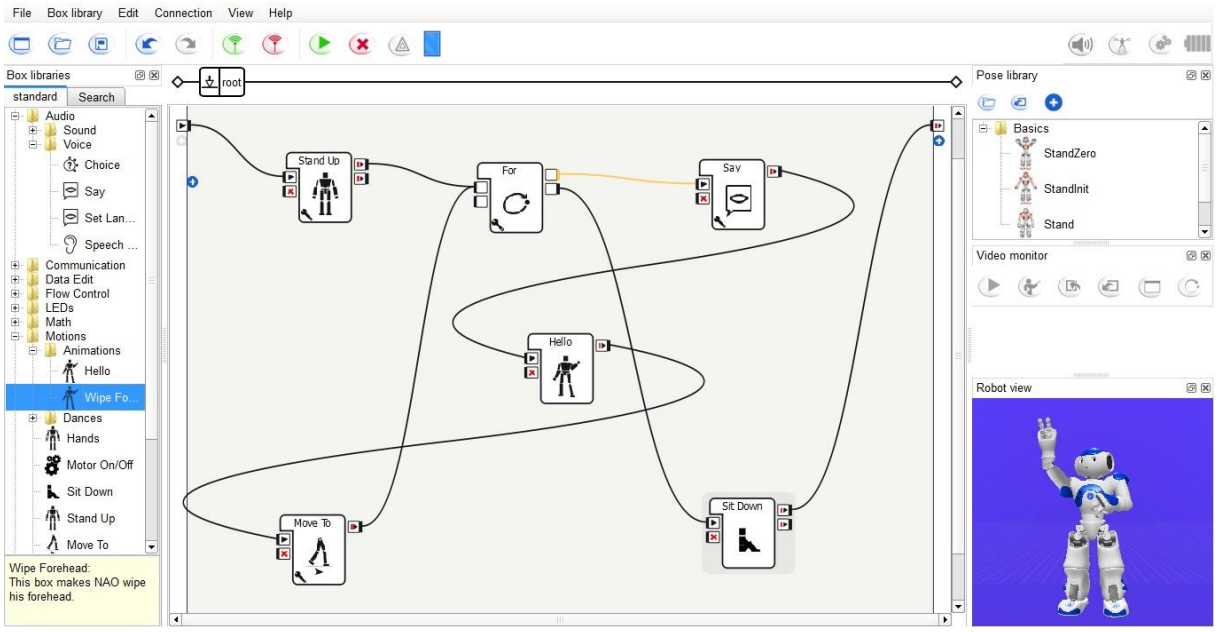


Figure 3.7: An example program in Choregraphe application

The left-hand side of the Fig. 3.7 presents the libraries of boxes collected according to the functionalities they offer. In the lower, right-hand corner there is an animated view of the robot. The middle part of Choregraphe's main window is dedicated to the workspace, which is called the flow panel. In this place the user can create his/her own program based on adequately linked boxes.

In the example presented in Fig. 3.7 the program begins with a robot's standing up action, then it gets into a "for loop" mode. Inside the loop, sequential speech and movement functions are executed. After overcoming the loop condition the program is ended by performing a sitting action. It is worth noticing that the user is able to design new boxes in Python, and then to submit them to the main library of Choregraphe.

The monitor application gives access to the sensor and memory of the robot. Thanks to this the user can very easily verify correctness of the robot's behaviour.

The Software Development Kit is a set of tools supporting software development for the NAO. Apart from the API libraries it provides tools for compilation and debugging. SDK is made available in 8 programming languages, such as C++, Python or Matlab. Besides, SDK makes it possible to develop own modules for NAOqi. As a result, code written in C++ or Python can be directly launched on the robot.

4. Interface for physical HRI

This chapter briefly discusses the design of the interface constituting a solution for the objectives that were set out and restricted in section 1.2. The chapter is divided into three subsections. Section 4.1 describes the conceptual structure of the interface. Moreover, particular elements of structure, their mutual interaction as well as their interpretations are discussed. Section 4.2 involves conceptual software modeling of the designed interface by means of SysML notation. Different diagrams illustrating software structure and dynamics are presented. Finally, in section 4.3 a general methodology of designing the interface is proposed.

The interface's structure consists of two layers, i.e. the hardware and the software. The hardware layer does not impose a strict framework, but it does determine a certain specification which should be held. The other layer is standardised by generic models which are presented and described in further parts of this section.

In general, the interface is supposed to provide physical interaction between a human and a robot. Additionally, it allows the user to control robot walking by exerting force on the robot arm. In effect, the robot compensates that impact by changing the direction and velocity of the walk according to the exerted force. In other words, it might be considered that the user leads the robot by the hand.

The interaction between the human and robot should be as natural as possible, i.e. it ought to resemble a corresponding action made by a parent and a child. Hence, the projected interface aims to fulfill that requirement. It is worth mentioning that the interface aims to be employed to any humanoid where there is a desire to conduct such interaction as the kind described above. Moreover, the project constitutes a solution for pHRI in the case of a lack of force/torque sensors.

First of all, the interface is intended for a humanoid robot equipped with DC motors which provide access to the motor's current sensors as well as encoders. By means of those sensors, information can be retrieved regarding the force/torque generated by the robot joints.

Secondly, the framework should be equipped with a position regulator controlling a particular joint. On the basis of that, the regulator intends to hold a specific joint in a given position by changing the duty cycle of the motor's supply voltage. The changes of voltage significantly depend on the motor's load, i.e. the exerted force/torque. On the other hand, the motor's voltage has a direct impact on the current flowing through the motor. More about this is discussed in the implementation part (Section 5).

4.1 The conceptual structure of the interface

Figure 4.1 illustrates the conceptual structure of the interface. It presents the most essential blocks needed to conduct interaction between the human and the robot. Some blocks represent the mechanical or hardware aspects, while others embody the software algorithms.

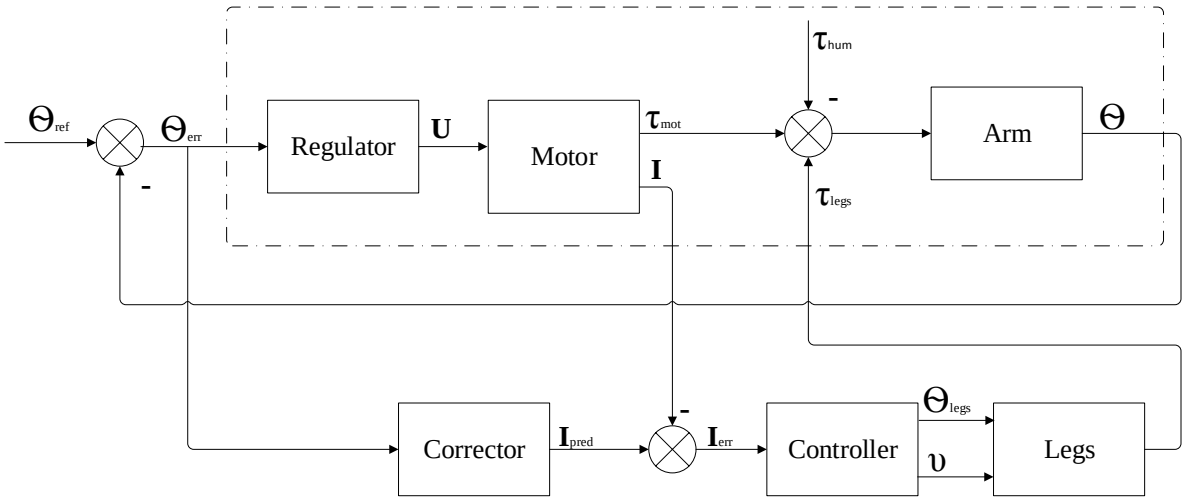


Figure 4.1: Conceptual structure of the interface

where:

Θ_{ref}	signifies reference position for the position controller as well as the desired position of the robot arm
Θ	is present position of the robot
Θ_{err}	denotes arm position error, i.e. the difference between Θ_{ref} and Θ
Θ_{legs}	represents the direction of robot walking
υ	is the velocity of robot walking
$\mathbf{U}$	signifies motor's supply voltage
$\boldsymbol{\tau}_{hum}$	represents human force/torque exerted on the robot arm
$\boldsymbol{\tau}_{legs}$	denotes force/torque generated by the robot legs
$\boldsymbol{\tau}_{mot}$	signifies motor's force/torque exerted on the robot arm
$\mathbf{I}$	symbolises motor's current
$\mathbf{I}_{pred}$	is motor's current estimated by a corrector
$\mathbf{I}_{err}$	reflects estimation error, i.e. the difference between $\mathbf{I}_{pred}$ and $\mathbf{I}$

In the above Figure, two control loops can be distinguished, i.e. feedback and feed-forward. Each of these is engaged in different tasks, but only their common collaboration enables the pHRI.

The feedback loop is partially marked by a dashed line and it illustrates the position control system, which is required by the previously made assumptions. The position control system is

in charge of holding the robot arm according to the initiated position - Θ_{ref}. The output of the system represents the actual position of the robot arm. The difference between the reference and output signals is introduced to the regulator block. The regulator block embodies the control algorithm, and its output is strictly dependent on its input. In general, the regulator is involved in generating such an output signal that contributes to displacing the robot arm in accordance with the desired position. In the case of setting the arm to a reference trajectory, the regulator does not generate any signal.

On the other hand, the regulator output affects the motor's input (supply voltage), and thus, in effect, also the output. The force/torque produced by the motor is related to its armature current.

In addition, the motor's force/torque is directly applied to the robot arm. Nonetheless, on the presented structure the force/torque signal is connected to the robot arm through the summation-node. The reason for this is the fact that other forces/torques are exerted on the robot arm as well. The force/torque is exerted by a human and the reaction force/torque is generated by the robot legs during the walking.

Basically, as was pointed out before, the motor's current is related to its torque. Moreover, these quantities are indirectly dependent on the regulator input, i.e. on the robot arm position. Assuming that the robot arm is on a desired trajectory, the controller does not produce any signal and, in effect, the motor's current is zero. In contrast to this, while the human displaces the robot arm by exerting the force/torque, the input signal of the regulator is changed. In effect, the motor's current and the produced torque increase as well.

The discussed properties are employed in the feed-forward loop which embodies the authorial contribution in the interface development for pHRI.

On the whole, the motor's current is compared in a summation-node together with the estimated signal by the corrector block $\mathbf{I}_{pred}$. The corrector block is in charge of refining the measured signal, for instance, by eliminating factors such as gravity effect or inaccuracy of position control. As a result the corrected signal is introduced to the controller block.

The controller block regulates the robot's walking by adjusting the supplied signal to the legs block. By and large, the employed controller significantly influences the whole interaction between the human and the robot. Nevertheless, the choice is up to the designer.

The legs block embodies general movements of the robot limbs. Therefore, its inputs are presented by means of direction and velocity. On the other hand, the generated output of the block, i.e. the force/torque impacts the final force/torque exerted on the robot arm.

4.2 Conceptual software modeling and analysis

In this subsection system modeling and analysis are conducted by means of System Modeling Language SysML. SysML is an extension of UML which provides notation that is well suited for a system engineering application such as automation or robotics.

In general, SysML distinguishes several kinds of diagrams describing the structure and behaviour of the designed system. Some of the diagrams are employed to model a software system for the projected interface. Therefore, this subsection is divided into four parts in accordance with the diagrams that are used.

4.2.1 Use case diagram

The use case diagram presents a functionality that is provided by the system from the user perspective. Each use case embodies a separate function. Figure 4.2 illustrates three generic facilities offered for the user. Two of them are in charge of system activation and deactivation. The last one represents the functionality to operate the system. By this means the user is able to interact with the system, which has already been discussed at the beginning of the chapter.

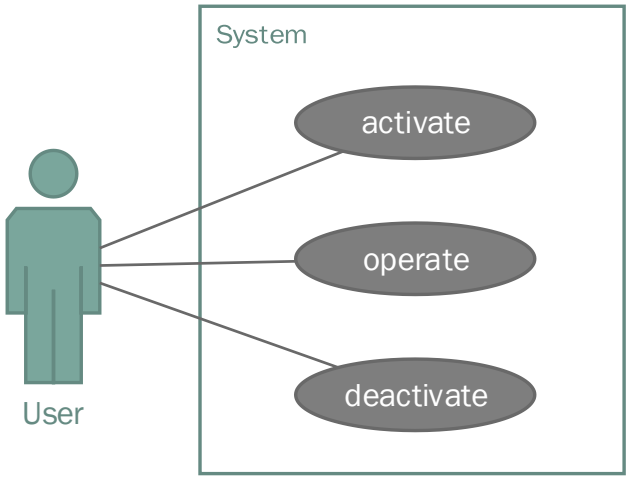


Figure 4.2: Use case diagram of the designed interface

4.2.2 Block definition diagram

The BDD is a kind of diagram that might be used to design and analyse the structure of any engineering system. On the whole, the diagram consists of blocks that are characterised by their properties and operations. Furthermore, the blocks are connected to each other by a relationship. In general, a block is a modular unit of structure that can define a component

such as a conceptual entity, logical abstraction, subsystem or item that flows through the system.

In this case the BDD is utilised to conceptually design the control application of the developed interface. Hence, there is no exact information regarding the properties and operations regarding each block. The choice of implementation is left to the designer's approval. Nonetheless, we will discuss what kinds of facilities should be offered by each block.

Figure 4.3 presents the application structure of the designed interface by means of BDD. The blocks are sorted in accordance with a tree structure and, therefore, the main block corresponding to the whole application is placed at the top of the structure.
From this block there are connections to the other blocks by means of composition relationships. This means that the other blocks constitute a part of the whole system and that no block can exist without the main, top block – the Application.

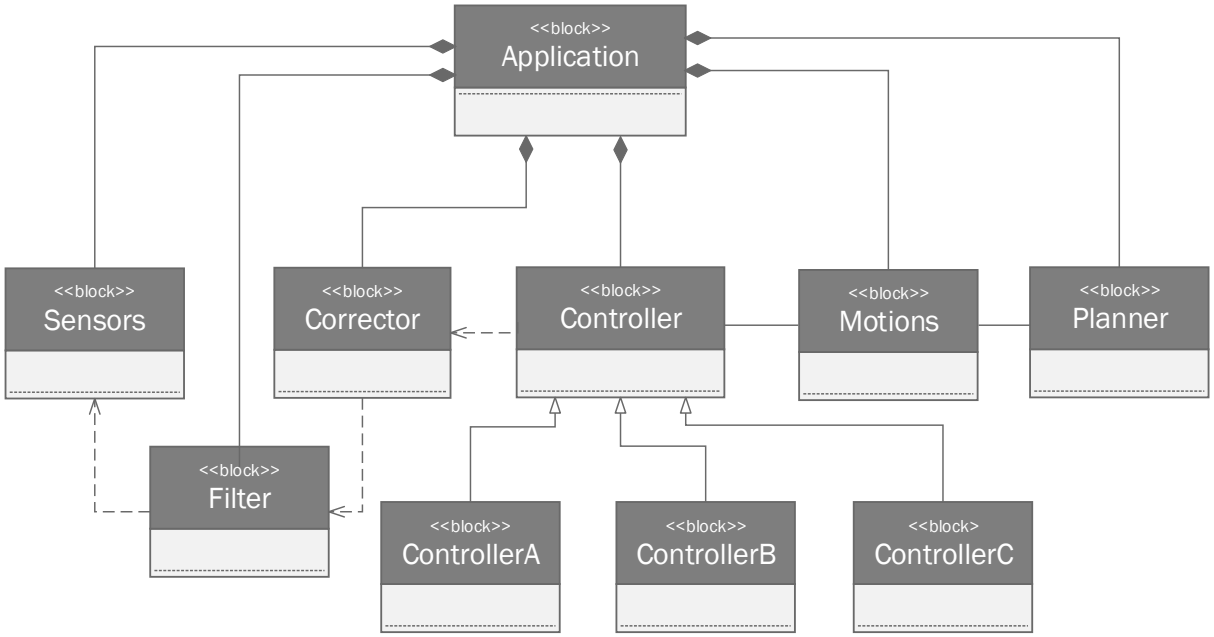


Figure 4.3: Block definition diagram of the designed interface

The first block on the left-hand side, i.e. Sensors, represents the application module which is in charge of retrieving information from the physical sensors, such as the motor's current or the encoder. Besides, the block stores captured information and shares it with the other blocks if there is a request for them.

The other blocks are Filter and Corrector. The first block implements the algorithm filtering the sensors' signals to useful forms. The block is in a dependency relationship with the Sensors block because its value directly depends on the values provided by the Sensors block.

Correspondingly, the Corrector block depends on the Filter block. The corrector embodies the software module which is in charge of refining the filtered signal from other inadequate factors, such as the gravity effect or inaccuracy of position control. Finally, the corrected signal is stored and facilitated.

The main task of the Controller block is to make proper decisions based on information from the Corrector block. The block includes methods to regulate the robot's behaviour. For this purpose it utilises the Motions block. Nevertheless, there is no restriction as to how control decisions are made and what information should be stored. It only embodies the general concept of the control module. As was shown in the above Figure, the Controller block is in a generalisation relationship with ControllerA, ControllerB, and ControllerC. The reason of this is the fact that according to the thesis objective (Subsection 1.2) it is required to compare the performance of several controllers. Therefore, it is most reasonable to integrate them with the interface's application.

The Motions block is used through the Controller block as well as the Planner block. It sends a command to the robot's actuators in order to make movements. However, those commands might control the chain of joints or the whole robot limb such as the legs or arms.

Finally, the Planner block is in charge of managing the schedule of the whole application for pHRI. Except for activating and deactivating the interface's application, the block is responsible for initiating the robot to the pHRI, e.g. the robot has to take an appropriate pose to start the interaction. Thus, the Planner uses the Motions block as is shown on the diagram.

4.2.3 Sequence diagram

The sequence diagram presented in Fig. 4.4 illustrates how particular blocks communicate with one another and which messages are transmitted. Moreover, the diagram has to be consistent with the use case diagram. Usually, there is a tendency to sketch a separate sequence diagram for each use case. In the presented system only three use cases are distinguished, in which two of them, i.e. activation and deactivation, are very simple. Therefore, all of them were placed in the same diagram.

Generally, the interface's application starts when the user either launches a proper program command or affects a certain robot sensor. Then the application communicates with the Planner in order to initiate interaction. As a result, the Planner sets the robot's interaction pose by means of the Motions module.

Supposing that the application is run, the user is allowed to operate it or to deactivate it. It has to be remembered that the software and hardware layers are tied together. Hence, in the first case, by operating the interface's application, it means that the user interacts with the robot.

The user acts with the application through exerting force on the robot arm. Thus, the Application block reacts by communicating with the Controller which evokes a proper reaction. Due to this request the Controller asks the Corrector module for information needed for robot steering. The information is correspondingly delivered through the Sensors and Filter blocks. Based on received messages from the Corrector module, the Controller makes an adequate decision about regulating robot walking. With respect to this decision the Controller sends a command to the robot actuators via the Motions module. Consequently, the robot moves accordingly to the user's expectations.

Deactivation of the application is conducted in the same way as the activation. The user informs the application about the intention of terminating the HRI. Then the Planner executes the termination procedures by means of the Motions module.

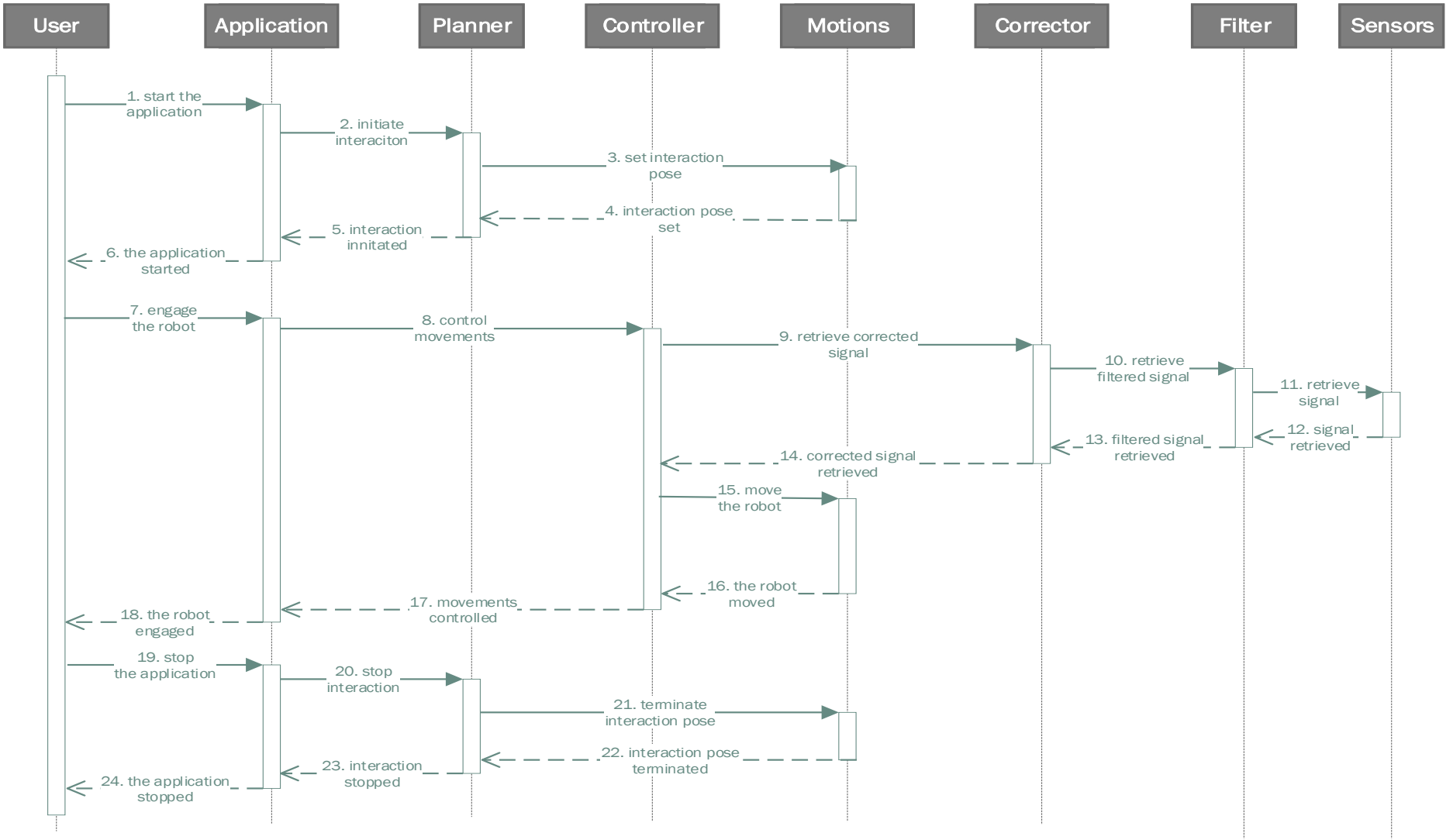


Figure 4.4: Sequence diagram of the designed interface

4.2.4 Activity diagram

Figure 4.5 presents the activity diagram of the designed interface. It illustrates the system from another perspective, i.e. system behaviour or dynamics.
First of all, the application can be utilised whenever the user wants to activate it. Then the application recursively checks if the user would like to interact with the robot. If there is a positive decision, a series of processes is conducted.

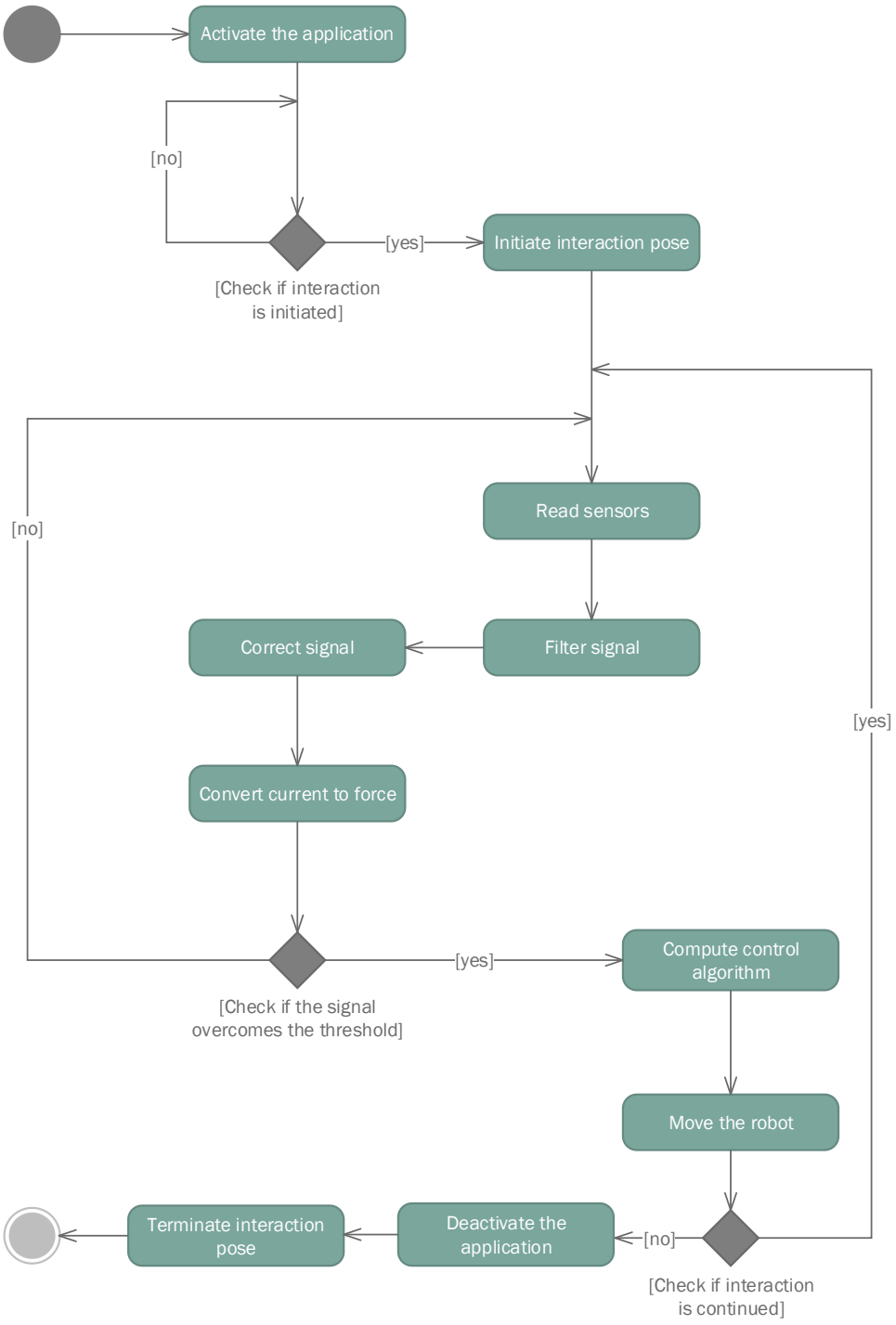


Figure 4.5: Activity diagram of the designed interface

Initially, the sensor signals are retrieved to the application module. Then signal conditioning processes such as filtering and correcting are conducted. After that the motor's current is converted to a corresponding force value. On the basis of this information it is checked whether the signal overcomes the threshold. The reason for this control is to prevent excessively sensitive robot reactions.

If the condition is fulfilled the control algorithm computes the properties for robot movement. Thus, the robot moves accordingly to the given parameters. If the user still tends to interact with the robot, the whole process repeats by reading the sensors. In other cases the user deactivates the application by terminating the interaction pose.

4.3 The algorithm design

To sum up the whole chapter, the design algorithm of the developed interface is proposed. The steps in the algorithm 1 are described with respect to the designer perspective. Thus, in order to implement the interface, a designer should make following steps:

Algorithm 1 The algorithm design for the developed interface

1: Determine the proper position of the robot arm

2: Implement position control for joints of the arm

3: Implement a filter for the signals carrying useful information

4: Implement a signals corrector with respect to other factors, such as gravity term or regulation error

5: Implement conversion rate for the motor's current and corresponding force value

6: Implement the control algorithm

5. Implementation of the interface

This section discusses implementation of the designed interface on the NAO platform. According to the design method presented in subsection 4.3, defined steps had to be carried out in order to apply the proposed solution. However, performing some of these steps was slightly limited due to framework restrictions.

First of all, the NAO's mechanical architecture imposes how the robot arm's position is set. The robot is short and, therefore, has to straighten out its arm to the vertical position in order to be guided. Only in this configuration the user is able to normally interact with the robot during the walking phase.

Secondly, there was no proper equipment available to conduct force measurement during the implementation. In effect, the exact force values corresponding to the motor current could not be found. Nevertheless, it will be suggested in the further parts of the chapter how the armature current and torque are related and how such measurement can be conducted.
The other implementation steps are explained in the corresponding subsections.

5.1 Implementing the positional control

As was presented in Chapter 3 describing the NAO platform, the manufacturer has already equipped the robot's joints with position control [21]. Therefore, it is not necessary to conduct such a step. Nonetheless, the general concept of implementing position control is explained below, also because it is used later in the work.

At the beginning, the controlling object has to be identified in order to implement a position control system. Assuming that the parameters of the motor and its conceptual scheme are known – Fig. 5.1, a mathematical model of the DC motor can be formulated [16].

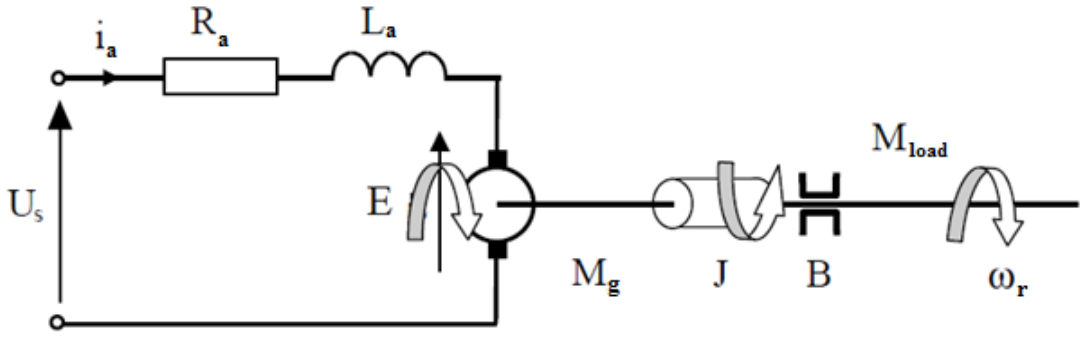


Figure 5.1: General model of the DC motor [27]

where:

U_s	denotes supply voltage	B	denotes rotational viscous friction
i_a	is armature current	J	is mass moment of inertia of the motor armature
R_a	represents armature resistance		
L_a	symbolises armature inductance	T_{load}	presents load torque
E	is back EMF	k_e	is electrical constant
ω_r	presents rotor velocity	k_m	denotes mechanical constant
T_g	is generated torque		

Based on Kirchhoff's second formula, the following equation can be formulated:

$$U_s = U_R + U_L + E \tag{5.1}$$

where:

U_R is the voltage drop on the armature resistance, which is proportional to the armature current:

$$U_R = R_a\, i_a \tag{5.2}$$

U_L is the voltage drop on the armature inductance, which is proportional to changes of the armature current in time:

$$U_L = L\frac{dIa}{dt} \tag{5.3}$$

E is the back EMF voltage, which is a function of the motor's angular velocity:

$$E = k_e\, \omega_r \tag{5.5}$$

In effect, the electrical equation of the DC motor is in the following form:

$$U_s = R_a\, i_a + L_a \frac{dIa}{dt} + k_e\, \omega_r \tag{5.6}$$

On the other hand, analogically to the electrical consideration (5.1), the account dynamics of the system may be taken into account. Thus, the mechanical equation of the DC motor by means of Newton's second law may be formulated:

$$T_g = T_a + T_v + T_{load} \tag{5.7}$$

where:

T_g is generated torque, which is proportional to the armature current (assuming a constant magnetic flux of the stator):

$$T_g = k_m\, i_a \tag{5.8}$$

T_a is torque related to rotor acceleration:

$$T_a = J\frac{d\omega}{dt} \tag{5.9}$$

T_v is torque related to the resistance of motion:

$$T_v = B\omega_r \tag{5.10}$$

By substituting corresponding elements in equation (5.7), the mechanical equation of the DC motor is in the following form:

$$k_m i_a = J\frac{d\omega}{dt} + B\omega_r + T_{laod} \tag{5.11}$$

Assuming zero initial condition for equations (5.6) and (5.11), they may be transformed to the Laplace domain and written as:

$$U_s(s) = R_aI_a(s) + L_aI_a(s)\cdot s + k_e\omega_r(s) \tag{5.12}$$

and

$$k_m I_a(s) = J\omega_r(s)\cdot s + B\omega_r(s) + T_{load} \tag{5.13}$$

Figure 5.2 illustrates the block diagram combining equations (5.12) and (5.13) with the assumption that $T_{load} = 0$.

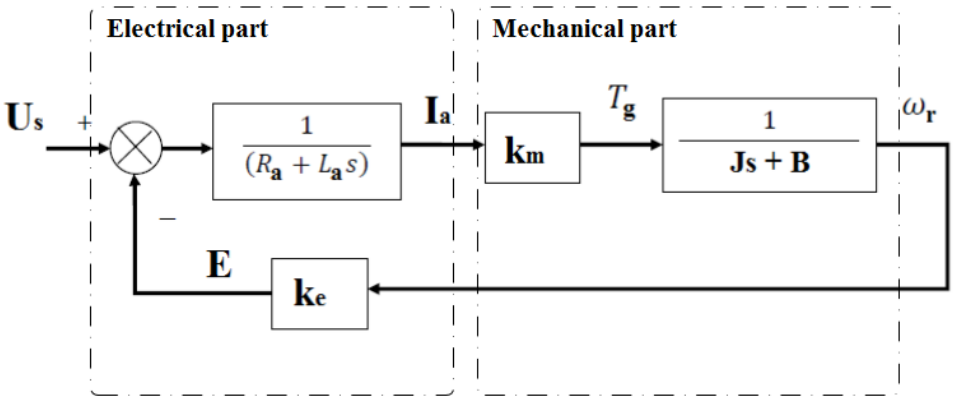


Figure 5.2: Block diagram of the DC motor

Supposing the DC motor is a SISO system, where the rotor velocity is the output signal and the supply voltage is the input, in effect the system as presented above might be simplified to the single transfer function:

$$G_{dc}(s) = \frac{U_s(s)}{\omega_r(s)} = \frac{km}{(sLa+Ra)(sJ+B)+ kmke} \tag{5.14}$$

or

$$G_{dc}(s) = \frac{U_s(s)}{\omega_r(s)} = \frac{k_m}{JL_a s^2 + (R_a J + BL_a)s + R_a B + k_m k_e} \quad (5.15)$$

In other words, the DC motor can be modeled by a second-order transfer function.

Finally, such a model is utilised in position control where the reference trajectory is compared with the present position of the motor's shaft and is then introduced to the regulator. In order to retrieve the position of the rotor instead of its velocity, the system output has to be integrated. In reality, the position is measured by means of an encoder mounted partially on the motor shaft.

Figure 5.3 illustrates the diagram of position control. Transfer functions of the regulator, DC motor and encoder as an integral term are marked.

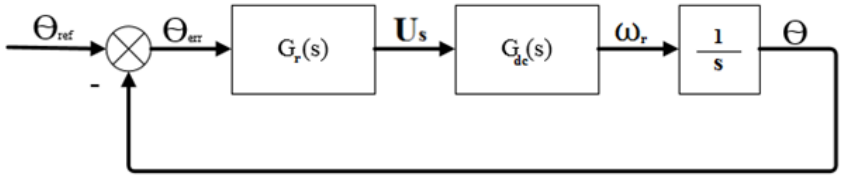


Figure 5.3: Block diagram of position control

where:

G_d is transfer function of the DC motor, G_r signifies transfer function of the position controller.

Assuming that the digital regulator adjusts the duty cycle of the motor's supply voltage, a position control algorithm can be implemented such as the following:

Algorithm 1 The proportional position control algorithm

1: Initiate variables for storing control signal, reference position, present position, position error, regulator parameter - ctrl_sig, q_ref, q_pres, q_err, kp

2: Set reference position and regulator parameter - q_ref, kp

3: Retrieve information about present position of motor's shaft - q_pres

4: Calculate position error - q_err = q_ref - q_prest

5: Calculate control signal based on position error - ctr_sig = kp*e_err

6: Apply control signal to the motor

* 4-6 steps are recursively repeated.

The reason of implementing position control in the developed interface results from the equation (5.8). The motor current and generated torque are directly related to each other. It is also known that torque is the cross-product of a force-producing rotation and lever-arm distance:

$$T = F \times r \tag{5.16}$$

Therefore, the relation between the force and the motor current can also be deduced. According to equation (5.12), the motor current is also in a relationship with the supply voltage regulated by the position controller. Hence, another conclusion can be made.
If a force/torque is exerted on the motor's shaft, it causes its position displacement. Moreover, due to increased position error, the regulator, in turn, enhances the supply voltage and the motor current.

On the other hand, based on Newton's first and third laws it might be observed that if the force exerted on the shaft and the force generated by the motor compensate each other, then the shaft is at rest or moves at a constant velocity. In addition, the force generated by the motor is of the same magnitude as the force exerted on the motor's shaft, but it is opposite in direction.

In conclusion, such a designed system allows to relate the armature current with the force exerted on the motor's shaft and, in effect, on the robot arm.

5.2 Signal filtering

According to the next step of the designing method as presented in subsection 4.3, in order to apply an interface for pHRI the signal carrying useful information has to be transparent and understandable. Therefore, the motor currents were examined to verify the above-mentioned criteria. Figure 5.4 presents the motor current as a function of time. The signal is derived from the NAO's shoulder joint which was placed in the vertical position, i.e. along the length of the body.

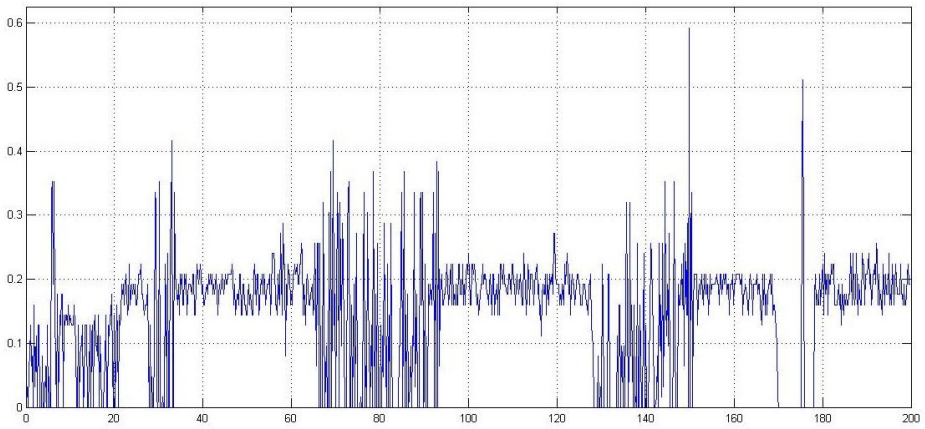


Figure 5.4: Motor current as a function of time - unfiltered

Based on the obtained measurement it can be seen that the signal was distorted and illegible. It was deduced that the signal had been disturbed by high frequency components, whose sources could have been phenomena such as system damping or electromagnetic interference. Therefore, it was decided to process the signal by means of a low-pass filter. For benchmark purposes two kinds of low-pass digital filters were tested, i.e. a mean filter and a simple moving average filter. The principles of operation for each of them are presented below:

The mean filter retrieves and averages M-measurements. Then the result of those calculations is assigned to the filter's output in the n-moment:

$$y_n = \frac{1}{M}\sum_{j=1}^{M} \mathbf{X}_M \tag{5.17}$$

In contrast to the mean filter, the moving average filter does not buffer data. It is just the unweighted mean of the previous M measurements:

$$y_n = \frac{1}{M}\sum_{j=0}^{M-1} \mathbf{X}(n\text{-}j) \tag{5.18}$$

where:

y_n is the output value of the filter in the n-moment, M denotes length of the filtering window.

On the basis of the conducted benchmark for mean and moving average filters it was concluded that the mean filter is a much better solution for this application. Therefore, the motor current signal with a filtering window M = 20 was employed. The result of this choice is illustrated in Fig. 5.5.

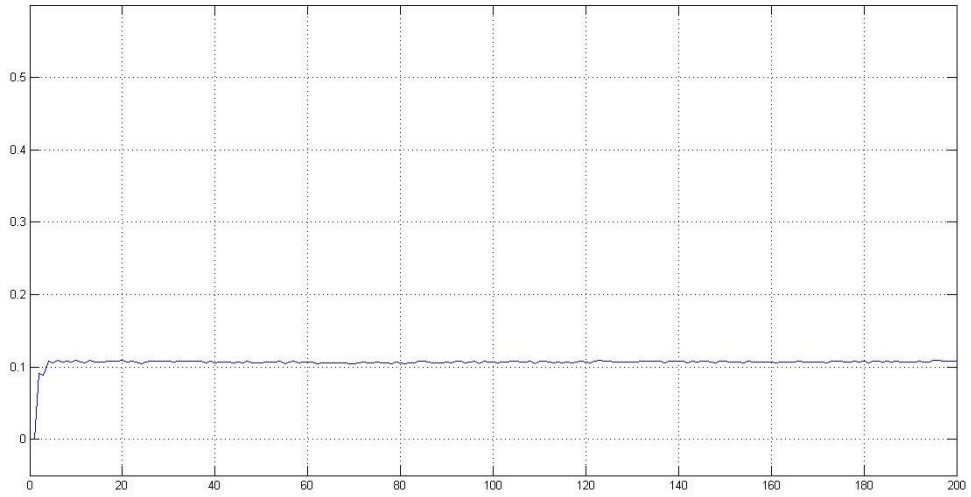


Figure 5.5: Motor current as a function of time - filtered by a mean filter

5.3 System identification

Subsection 5.1 discussed the general concept of the position control that was applied in the NAO's joints. However, the lack of controller specification involves the additional risk that the proposed interface will not be applicable to the NAO framework. Therefore, system identification (SI) was conducted to ensure whether it matched the methodology of the interface design.

5.3.1 Introduction to SI

In general, a system can be defined as part of the real world which can interact with its environment. It can be described by input u, output y and disturbance e, which is presented in Figure 5.6. System identification allows us to build the mathematical model of a system based on measured data.

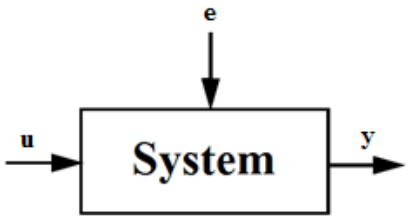


Figure 5.6: Conceptual model of a system

Building the analytical model of a complex system might be very problematic. Therefore, the model usually constitutes a certain idealisation that captures essential information about the real system.

The identification methods can be grouped according to various criteria, for instance, parametric and non-parametric criteria [28].

- Parametric identification methods are techniques used to estimate the parameters of a determined model structure. In general, the aim is to numerically find such parameters that fit the model and system outputs for given input data.
- The non-parametric identification method can provide basic information about a system, such as time delay, time constant, etc. The results of identification are usually non-parametric models such as curves or tables.

Another grouping based on *a priori* knowledge of the system is the following:

- White-box identification estimates the parameters of the physical model.
- Grey-box identification estimates the parameters of a given generic model structure.
- Black-box identification determines the model structure and then estimates its parameters.

On the whole, the system identification procedure can be illustrated by Fig. 5.7.

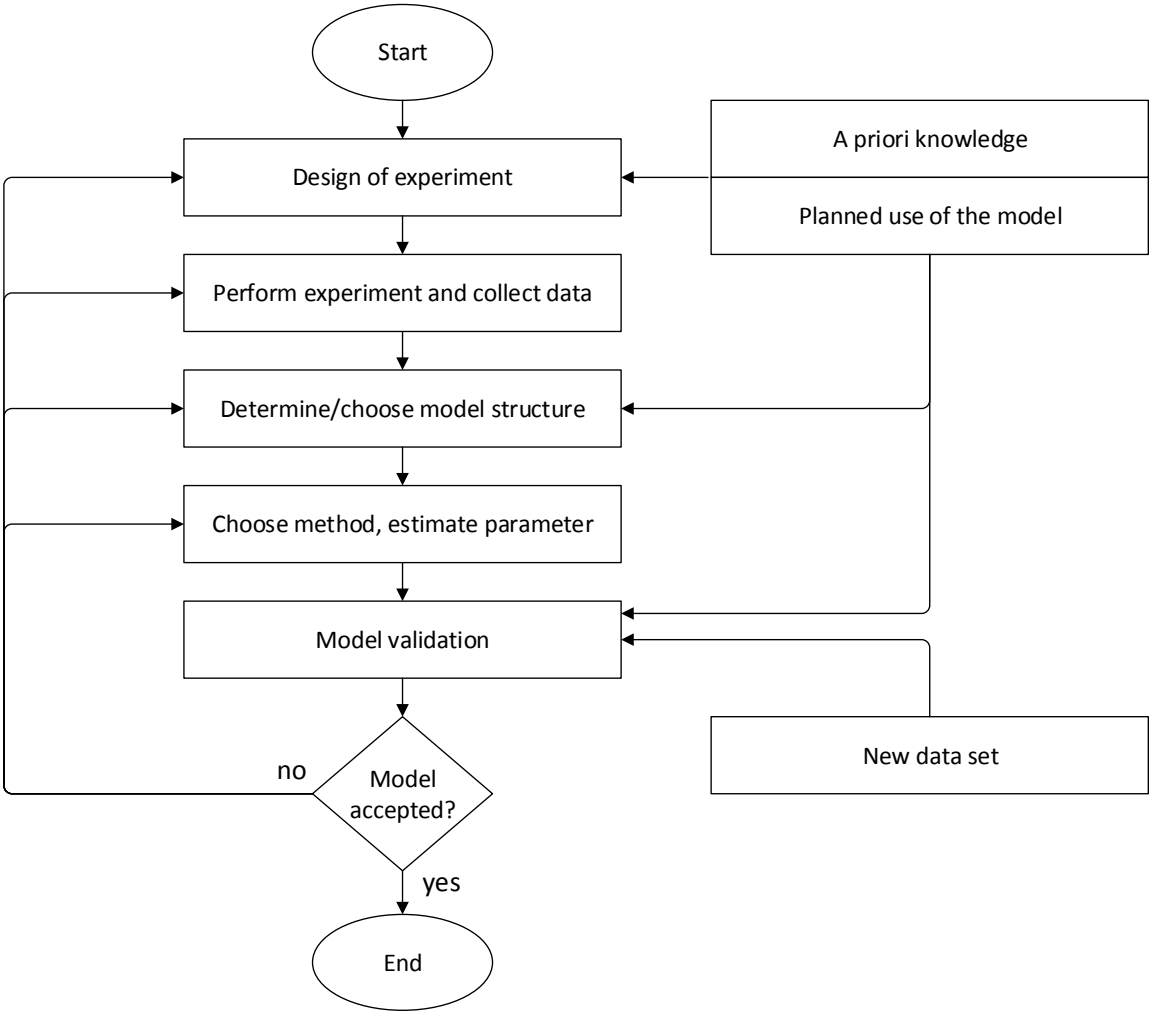


Figure 5.7: System identification procedure [29]

In order to design a proper identification experiment, *a priori* knowledge about the system is required. Unless the nature of the system is known, it is recommended to investigate the system. This system research might contribute significant information to further works.

The identification experiment should be conducted on representative data. Therefore, the collected data ought to be maximally informative. This can be done by reducing the impact of disturbance.

The next and most difficult step is to choose or determine the model structure. Apart from *a priori* knowledge of the system, the decision is based on the experience and intuition of the designer. After that the parameters of the model are estimated on the basis of an identification algorithm. In effect, the best model within the model structure is chosen.

Finally, the obtained model is validated based on data other than that captured during the identification experiment. If the model is good enough that it fits the data, then system identification is carried out. Otherwise the steps should be repeated.

5.3.1 System investigation

System investigation is aimed at inspecting whether the motor current can be utilised for the interface design. Therefore, we conducted experiments to determine the impact of various factors on the current value.

5.3.1.1 Motor current as a function of robot arm position

The first investigation examined the influence of the robot arm's position on the motor current value. Each measurement started by placing the robot arm at its initial position A. Then, the robot raised its arm to a specified position as shown in Fig. 5.8.

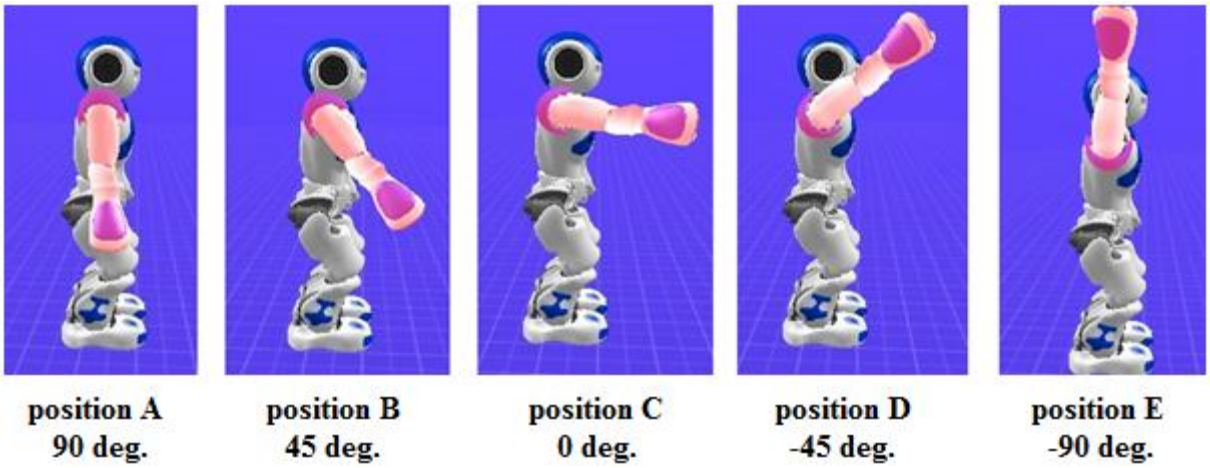


Figure 5.8: Positioning of the robot arm

The experiment was conducted for five positions of the robot arm (A - E). In addition, measurement for a particular arm position was repeated many times. The reason for this was the ambiguity in the received results. Thus, Figures 5.9-5.18 below present two measurements for the same position condition to illustrate the difference between them. The number in the square bracket of the Figure caption denotes the moment when the robot raised its arm.

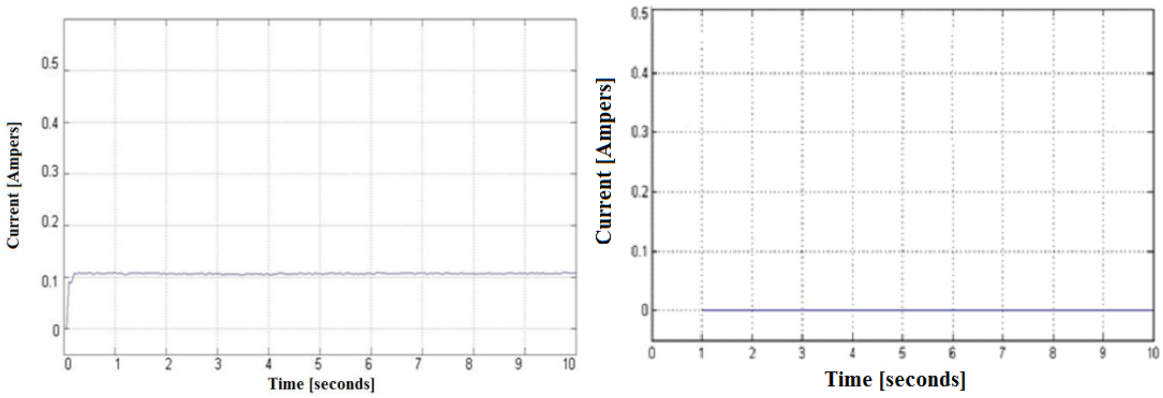


Figure 5.9, 5.10: Current as a function of time - position A

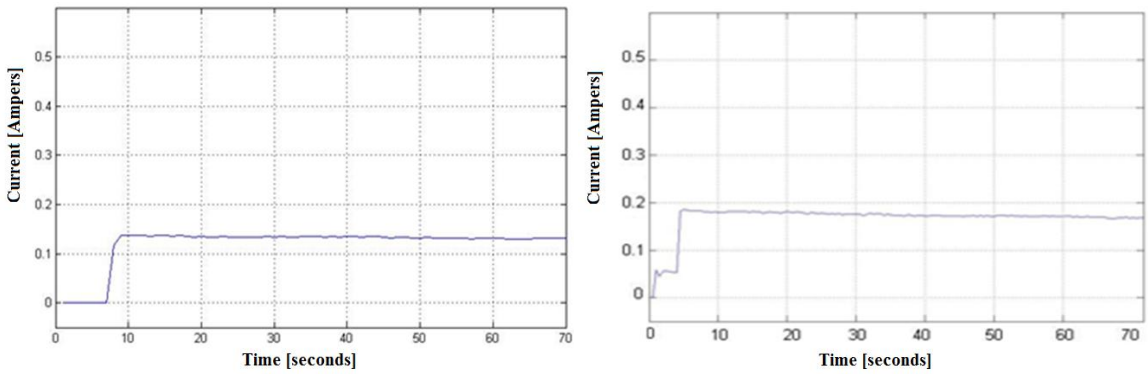


Figure 5.11, 5.12: Current as a function of time - position B [8th and 3rd sec.]

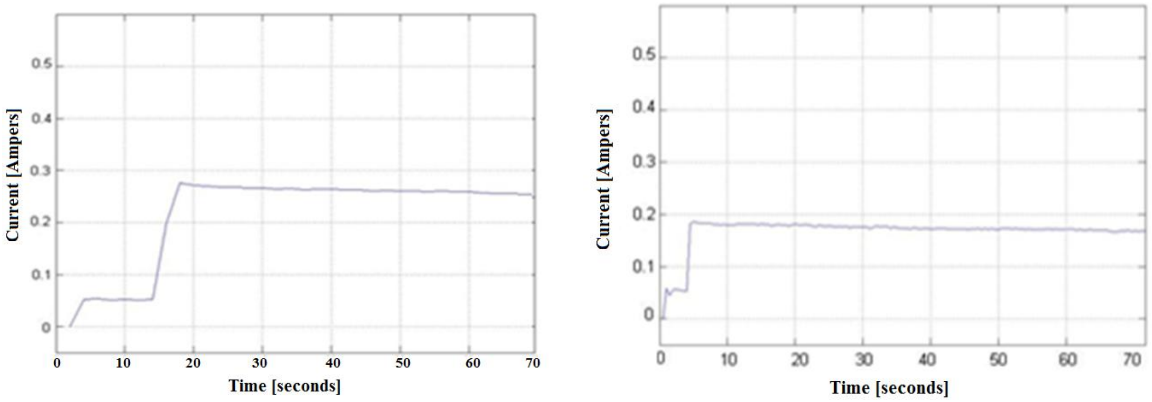


Figure 5.13, 5.14: Current as a function of time - position C [14th and 3rd sec]

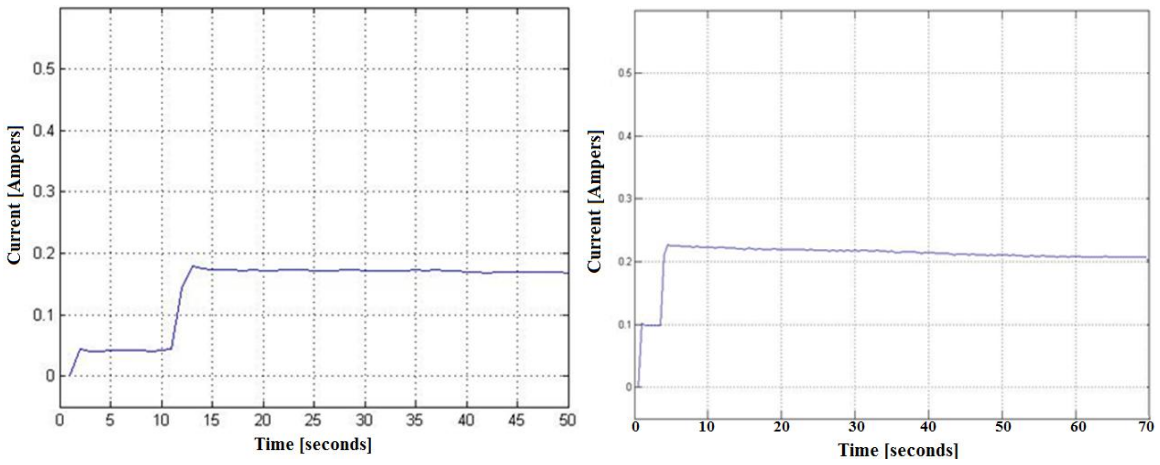


Figure 5.15, 5.16: Current as a function of time - position D [11th and 3th sec.]

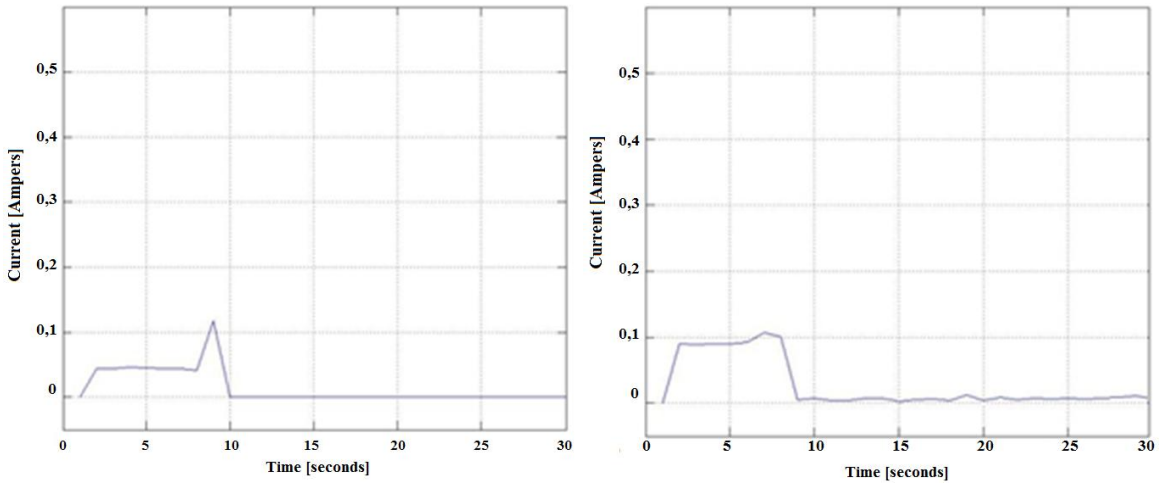


Figure 5.17, 5.18: Current as a function of time - position E [7th and 5th sec.]

According to the assumptions, the regulator controls the motor's voltage based on the reference and present position of the robot arm. Thus, unless the robot's arm achieves the expected position, the regulator increases the motor's voltage. In effect, the motor current increases.

On the other hand, the figures illustrate a difference in the obtained results for the same conditions. It seems that the regulator has some inaccuracy. During the experiment the regulator placed the robot's arm at a slightly different position than was expected, i.e. +/- 3 degrees. Nevertheless, the range of those variations can be roughly estimated.

5.3.1.2 The relationship between motor current and position error

Based on the obtained results, the relation between motor current and position error was concluded. Therefore, the second experiment was performed to investigate this dependency.

The robot arm was initially set at position A. This configuration minimised the gravity torque that was exerted on the robot's arm. In effect, the position error was insignificant. After that the sequence of human-robot interaction was performed, such as pulling and pushing the robot arm.

In general, three different scenarios of the experiment were carried out. The position's error and motor current were recorded for each scenario. The description and figures for particular scenarios are presented below.

The first scenario:

15 sec - pulling the arm	50 sec - releasing the arm
20 sec - releasing the arm	70 sec - pushing the arm
40 sec - pulling the arm	82 sec - releasing the arm

Table 5.1: Description of the first scenario

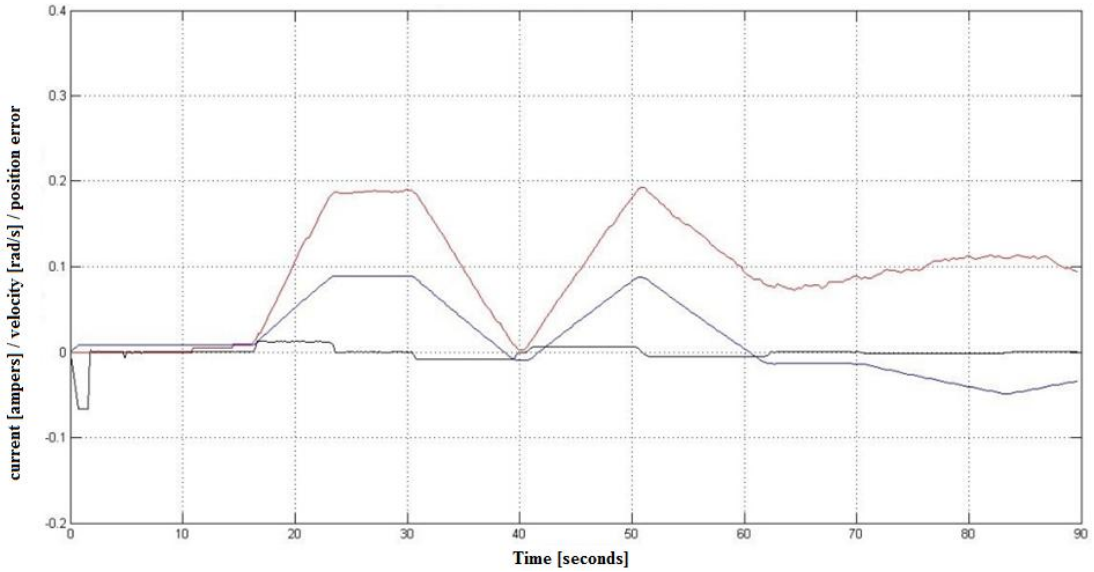


Figure 5.19: Characteristics of motor current, position error and velocity - 1st scenario. *red line* is motor current, *blue line* denotes position error, *black line* is velocity of the robot arm.

The second scenario:

10 sec - pushing the arm	72 sec - releasing the arm
30 sec - releasing the arm	100 sec - pulling the arm
60 sec - pushing the arm	

Table 5.2: Description of the second scenario

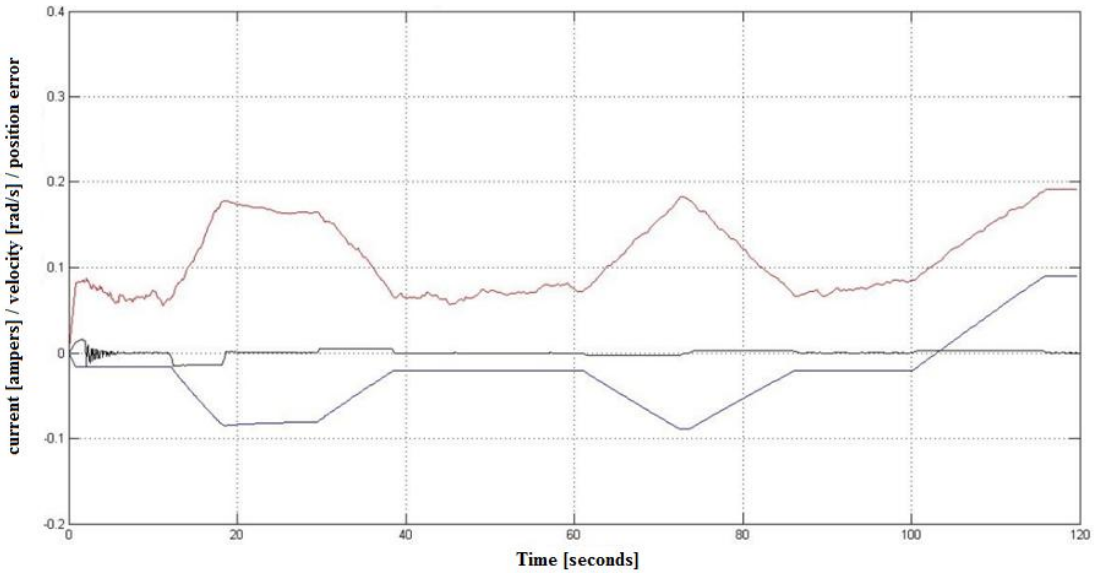


Figure 5.20: Characteristics of motor current, position error and velocity - 2nd scenario. *red line* is motor current, *blue line* denotes position error, *black line* is velocity of the robot arm.

The third scenario:

10 sec - pushing the arm	60 sec - pulling the arm
20 sec - releasing the arm	70 sec - releasing the arm
40 sec - pushing the arm	90 sec - pushing the arm
50 sec - releasing the arm	100 sec - releasing the arm

Table 5.3: Description of the third scenario

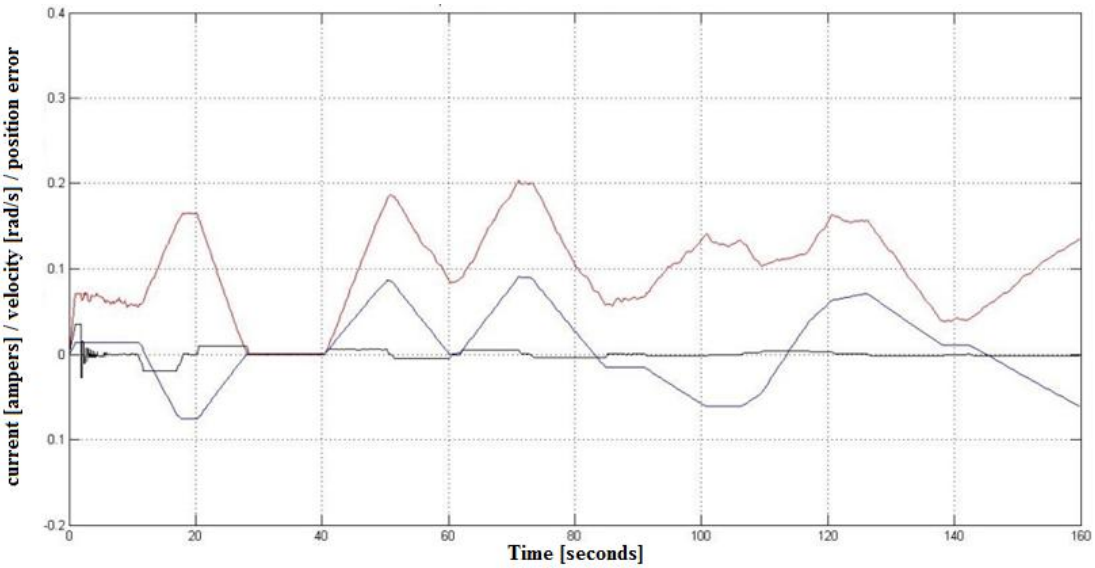


Figure 5.21: Characteristics of motor current, position error and velocity - 3nd scenario. *red line* is motor current, *blue line* denotes position error, *black line* is velocity of the robot arm.

The results of the second experiment Fig. 5.19 - 5.21 show that a relationship between the absolute value of the position error and motor current takes place. The larger the position error is, the higher the value of the current flows through the motor. In addition, an impact of the position error's growth rate on the motor current was observed. The reason for this could have been the derivative component of the position controller.

5.3.1.3 Motor current as a function force applied to the robot arm

The next investigation examined the relationship between the force exerted on the robot arm and the motor current. The robot arm was placed at position C (subsection 5.3.1.1), and then data acquisition started. Starting at the 20th second the arm was loaded by an additional weight attached to the robot's wrist. After 40 seconds the load was relieved until the 65th second. Then again the robot arm was loaded with the same weight.

The experiment was conducted with varied weight of the load, i.e. from 0 to 600 grams, with 100-gram steps. Due to the ambiguity of the obtained results, each measurement was repeated several times - Fig. 5.22-5.35. In addition, measurement without a load was recorded with additional information about the motor's temperature.

The Figures below illustrate the motor current and position error in the function of time for different configurations of the loaded arm. The red line corresponds to the motor current, the blue line signifies position error. The green line denotes the motor's temperature for the characteristic without external load. The temperature on the plot is scaled by dividing the real temperature by 200. The physical magnitude is given in Celsius units.

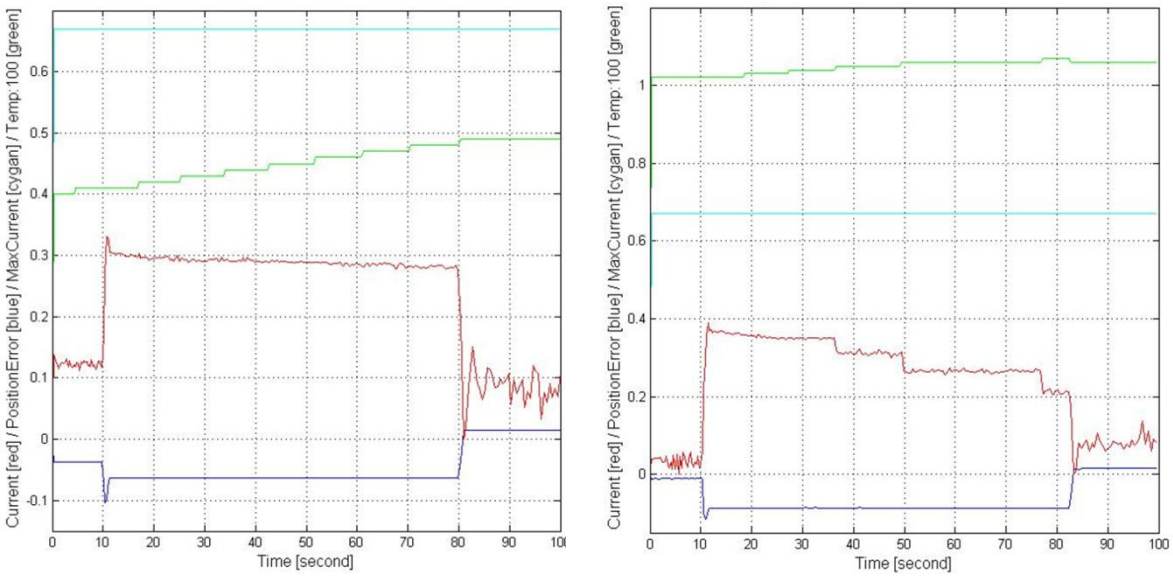


Figure 5.22, 5.23: Current as a function of position error - robot arm without load

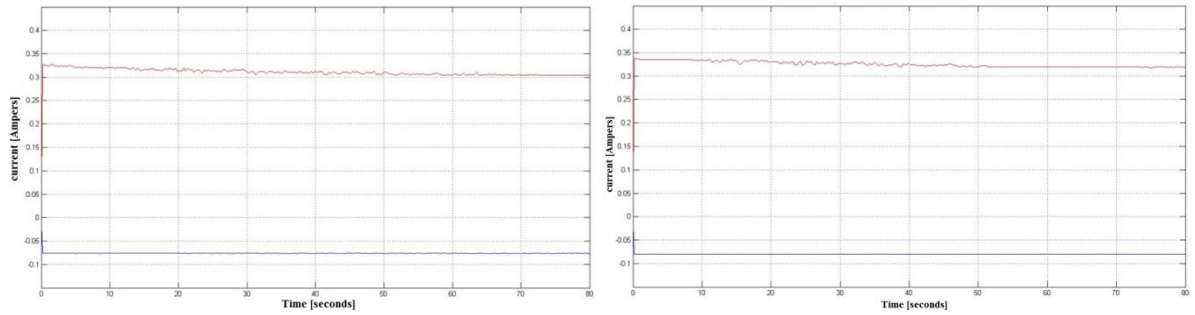


Figure 5.24, 5.25: Current as a function of position error - robot arm loaded with 100 g

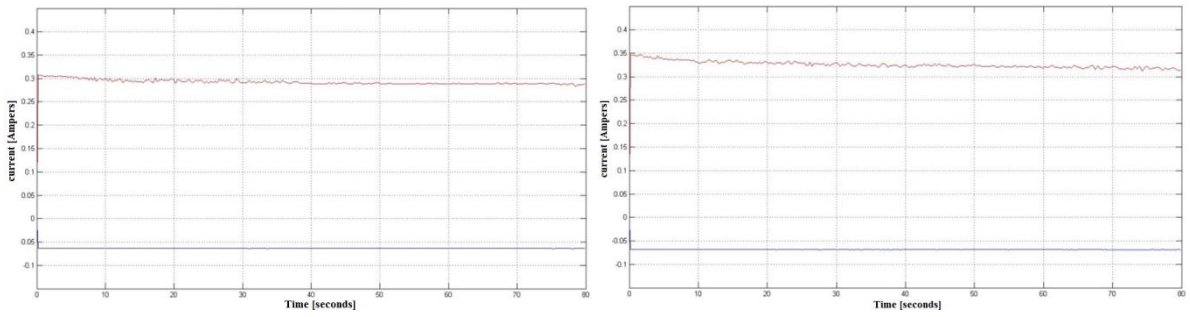


Figure 5.26, 5.27: Current as a function of position error - robot arm loaded with 200 g

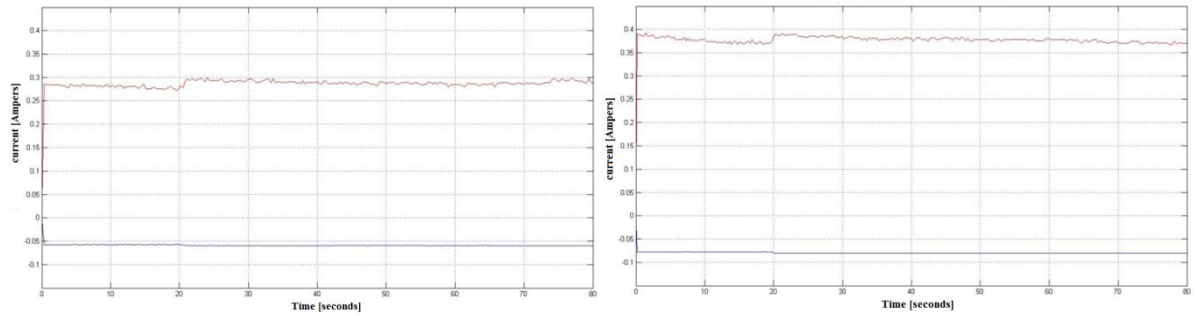


Figure 5.28, 5.29: Current as a function of position error - robot arm loaded with 300 g

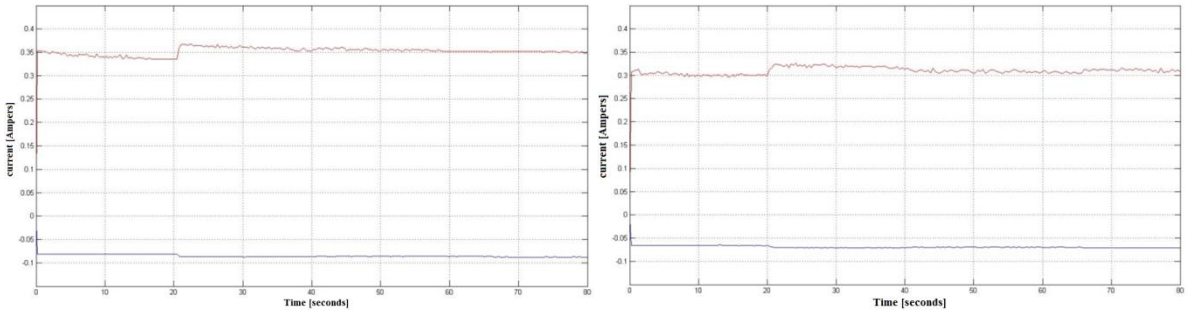


Figure 5.30, 5.31: Current as a function of position error - robot arm loaded with 400 g

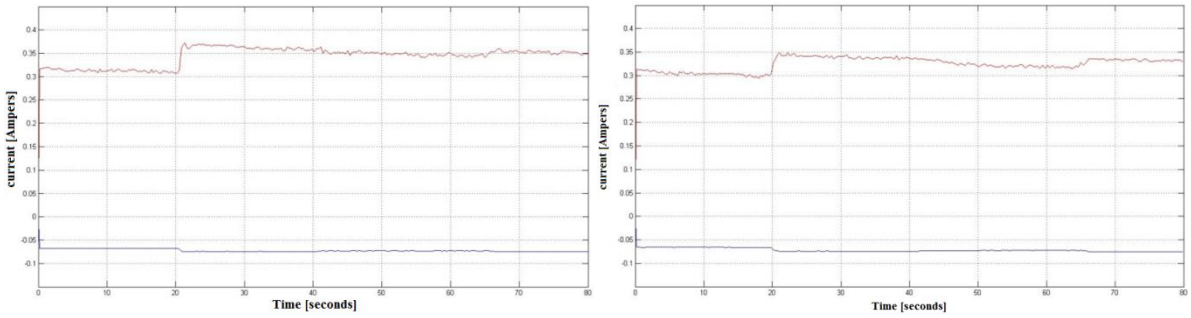


Figure 5.32, 5.33: Current as a function of position error - robot arm loaded with 500 g

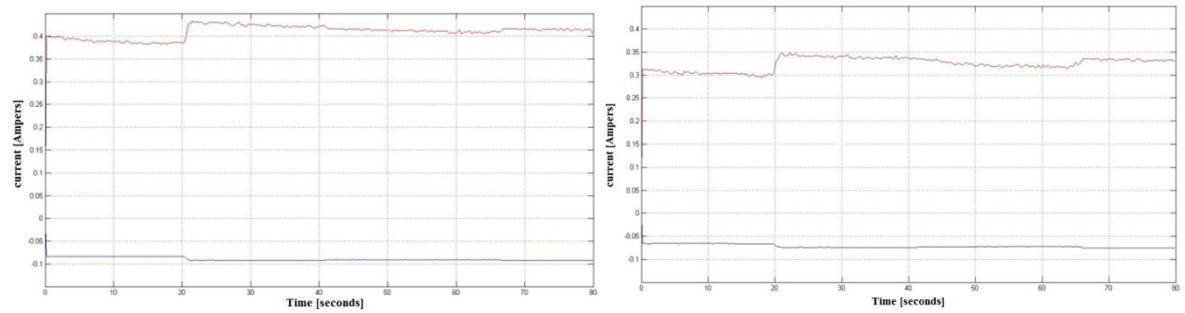


Figure 5.34, 5.35: Current as a function of position error - robot arm loaded with 600 g

The results of the measurements illustrate that the force exerted on the robot arm implies changes in the position error and motor current. The value of the motor current is within the range of 0.3-0.6 A. Thus, the larger force applied to the robot arm, the bigger the motor current is. In addition, the force that was exerted on the robot arm permanently displaced it. Once the load had been removed, the arm did not return to the initial position. Moreover, it was held in the new equilibrium point. When the robot arm was loaded again, the arm did not displace at all.

Another observation is that the same value of force applied to the robot arm implies a slightly different increase of the motor current value. Nonetheless, the range of those values might be estimated for a respective force; for instance, once the robot arm was placed in the horizontal position and weighted by 400 g, the increase of the motor current value was within the range of 0.025-0.05 A. It is believed that this given ambiguity derives from the fact that the instantaneous value of the applied forces was dissimilar. Moreover, the increase of the motor current depends on the maximum value of the instantaneous force in a transient state. Due to the lack of proper measuring instruments, the conclusions could not be verified.

The differences in the motor current value for the same position of the robot's arm without load were much higher than the increase of the motor current caused by the applied force. Therefore, the relationship between the motor current value and the force applied to the robot arm could not be directly determined.

5.3.1.1 The temperature effect

It is believed that the motor's temperature was another factor that could affect the measurements. Thus, another experiment was conducted to investigate this. Figure 5.36 illustrates the position error, motor current and motor temperature. During the experiment the robot raised its arm to position C (subsection 5.3.1.1) and held it for 20-50 sec. Next, the robot lowered its arm. These movements were repeated many times. Figure 5.36 presents the obtained results.

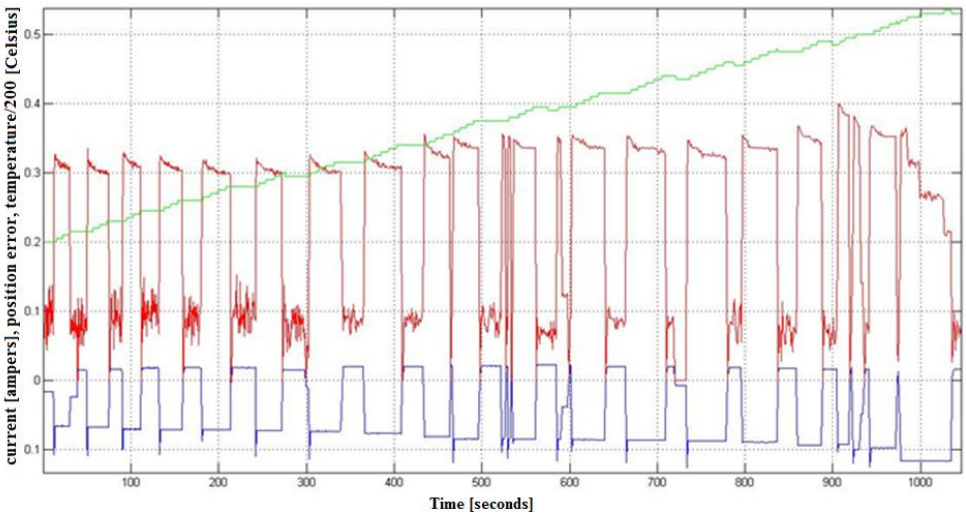


Figure 5.36: Temperature effect in the NAO's motor. *red line* denotes motor current, *blue line* is position error, *green line* represents motor temperature.

As it can be seen, the same settings for the robot arm resulted in varied values of position error and motor current. Motor temperature was one of the factors that could have an impact on the inaccurate motor control.

In [30] the impact of electrical and mechanical constants on the dynamic response of the DC motor is discussed. The constants are the functions of magnetic flux density produced by the motor's magnets. In contrast to the nomenclature, both "constants" vary with the temperature:

$$k_{e,\,m}(T) = k_{e\,,m}(T_0)[1\text{-}B(T\text{-}T_0)] \tag{5.19}$$

where:

k_e denotes electrical constant, k_m signifies mechanical constant, T_0 represents cold temperature, T is winding temperature, B symbolises coefficient of magnet material

Besides, the motor's resistance is a function of the winding temperature. Thus, the electrical and mechanical time constants also change correspondingly:

$$t_e = \frac{L}{R(T)} \tag{5.20}$$

$$t_m = \frac{R(T)J}{ke(T)km(T)} \tag{5.21}$$

where:

t_e is electrical time constant, t_m represents mechanical time constant, R signifies motor's resistance, L denotes motor inductance, J is total inertia at the motor.

As is shown, an increase in the motor's winding causes a decrease in the electrical time constant and increases the mechanical time constant. Hence, the dynamic response of the motor changes as well.

5.3.2. Identification Process

The identification experiment was designed based on *a priori* knowledge of the system structure (Subsection 4.1) as well as system investigation. Thus, the conceptual structure of the identified system can be presented in Fig. 5.37. This system identification was carried out by means of position error and motor current. Hence, the obtained model describes the relationship between these two quantities.

Basically, the interface design methodology restricted the choice of collected data, i.e. of motor current and position error. The identification aimed to examine if the system could be utilised according to the design assumptions.

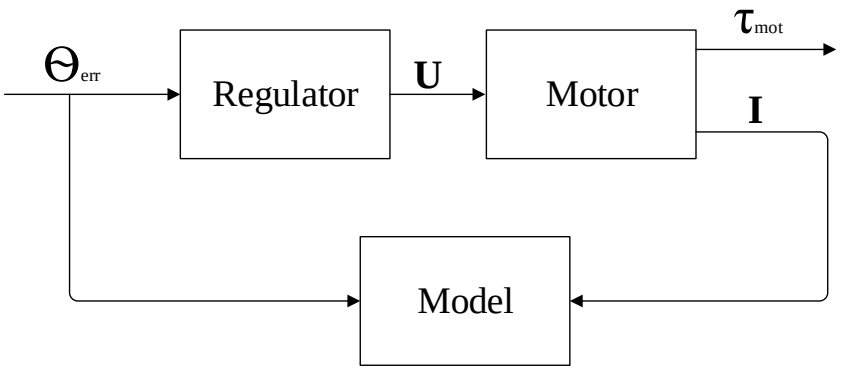


Figure 5.37: Conceptual structure of the identified system

where:

Θ_{err} is position error, **U** is motor's supply voltage, $\boldsymbol{\tau}_{mot}$ denotes motor's force/torque exerted on the robot arm, **I** signifies motor current.

The system identification was conducted using a Matlab application and the software presented in subsection 3.

By and large, the system model strictly depends on the quality of data that are utilised during the identification process. Thus, the data of the motor current and position error were recorded in real time. The Matlab application delayed data acquisition due to the necessity of having to connect with the robot by TCP/IP protocol. Moreover, the sampling frequency was surely low, that is, 7 Hz. The high-frequency component of noise, such as the electrical grid or the mechanical damping, could not be adequately sampled. The sampling frequency had to be increased based on the Nyquist–Shannon theorem. Therefore, the data acquisition program was compiled and executed directly on the robot. In effect, the sampling frequency was augmented to 150 Hz.

Data collection began after the robot had placed its arm at position A (subsection 5.3.1.1), i.e. along the robot's body. Next, the robot operator rapidly pulled up and, in effect, displaced the robot arm. The operator constantly held the arm until the end of data acquisition. The experiment aimed to acquire system behaviour that would be similar to the step response. The collected data were converted to Matlab file format. Then, system identification was conducted using the Matlab System Identification Toolbox [28].
The first step was to determine the identification method. The model structure was expected to be a second-order transfer function (subsection 5.1); therefore, the parametric identification

method was employed. Matlab offers such identification by means of Process Models. The designer has to determine the structure of the continuous-time model by:

- K_p - static gain
- T_z - possible process zero
- T_d - possible time delay (dead time)
- possible number of poles
- T_{pk} - one or several time constants (such as T_w or ζ – the damping parameter)
- a possible integral term

As a result, several different model structures were examined and compared. The result is presented in Fig. 5.38.

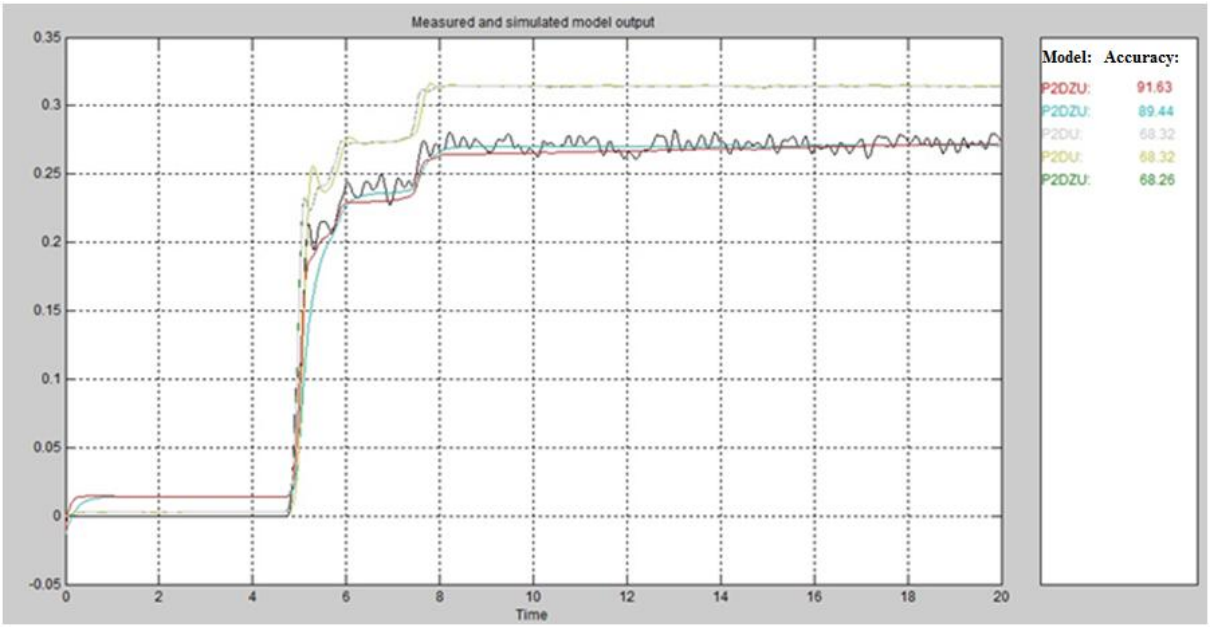


Figure 5.38: Comparison of particular models

red line - the best model structure (chosen model), *black line* - real system

As is shown in Fig. 5.38 above, the best model structure describes the real system with 92% accuracy. The transfer function of this structure is in the following form:

$$G(s) = \frac{K_p(1+T_{z*}s)}{(T_{w*}s)^2+T_{w*}s*\zeta*2+1} \tag{5.22}$$

The Matlab application imposed the method for parameter estimation, thus, it utilised the Prediction Error Minimization method [29]. Generally, PEM aims to optimise the cost function:

$$V_N(G, H) = \sum_{t=1}^{N} e(t)^2 \tag{5.23}$$

where:

N is number of data, G denotes system model, H signifies disturbance model, *e(t)* is the difference between the measured and the predicted output of the model.

For a linear model, this error is defined by the following equation [28]:

$$e(t) = H(q)^{-1}[y(t) - G(q)u(t)] \tag{5.24}$$

In order to validate the obtained model, the real system and the model responses were compared with the given input - Fig. 5.39.

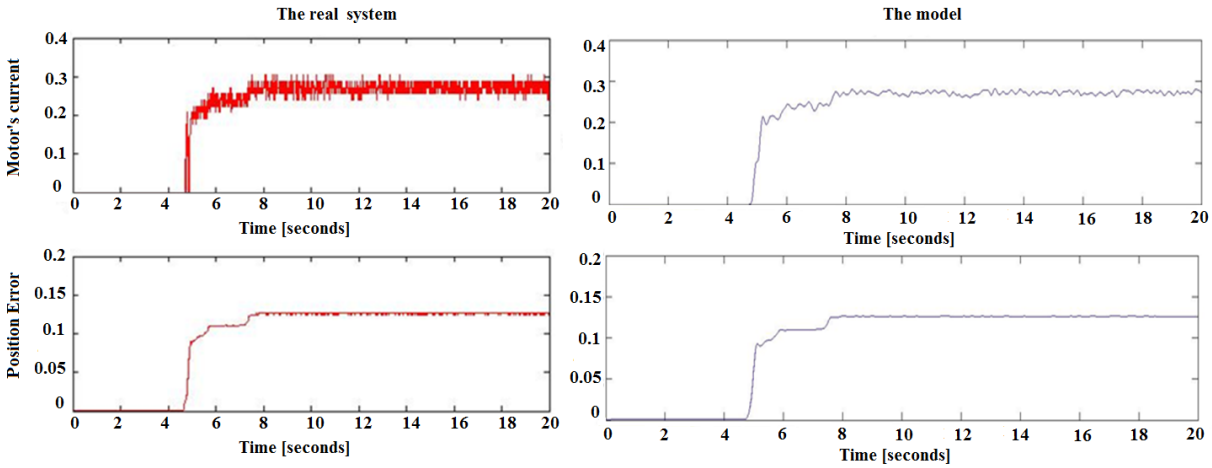


Figure 5.39: Comparison of model and system responses for given input

The above identification procedure was repeated three times due to the fact that the three robot arm joints were engaged to the interface design. Thus, the parameters for corresponding models are presented in Tab. 5.4.

	ShoulderPitch	**ShoulderRoll**	**ElbowRoll**
$\mathbf{K_p}$	2.5393	2.2876	2.1796
$\mathbf{T_z}$	56.152	0.021461	0.48916
$\mathbf{T_w}$	2.1787	0.016406	0.043098
Z	16.501	0.10714	5.8713

Table 5.4: Parameters of the identified systems

The first row presents the names of the robot's joints related to the obtained models. The left column contains a list of parameters describing a particular model in the form of (5.22).

To sum up, the system identification process resulted in obtaining a mathematical model which described the relationship between the position error's module and motor current. The model structure can be formulated by the second-order system with two underdamped poles

and one zero (5.22). Moreover, the identification process and the system investigation (subsection 5.3.1) confirmed the fact that the interface design methodology (section 4) can be applied on the NAO platform.

5.3.3 Signal correction

The system investigation examined the relationships between the position error, the motor current and the force applied to the robot arm. The experiments confirmed the usefulness of the motor current for force determination. However, due to the ambiguity in measurements as described in subsection 5.3.1, some signal correction was necessary.

In general, the force applied to the robot arm was identified on the basis of the motor current increase instead of the current value. Several different reasons could have impacted this, for instance, temperature effect or position system inaccuracy.
Basically, once the robot had received a command to place its arm in a particular position, a systematic position error always occurred. As a result, a systematic error of the motor current was implied as well.

The intention was to correct the motor current signal to obtain a direct relationship between the current value and the applied force. Therefore, the corrective signal reduced the systematic error of the motor current based on position error information. Consequently, the relationship between the refined value of the motor current and the applied force was obtained.
Figure 5.40 illustrates the principle of the motor current correction.

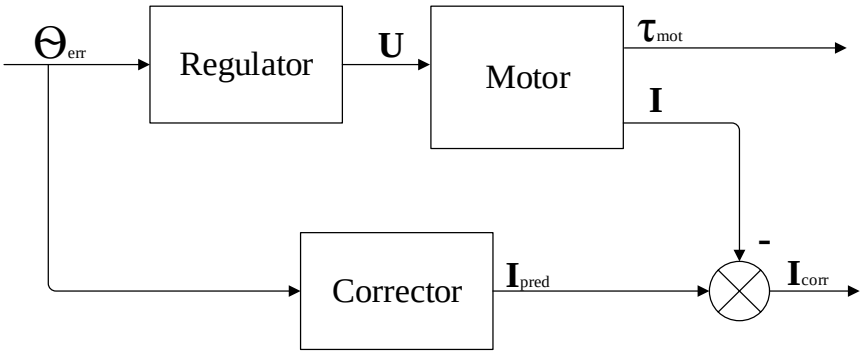


Figure 5.40: Principle of the motor's correction

As is shown in the Figure, the corrected signal comes into being in summation node by subtracting the motor current from the corrector's output. Hence, the corrector working principle constitutes a crucial aspect. Generally, the corrector's model is the same structure as the model of the identified system (5.22). Differences occur among the parameters of both models.
Thus, the real system and the corrector responded variously to a given position error. The corrector model's parameters were empirically adjusted so that the corrector and real system responded similarly to the small position error but differently to the large values.

Figure 5.41: presents the responses of the corrector and the real system for the same given input. The red line denotes the output of the real system when the black line corresponds to the corrector's output.

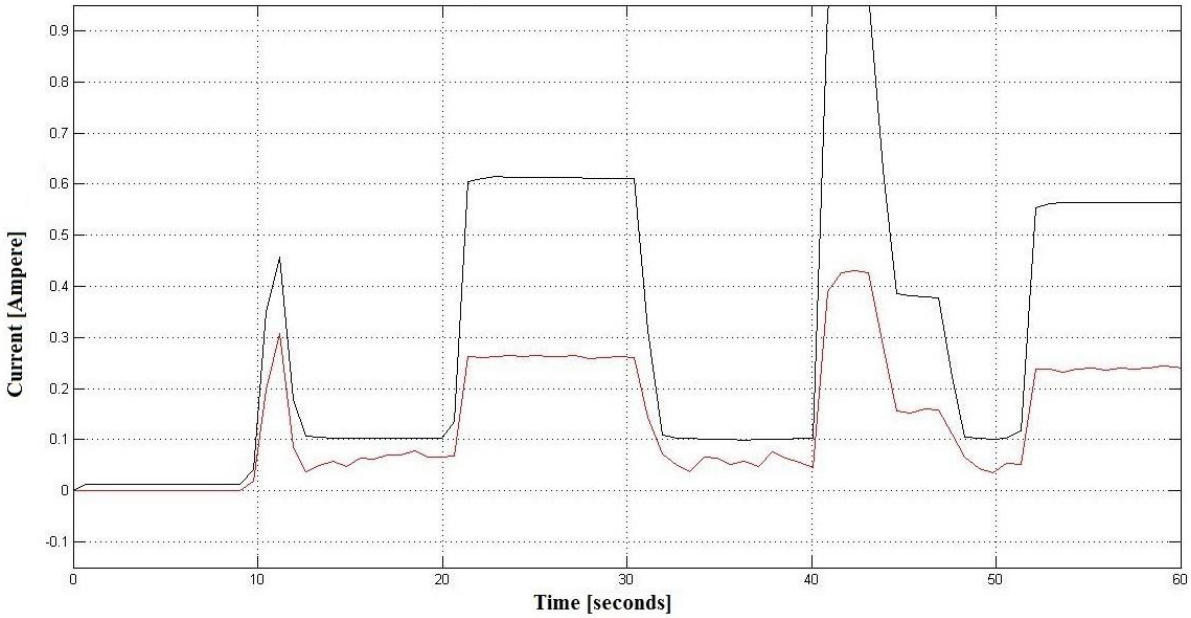


Figure 5.41: Responses of the corrector and the real system for the same given input

Consequently, the combined signal constituted legible information regarding the relationship between the motor current and the applied force. Figure 5.42 illustrates the corrected signal, denoted as a blue line.

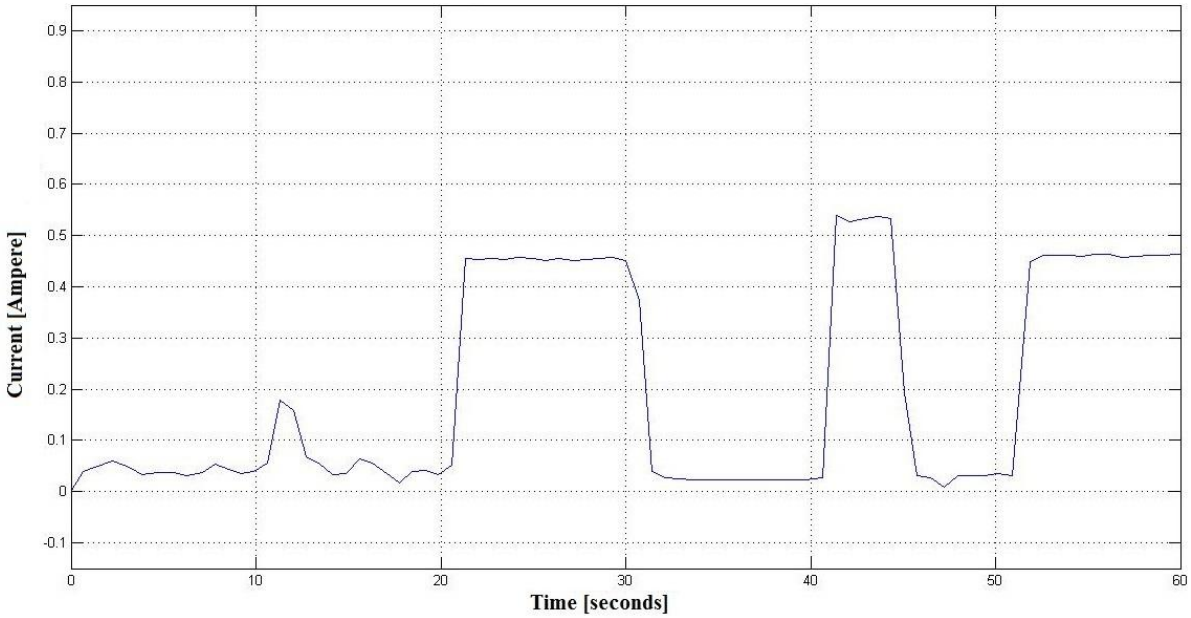


Figure 5.42: Corrected signal

The corrector was implemented in software. The equation (5.22) presents the generic model of the corrector in the frequency domain. Therefore, the model was firstly converted to the discrete form and then implemented as a difference equation. The procedure was similar to the design of the IIR filter. In general, the equation (5.22) can be presented as:

$$G(s) = \frac{A+Bs}{Cs^2+Ds+E} \tag{5.25}$$

where:

A, B, C, D, E are coefficients of Laplace's transfer function

After the transformation of (5.25) to the discrete domain using, e.g. the FOH method [eac], the equation has the following form:

$$G(z) = \frac{Y(z)}{X(z)} = \frac{Fz^2+Gz+H}{Iz^2+Jz+K} \tag{5.26}$$

where:

F, G, H, I, J, K are coefficients of the discrete transfer function

Next, the equation (5.26) was converted to the difference form:

$$y[n] = (Fu[n] + Gu[n-1] + Hu[n-2] - Jy[n-2] - Ky[n-1])\, I^{-1} \tag{5.27}$$

where:

y is filter's output, u is filter's input, n is iteration

Figure 5.43 illustrates the structure of the implemented corrector.

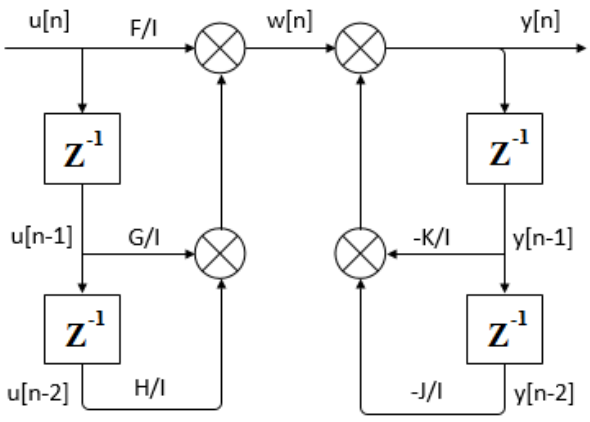


Figure 5.43: Structure of the implemented corrector

The corrector's designing procedure was conducted for each of the robot arm's joints. Thus, the parameters for corresponding correctors are presented in Tab. 5.5.

	ShoulderPitch	**ShoulderRoll**	**ElbowRoll**
F	3.871	5.454	4.786
G	-3.27	-0.3789	-1.682
H	0.5838	0.01722	-0.01305
I	1	1	1
J	-0.9936	0.03364	-0.3696
K	0.0005137	0.001458	0

Table 5.5: Parameters of the implemented correctors

5.4 Implementation of force control algorithms

This subsection discusses force control algorithm implementation. As was mentioned before, the force applied to the robot arm was estimated based on the motor currents of particular joints. Figure 5.44 illustrates the location of those joints, i.e. Elbow Roll - 1, Shoulder Pitch - 2 and Shoulder Roll - 3. In general, by exerting force on the robot arm, the corresponding motor currents increased. In effect, the resultant current signal was utilised to control the robot's walk.

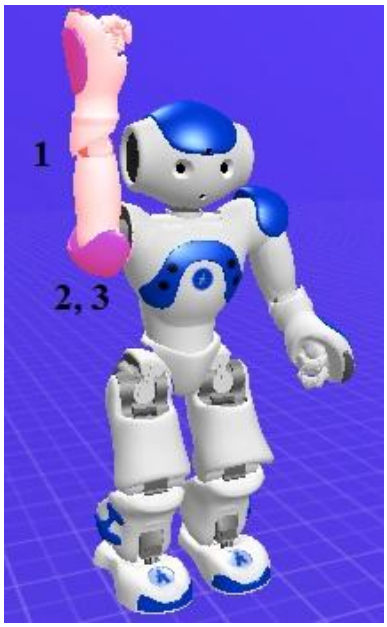


Figure 5.44: Robot pose to human–robot interaction

According to the objectives of the thesis (subsection 1.2), two different control algorithms were implemented. The basic principle of regulating the robot's legs was the same for both controllers - Fig. 5.45.

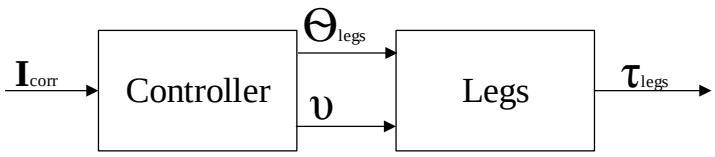


Figure 5.45: Principle of controlling the robot's legs

where:

$\boldsymbol{\tau}_{legs}$ is force/torque generated by the robot legs, $\mathbf{I}_{corr}$ presents corrected signal, Θ_{legs} denotes direction of the robot's walk, υ signifies velocity of the robot's walk.

By and large, the resultant signal of corrected motor currents was introduced to the controller block. Next, the regulation algorithm properly steered the robot legs *via* the settings of the direction and velocity of the robot's walk. Basically, the NAO's API provides ready-made functions to control the motions of whole robot legs by specifying parameters such as the size of the steps, the frequency of making steps by the robot, and the direction of walking. Therefore, the controllers generally regulated the robot's walk by adjusting those parameters.

The first control algorithm was implemented on the basis of the proportional controller. Thus, the control signals, i.e. the velocity and direction of the robot's walk, were proportional to the resultant motor current. The principle of the proportional control algorithm is presented below.

Algorithm 2 The proportional control algorithm

1: Initiate a constant size of the steps - S

2: Initiate the gain for the proportional control signal - K_p

3: Retrieve current information about the resultant motor current - U(t)

4: Calculate the frequency of making steps - $Y(t) = K_p*U(t)$

5: Apply motion command with S and Y(t) parameters

* t - moment in time

For the first controller, the velocity of the robot's walk was proportionally adjusted to the resultant motor current. Since the size of the step was constant, the frequency of making steps was the only one parameter that was actively involved in the regulation.

The second algorithm implemented the PID (proportional-integral-derivative) controller whose working principle is as follows:

$$y(t) = K_p u(t) + K_i \int_0^t u(t) + K_d \frac{du(t)}{dt} \quad (5.28)$$

where:

K_p is proportional term, K_i denotes integral term, K_d is derivative term, u(t) presents controller's input, y(t) signifies controller's output, t symbolises moment in time.

The equation (5.28) consists of three components. The first element corresponds to the proportional controller that was utilised in the previous algorithm. The second component is in charge of integrating the input signal up to t – the moment of time. As a software implementation the iteration was executed as a sum of the input signal between moments 0 and t. This means that the velocity of the robot's walk depended on present and past values of the resultant motor current. The last component, i.e. differentiation of the controller's input, caused the control signal to be more sensitive to the motor current's rapid changes. Algorithm 3 illustrates the working concept of PID control.

Algorithm 3 The PID control algorithm

1: Initiate the gains for proportional, integral, derivative terms - K_p, K_i, K_d

2: Initiate the proportional term to calculate step size - K_s

3: Retrieve current information about the resultant motor current - U(t)

4: Calculate the size of the step - $S = K_s * U(t)$

5: Calculate the frequency of making steps - $Y(t) = K_p U(t) + K_i \int_0^t U(t) + K_d \frac{dU(t)}{dt}$

6: Apply motion command with S and Y(t) parameters

In contrast to the algorithm 2, the PID control algorithm regulated the velocity of the robot's walk by means of adjusting the frequency and the size of the steps. The size of the step was determined by the motor current and proportional term K_s, when the frequency of making steps was obtained by applying the PID regulator. In addition, the integral term limitation was software-implemented in order to prevent the "wind-up" effect.

The above algorithms present a certain simplification of the implemented control algorithms. Nevertheless, they reflect the working principle of proportional and PID controllers.

6. Usability testing for the designed interface

This section presents usability testing for the developed interfaces. In addition to a brief introduction into the aspect of usability testing, the general rules of conducting and designing such a test are illustrated.

Usability testing is a technique used to examine and assess a product by its potential users [31]. The fact that users are actively engaged in the evaluation process allows for the product to be improved towards the users' needs and expectations.

The primary goal of testing is to improve the usability of the product that is being tested. Another intention is to improve the process of designing and developing a product in order to avoid any problems found during the testing phase. Thus, the usability test can be carried out at different phases of developing the product. Usability testing then allows researchers to determine a potential product's faults before introducing it onto the market. In addition, usability testing can be employed to distinguish one of the products with the best usability or ergonomics.

By and large, usability testing aims to qualitatively and/or quantitatively assess a product regarding usability requirements. Therefore, the testing criteria have to be defined in advance. If an evaluated product meets the usability objectives, it can be qualified as usable or exploitable; for instance, the following criteria can determine the usability and ergonomics of a product based on user observations [32, 33]:

- Effectiveness - how quickly and correctly specific tasks can be accomplished using product. Also, it can determine how much time and how many steps a user needed to accomplish an intended task by means of that product.
- Accuracy - specifies how precise the user was when performing the intended task and how many mistakes he or she made.
- Learnability - determines how quickly a new user is able to operate a product.
- Attitude – the user's general impression and perception regarding a product.

Nevertheless, the criteria can differ and are up to the test designer or user. On the whole, the usability testing process consists of several steps, as is presented in Fig. 6.1. The first step involves planning and preparing the test. The test plan usually does not need to be complex, but it should cover the essential requirements, including the overall testing goals.

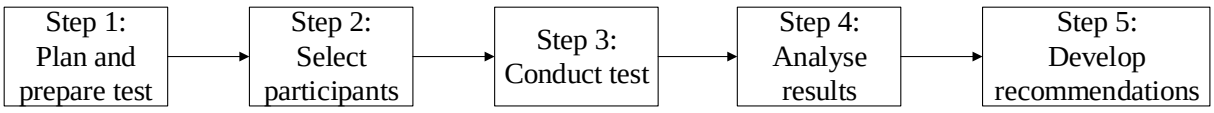


Figure 6.1: Usability testing process [34]

Next, testers are selected from among potential product users. The number of participants usually varies from between 8 and 20. Once the tests are conducted, the results are analysed and the product developer makes a recommendation towards improving a product.

The above-described procedure was conducted to investigate the usability of the developed interfaces (Chapters 4 & 5). The corresponding subsections of this chapter present the particular steps.

6.1 Usability test preparation

Usability test preparation was initiated by sketching out a testing plan. The plan was elaborated in order to define information, such as overall goals, audience profile, the scope of testing and the testing methodology.

Goals: The main goal of the conducted test was to provide feedback regarding usability of the designed interfaces. Moreover, the users specified performances of each interface and determined the problems obstructing human–robot interaction. Finally, testers had to compare the developed solutions and to point to the best solution.

Problem statement: Several questions had to be answered by the user:
1. Were the proposed interfaces easy to use?
2. Did the user feel comfortable during interaction with the robot?
3. What would a user like to improve in the proposed interfaces?
4. Which of the proposed interfaces was the best?
5. How effective, accurate and natural was interaction by means of a particular interface?

User profile: The target users were students who had not experienced pHRI before. To qualify the developed interfaces the users had to be trained to interact with the robot. After this the users assessed the proposed solutions on the basis of executing certain tasks.

Methodology: Usability testing of 8 participants was held at the lobby of the TUAS building. Each session lasted approximately 8 minutes. During the session a user briefly learnt how to interact with the robot and then evaluated two interfaces. Next, a test participant filled in an anonymous questionnaire regarding the performed test (see Appendix A). Additionally, in order to uncover other potential dilemmas, an interview with the user was conducted afterwards.

Testing Scenario: There was no specified scenario describing how the user had to interact with the robot. In general, the user could freely guide the robot during a specific time period. Each interface was evaluated during a 3-minute mini-session.

A usability test of the developed interfaces was conducted in accordance with the plan. Next, the results were analysed and then a recommendation was developed.

6.2. Results analysis and recommendation

Table 6.1 presents additional information about the test participants such as gender, age or educational level. As was mentioned earlier, all of the users were selected from among students. None of them were familiar with the designed interfaces before the test. As shown in the table, most of the interviewees were men aged 20-24.,

Category	**Results**	
The number of test participants		8
Sex	Male	75%
	Female	25%
Age	20 - 24	75%
	24 - 33	25%
Educational level	Bachelor	37%
	Master	63%

Table 6.1: User information

On the whole, the users filled in the questionnaires after completing the testing session. They evaluated interfaces A and B on the basis of the experience they had obtained during their interaction with the robot. Interface A implemented a proportional controller, whereas interface B was extended by the PID algorithm. The parameters of both the controllers were empirically tuned.
A comparison between the two interfaces was performed based on specific criteria (see Appendix A). A user could assess each criterion on a scale of 1–5 (where 5 was the highest grade). The mean grades and standard deviations of the corresponding category are presented in Table 6.2. Moreover, the detail rankings of grading can be found in Fig. C1 - C6 (see Appendix B).

	Effectiveness		**Accuracy**		**Naturalness**	
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**
Interface A	3	0,76	2,75	0,71	2,88	0,99
Interface B	3,63	1,06	2,38	0,52	3,23	0,71

Table 6.2: Mean grades and standard deviations of particular evaluation categories

In order to compare the above-mentioned interfaces, the users could mark each interface in accordance with three categories. Effectiveness denoted the easiness of operating a particular interface, i.e. guiding the robot. Also, it represented the efficiency of performing particular criteria based on time. Accuracy embodied control precision that was provided by a particular interface. During the testing session the user could lead the robot among the obstacles and could dynamically change the environment, thus the robot had to react appropriately. The last category evaluated the naturalness of each approach. The user estimated how naturally and casually the interaction was conducted.

By and large, the interviewees graded interface A as more accurate than interface B; however, the former featured higher effectiveness and naturalness. Also, 63% of the test participants marked interface B as the one with the best usability and ergonomics. Nonetheless, some of results could have been obtained randomly. Thus, in order to evaluate the received outcomes, a statistical hypothesis test was conducted. The main objective was to compare two sets of measurements to assess if their population means differ. Due to the conditions of conducted usability testing, Wilcoxon signed-rank test was employed [35]. The test allowed to validate if the interfaces A and B were significantly different for evaluated categories, i.e. effectiveness, naturalness, accuracy. The test was performed accordingly to the algorithm presented below.

Algorithm 4 Wilcoxon signed-rank test (for $N_r < 10$) [35]

1: Determine the hypothesis H_0 and H_1

2: Determine the significance level α

3: Calculate $|X_{2,i} - X_{1,i}|$ and $\text{sgn}(X_{2,i} - X_{1,i})$, for i =1..N

4: Exclude pairs where $|X_{2,i} - X_{1,i}| = 0$. Let N_r be reduced sample size.

5: Sort the N_r pairs from smallest absolute difference to largest absolute difference, $|X_{2,i} - X_{1,i}|$.

6. Rank the pairs, starting with the smallest as 1. Ties receive a rank equal to the average of the ranks they span. Let R_i denote the rank.

7. Calculate the test statistic $W = |\sum_{i=1}^{Nr}[\text{sgn}(X_{2,i} - X_{1,i}) \cdot R_i]|$

8. Compare W to a critical value from a reference table (see Appendix C).

9: Make a decision about H_0. If $W \geq W_{\text{critical, Nr}}$ then reject null hypothesis.

---

* N is the sample size, the number of pairs. In addition, for i = 1...N , $X_{1,i}$ and $X_{2,i}$ symbolise the measurements.

First step in the algorithm was to define the test hypothesis. H_0 assumed that the median difference between the pairs was zero. In contrast, the alternative hypothesis H_1 proclaimed that X_1 and X_2 had significantly various distributuions.

The significance level was chosen as 0.05. This parameter was the criterion used for rejecting the null hypothesis. It specified the maximum risk of the error that researcher could accept. Further calculation were performed according to the steps 4-9 (see Appendix D). Thus, the following results were obtained for particular categories:

Effectiveness	$W = 11 < W_{0.5,\,6} = 20$	(6.1)
Accuracy	$W = 6 > W_{0.5,\,4} = 5$	(6.2)
Naturalness	$W = 9 < W_{0.5,\,5} = 15$	(6.3)

As can be seen from the above equations (6.1) and (6.3), the null hypothesis H_0 could not have been rejected. Hence, there was no significant different in distributuions of X_1 and X_2. The distinctions between the interfaces in terms of effectiveness and naturalness were not confirmed by the test. However, (6.2) points in favour of alternative hypothesis H_1. Thus, the difference in accuracy between the interfaces A and B was validated.

Finally, users declared their satisfaction regarding both interfaces; however, the survey results show a lack in the accuracy and naturalness of the developed interfaces. Also, test participants made other remarks during the verbal consultation. Firstly, the testers complained about the slow interface reaction, hence they poorly evaluated interface accuracy. The reason for this was the fact that the control program was run on a separate workstation and was connected to the robot via a Wi-Fi connection. As a result a significant delay was introduced to the whole system response. In the future the program should be implemented and compiled directly on the robot, as only this solution can provide full performance of the resources used and will, in turn, improve system accuracy.

Furthermore, all of the participants noticed relevant differences between both interfaces in contrast to the result of Wilcoxon signed-rank test. Interface A was more accurate, however, users complained that the robot moved too slowly. An advantage of interface A was that it provided more predictable control of robot behaviour. The other interface was less precise, but the robot moved much faster in response to a larger force applied to the arm. Thus users evaluated the interface as providing better naturalness and preferable dynamics of robot movements. Besides, if users constantly guided the robot in one direction, the humanoid relevantly speeded up its walking. However, due to the delay in control program, the robot's behaviour was more challenging to predict, therefore, interface accuracy was marked as lower.

The distinction in performance of both interfaces derived from the fact that they implemented various control algorithms. It seems also that better capability of the developed interfaces could be achieved by better tuning of the implemented regulators. A further discussion regarding the users' observations and interface implementation is discussed in the conclusion of the thesis.

7. Conclusion

This section summarises the conducted investigation with respect to thesis objectives. Conclusions are presented on the basis of the obtained results. Besides, further improvements and an extension of the developed interface are proposed.

The main motivation behind this thesis was to develop an interface for physical-human interaction. The interface was to enable a human to control a robot's walk by applying force to the robot's arm. In effect, the robot should have compensated the exerted force by adapting its walk. Therefore, a design methodology of the interface was developed to handle this objective. Next, the solution was implemented and examined using an NAO robot.

The conceptual structure of the interface fulfilled the assumption that the robot lacked sensors enabling it to measure the exerted force. The force applied to the robot arm was estimated by means of information derived from motor current. On the whole, the design procedure required that position control was implemented on the robot arm's joints. Despite the fact that the NAO platform met the criteria by default, the absence of knowledge of the utilised position controller implied the risk that the proposed interface could not match the platform that was used. Therefore, the position control system was firstly investigated to verify if the motor current could be utilised in accordance with the established assumption.

The conducted investigations included identification of the positioning system. The obtained outcomes confirmed usability of the motor current signal with respect to force estimation. However, the current signal had to be corrected due to ambiguity in the measurements. The existing positioning system was inaccurate and did not place the robot arm accordingly to the commanded position criterion. The small encoder's resolution and/or inexact tuning of the position controller could have caused this dilemma. Thus, fair differences of current values were obtained for the same position setting of the robot arm.

The corrector design was based on the model acquired during the identification process. Accordingly, the relationship between the applied force and motor current values was observed. The exact dependency between the physical quantities could not be determined because of the lack of force measurement instruments. Nevertheless, the relation could be estimated by means of external sensors mounted on the robot's wrist. In this case both quantities, exerted force and motor current, could be recorded together for various loads. Hence, the relationship between measurements could be approximated by the interpolation.

Two different control algorithms were implemented on the basis of the refined motor current. The first interface utilised a proportional controller, therefore, the velocity and direction of the robot's walk were proportional to the motor current. The other interface had an embedded PID algorithm, thus the control signal was based on the present as well as past value of the motor current.

The final step selectively examined the ergonomics of the developed interfaces via usability testing. The test was carried out among random students who evaluated each interface with respect to Effectiveness, Accuracy, and Naturalness. During the testing session the users led the robot among obstacles. Based on test questionnaires the interface equipped with the PID controller was marked as the one with the best usability. By and large, the test participants concluded that this interface provided better dynamics of robot walking than the one with the proportional controller. The users observed that the robot's behaviour was more sensitive to sharp changes in applied force. The reason for this were the derivative terms of the control algorithm. Moreover, integration of the current caused that the robot's walking velocity relied on the present as well as past values of the motor current. When users constantly guided the robot in one direction, the humanoid relevantly speeded up its walking.

To sum up, the thesis objectives were conducted with the use of the developed interface for pHRI. Users declared their satisfaction regarding the general solution. Nevertheless, the survey results presented a lack in the accuracy of the developed interfaces. The reason for this could have been the fact that the control program was run on a separate workstation and that it was connected to the robot via a Wi-Fi connection. In effect, an additional delay was introduced. Therefore, the next step should be implementation of a control algorithm directly on the robot platform.

The conducted research was restricted to interaction, where the robot arm was placed in a well-defined position. Nonetheless, it seems that the presented methodology of sensorless force control could be extended to various settings of the robot arm. Consequently, the robot could be employed for diverse tasks where force sensing is required. Thus, further work should investigate this issue.

References

[1] J. Casper and R.R. Murphy, Human-robot interactions during the robot-assisted urban search and rescue response at the World Trade Center, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 33, no. 3, pp. 367 - 385, June 2003.

[2] Kai-Yi Chin, Chin-Hsien Wu, and Zeng-Wei Hong, "A Humanoid Robot as a Teaching Assistant for Primary Education", In *Proceedings of IEEE Conference on Genetic and Evolutionary Computing*, pp. 21-24, May 2011.

[3] Jon Eriksson, Maja J. Mataric and Carolee J. Winstein, " Hands-off Assistive Robotics for Post-Stroke Arm Rehabilitation", In *Proceedings of IEEE Conference on Rehabilitation Robotics*, pp. 21 - 24, June 2005.

[4] David J. Feil-Seifer and Maja J. Matarić, "Human-Robot Interaction", *Invited contribution to Encyclopedia of Complexity and Systems Science*, Robert A. Meyers (eds.), Springer New York, 4643-4659, 2009.

[5] Kerstin Severinson-Eklundh, Anders Green and Helge Hüttenrauch, "Social and collaborative aspects of interaction with a service robot", *Robotics and Autonomous Systems*, vol. 42, no. 3 - 4, pp. 223 - 234, 31 March 2003.

[6] A. De Santis, B. Siciliano, A. De Luca, and A. Bicchi, "An atlas of physical human-robot interaction", *Mechanism and Machine Theory*, vol. 43, no. 3, pp. 253–270, 2008.

[7] C.H. Morimoto, M. Flickner, "Real-Time Multiple Face Detection Using Active llumination", In *Proceedings of IEEE Conference on Automatic Face and Gesture Recognition*, pp. 8 - 13, March 2000.

[8] Liying Cheng, China Qi Sun, Han Su, Yang Cong and Shuying Zhao, "Design and implementation of human-robot interactive demonstration system based on Kinect", In *Proceedings of IEEE Conference on Control and Decision Conference*, pp. 971 - 975, May 2012

[9] A. Aly, A. Tapus, "A Model for Synthesizing a Combined Verbal and NonVerbal Behavior Based on Personality Traits in Human–Robot Interaction", In *Proceedings of IEEE Conference on Human-Robot Interaction*, pp. 325 - 332, March 2013.

[10] K.Kosuge, "Dance Partner Robot: An engineering approach to human-robot interaction", In *Proceedings of IEEE Conference on Human-Robot Interaction*, pp. 201, March 2010.

[11] T. Wosch, W. Feiten, "Reactive motion control for human-robot tactile interaction", In Proceedings of *IEEE International Conference on Robotics and Automation*, vol. 3, pp. 3807 - 3812, May 2002.

[12] V. Duchaine, N. Lauzier, M. Baril, M. Lacasse and C. Gosselin, " A Flexible Robot Skin for Safe Physical Human–Robot Interaction", In *Proceedings of IEEE Conference on Automatic Face and Gesture Recognition*, pp. 8 - 13, March 2000.

[13] M. Fritzsche, N. Elkmann, E. Schulenburg, " Tactile sensing: A key technology for safe physical human robot interaction", In *Proceedings of IEEE Conference on Human-Robot Interaction*, pp. 139 - 140, March 2011.

[14] A.A.N. Kumaar, A. V. Vidyapeetham, " Mobile Robot Programming by Demonstration", In *Proceedings of IEEE Conference on Emerging Trends in Engineering and Technology*, pp. 206 - 209, November 2011.

[15] M.L. Infante, V. Kyrki, "Usability of force-based controllers in physical human-robot interaction", In *Proceedings of IEEE Conference on Human-Robot Interaction*, pp. 355 - 362, March 2011.

[16] M.W. Spong, M. Vidyasagar, "*Robot Dynamics and Control*", John Wiley and Sons, 1 edition, January 1989.

[17] John J. Craig, "*Introduction to Robotics: Mechanics and* Control", Prentice Hall, 3 edition, August 2004.

[18] Tomasz Winiarski, Cezary Zieliński, "Basics of robot force control (In Polish)", *Pomiary Automatyka Robotyka*, vol. 6, pp. 5 - 10, June 2008.

[19] T. Fukuda, et al., " Manipulator for man-robot cooperation", In *Proceedings of IEEE Industrial Electronics, Control and Instrumentation*, vol. 2, pp. 996 - 1001, October 1991.

[20] G. Ferretti, G. Magnani, and P. Rocco, "Assigning virtual tool dynamics to an industrial robot through an admittance controller", In *Proceedings of IEEE Conference on Advanced Robotics*, June 2009.

[21] Aldebaran Robotics, "*NAO Software 1.14.3 documentation*", Online available at http://www.aldebaran-robotics.com

[22] J. l. Simpson, C. David. Cook and Z. Li, "Sensorless Force Estimation for Robots with Friction", in *Proceedings of the Australasian Conference on Robotics and Automation*, 2002.

[23] Heinrich Mellmann, Giuseppe Cotugno, "Dynamic Motion Control: Adaptive Bimanual Grasping for a Humanoid Robot", *Journal Fundamenta Informaticae - Concurrency Specification and Programming*, vol. 112 no. 1, pp. 89-101, January 2011.

[24] Chris Kilner, et al., "Mechatronic design of NAO humanoid", In *Proceedings of IEEE International Conference on Robotics and Automation*, May 2009.

[25] Jason Kulk and James Welsh, "A Low Power Walk for the NAO Robot", In *Proceedings of Australasian Conference on Robotics and Automation*, 2008.

[26] Giuseppe Oriolo, "*Biped robots 3: An introduction to the NAO humanoid*", Online available at http://www.dis.uniroma1.it/~oriolo/amr/

[27] Jacek Snamina, "*Laboratory in Fundamentals of* Automation (in Polish)", AGH University of Science and Technology, Poland, Online available at http://home.agh.edu.pl/~pautom/pliki/laboratoria/lab2.pdf.

[28] Lennart Ljung, "*System identification toolbox User's Guide*", MathWorks, March 2013, Online available at http://www.mathworks.com/help/pdf_doc/ident/ident.pdf

[29] T. Söderström and P. Stoica, "*System Identification*", Prentice Hall, 1989.

[30] R.H. Welch, G. W. Younkin, "How Temperature Affects a Servomotor's Electrical and Mechanical Time Constants", In *Proceedings of IEEE Conference on Industry Applications Conference*, vol. 2, pp. 1041 - 1046, October 2002.

[31] Joseph S. Dumas, Janice C. Redish, "*A practical guide to usability testing*", Intellect Ltd, January 1999.

[32] Jeffrey Rubin , Dana Chisnell, Jared Spool, "*Jeffrey Rubin's Handbook of Usability Testing*", Wiley, 2 edition, May 2008.

[33] Gerry Gaffney, "*Usability Techniques series*", September 2013, Online available at http://infodesign.com.au/usabilityresources/

[34] Kelly Goto, "*Usability testing, Assess Your Site's Navigation & Structure*", September 2013, Online available at http://www.gotomedia.com/downloads/goto_usability.pdf

[35] Frank Wilcoxon, "*Wilcoxon signed-rank test*", October 2013, Online available at http://en.wikipedia.org/wiki/Wilcoxon_signed-rank_test.

Appendix A - Usability test questionnaire

Please complete the following questionnaire

1. Information about the user:

Sex _____
Age _____
Education level _____

2. Assessment of the proposed interfaces

Scale of the evaluation:
1: very poor 2: poor 3: average 4: good 5: very good

- **Interface A**

Effectiveness	Accuracy	Naturalness
1 2 3 4 5	1 2 3 4 5	1 2 3 4 5

Other comments:

- **Interface B**

Effectiveness	Accuracy	Naturalness
1 2 3 4 5	1 2 3 4 5	1 2 3 4 5

Other comments:

3. General impression regarding physical human–robot interaction

3.1 Which of the presented interfaces do you prefer?

A B

Other comments:

3.2 What is your attitude regarding interaction with the robot?

1 2 3 4 5

Other comments:

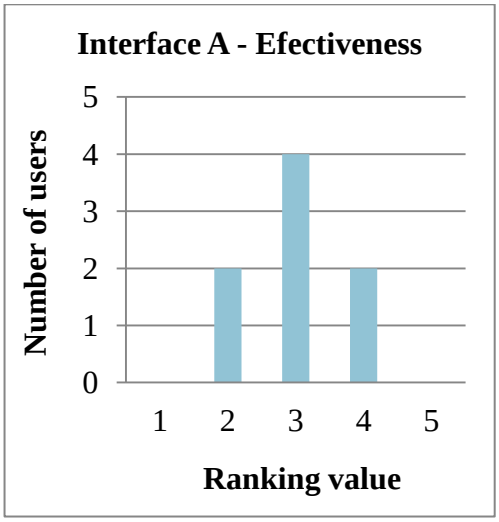
3.2 How would you assess the learnability of the proposed interfaces?

1 2 3 4 5

Other comments:

3.3 What would you like to improve in the proposed approaches? *(optional)*

Appendix B - The rankings of grading



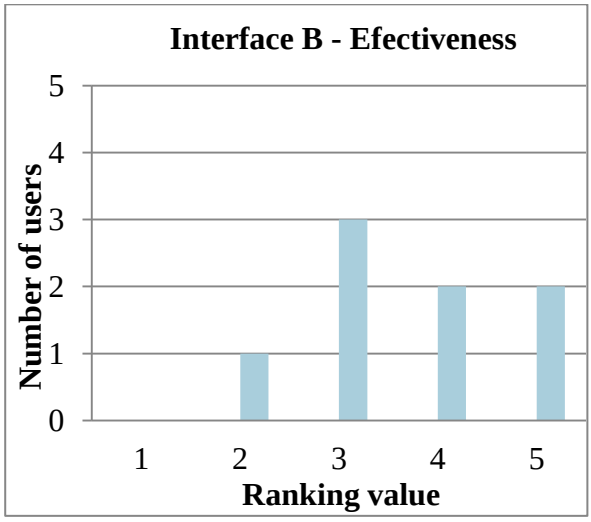
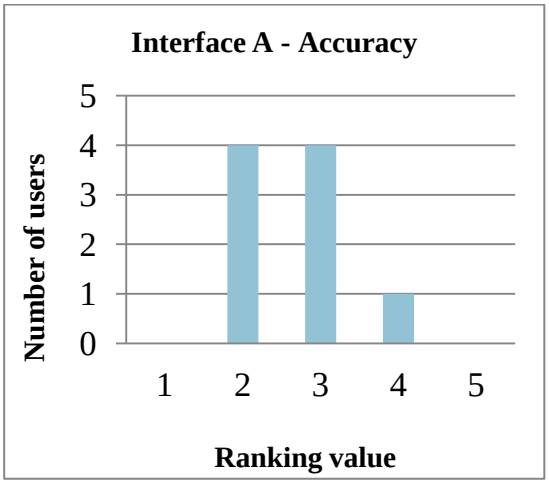


Figure C1, C2: Grading of Effectiveness



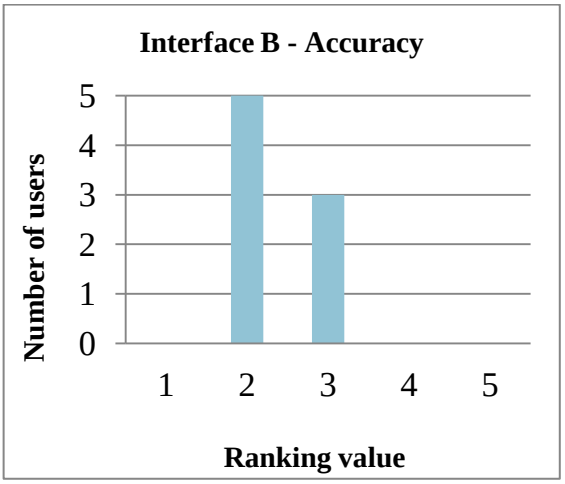
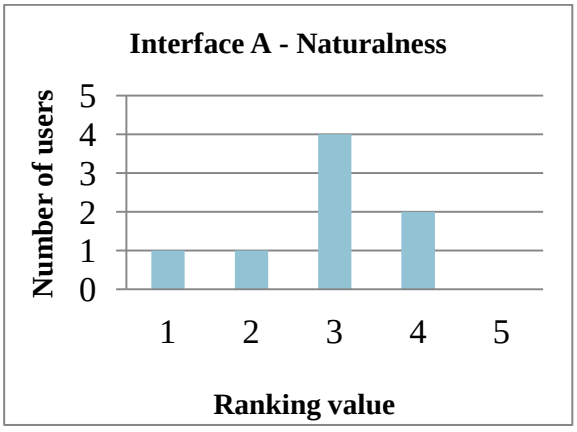


Figure C3, C4: Grading of Accuracy



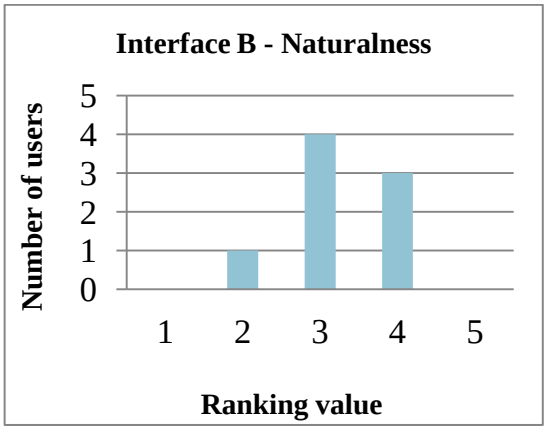


Figure C5, C6: Grading of Naturalness

Appendix C - Critical values for the Wilcoxon Rank-Sum

One Tail Areas	0.005	0.01	0.025	0.05	0.10	0.50
$n = 4$	0	0	0	0	1	5
5	0	0	0	1	3	7.5
6	0	0	1	3	4	10.5
7	0	1	3	4	6	14
8	1	2	4	6	9	18
9	2	4	6	9	11	22.5
10	4	6	9	11	15	27.5
11	6	8	11	14	18	33
12	8	10	14	18	22	39
13	10	13	18	22	27	45.5
14	13	16	22	26	32	52.5
15	16	20	26	31	37	60
16	20	24	30	36	43	68
17	24	28	35	42	49	76.5
18	28	33	41	48	56	85.5
19	33	38	47	54	63	95
20	38	44	53	61	70	105
21	43	50	59	68	78	115.5
22	49	56	66	76	87	126.5
23	55	63	74	84	95	138
24	62	70	82	92	105	150
25	69	77	90	101	114	162.5
26	76	85	99	111	125	175.5
27	84	93	108	120	135	189
28	92	102	117	131	146	203
29	101	111	127	141	158	217.5
30	110	121	138	152	170	232.5

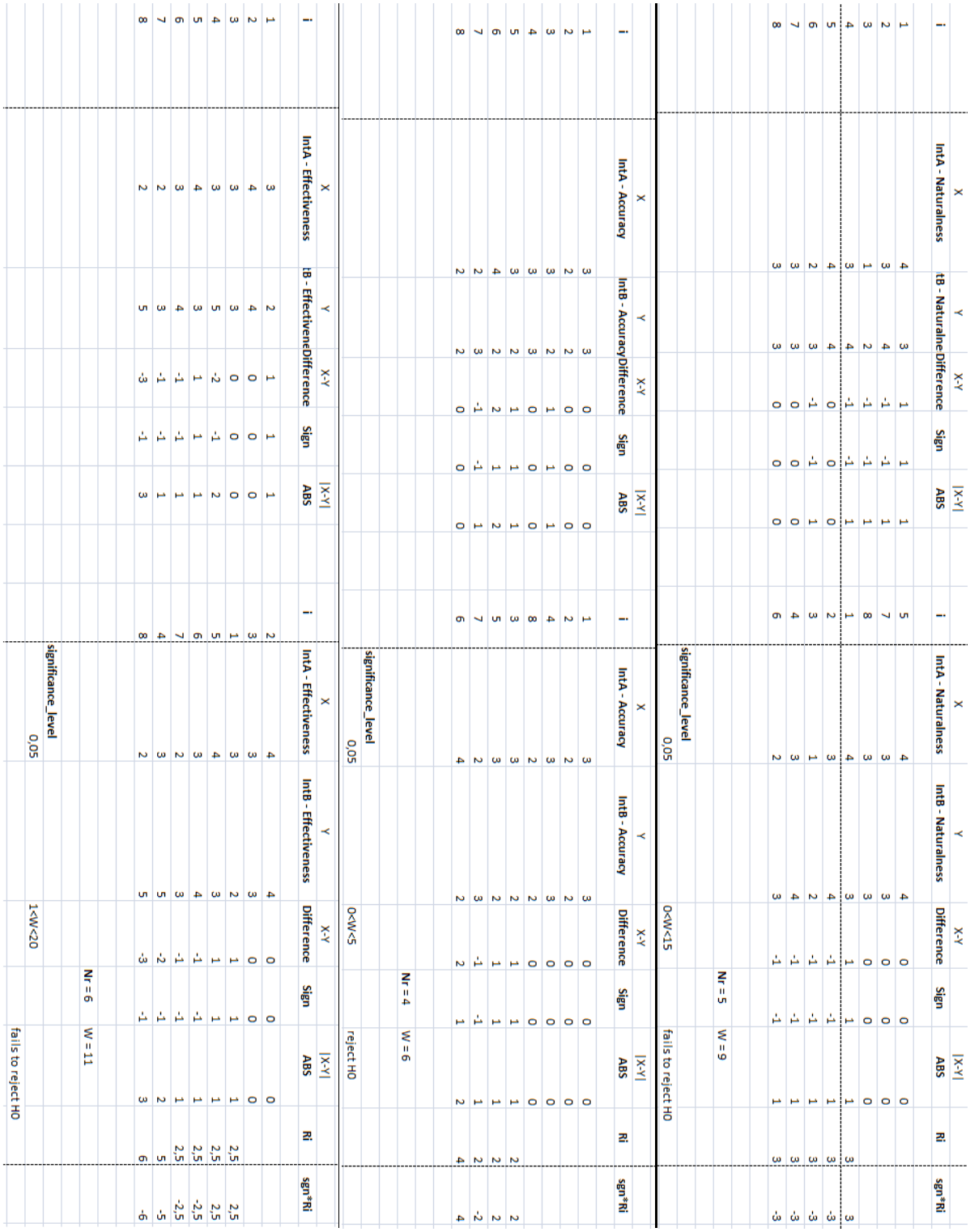
Upper tail critical value is given by:

2 × (value under column 0.50) — (lower tail value).

The Wilcoxon test statistic must be LESS than the lower tail value or GREATER than the upper tail value for significance.

Table 6.3: Critical Values for the Wilcoxon Rank-Sum [35]

Appendix D - The results of Wilcoxon signed-rank test



i	X IntA - Naturalness	Y IntB - Naturalness	X-Y Difference	Sign	|X-Y| ABS	i	X IntA - Naturalness	Y IntB - Naturalness	X-Y Difference	Sign	|X-Y| ABS	Ri	sgn*Ri
1	4	3	1	1	1	5	4	4	0	0	0		
2	3	4	-1	-1	1	7	3	3	0	0	0		
3	1	2	-1	-1	1	8	3	3	0	0	0		
4	3	4	-1	-1	1	1	4	3	1	1	1	3	-3
5	4	4	0	0	0	2	3	4	-1	-1	1	3	-3
6	2	3	-1	-1	1	3	1	2	-1	-1	1	3	-3
7	3	3	0	0	0	4	3	4	-1	-1	1	3	-3
8	3	3	0	0	0	6	2	3	-1	-1	1	3	-3
							Nr = 5	W = 9					
							significance_level				fails to reject H0		
							0,05		0<W<15				

i	X IntA - Accuracy	Y IntB - Accuracy	X-Y Difference	Sign	|X-Y| ABS	i	X IntA - Accuracy	Y IntB - Accuracy	X-Y Difference	Sign	|X-Y| ABS	Ri	sgn*Ri
1	3	3	0	0	0	1	3	3	0	0	0		
2	2	2	0	0	0	2	2	2	0	0	0		
3	3	2	1	1	1	4	3	3	0	0	0		
4	3	3	0	0	0	8	2	2	0	0	0		
5	3	2	1	1	1	3	3	2	1	1	1	2	2
6	4	2	2	1	2	5	3	2	1	1	1	2	2
7	2	3	-1	-1	1	7	2	3	-1	-1	1	2	-2
8	2	2	0	0	0	6	4	2	2	1	2	4	4
							Nr = 4	W = 6					
							significance_level				reject H0		
							0,05		0<W<5				

i	X IntA - Effectiveness	Y tB - Effectiveness	X-Y Difference	Sign	|X-Y| ABS	i	X IntA - Effectiveness	Y IntB - Effectiveness	X-Y Difference	Sign	|X-Y| ABS	Ri	sgn*Ri
1	3	2	1	1	1	2	4	4	0	0	0		
2	4	4	0	0	0	3	3	3	0	0	0		
3	3	3	0	0	0	1	3	2	1	1	1	2,5	2,5
4	3	5	-2	-1	2	5	4	3	1	1	1	2,5	2,5
5	4	3	1	1	1	6	3	4	-1	-1	1	2,5	-2,5
6	3	4	-1	-1	1	7	2	3	-1	-1	1	2,5	-2,5
7	2	3	-1	-1	1	4	3	5	-2	-1	2	5	-5
8	2	5	-3	-1	3	8	2	5	-3	-1	3	6	-6
								Nr = 6	W = 11				
							significance_level						
							0,05		1<W<20				
											fails to reject H0		

Figure E1, E3: The results of Wilcoxon test