Final Master Thesis
Report

SPEECH AND MOTION RECOGNITION FOR A ROBOT ASSISTANT IN DRESSING

January 21, 2017

Master
in
Artificial Intelligence

Andrés Flores Valle

Advisor
Aleksandar Jevtić
Institut de Robòtica i Informàtica Industrial, CSIC-UPC

Tutor
Cecilio Angulo Bahón
Automatic Control Department ESAII (UPC)

Institut de Robòtica i Informàtica Industrial, CSIC-UPC
Universitat Politècnica de Catalunya (UPC)

Abstract

Service robotics is a fast-growing field that attracts interest from both academic and industrial community. In particular, assistive robotics targets the healthcare domain that struggles with increased costs of care and lack of nursing staff. Assistive robots have an enormous potential to improve the life quality of patients with reduced mobility by providing assistance with activities of daily living, such as dressing. However, very few of these robots can be found in real commercial settings. One of the reasons is the complexity of human-robot interaction (HRI) with non-expert users, which was the main motivation for this work.

The main contribution of the presented work is the development of an HRI framework that allows an autonomous robot to interact with the users in order to assist them to dress. Several algorithms that rely on speech and motion recognition were developed. In particular, the algorithms for recognition of voice commands, gestures and body postures were used to infer user's attention and intentions during dressing. Additionally, a user adaptation method, which consist of pointing calibration and robot position adjustment algorithms, was proposed to improve the robot task performance and reduce the users' overall effort and frustration. The algorithms were implemented on a robot manipulator arm equipped with two RGB-D cameras and a microphone array. The integration of the entire robotic system was done in Robot Operating System (ROS), which allows modular programming for easier integration of hardware and software components. The robotic system was evaluated in a series of experiments with users, with the goal of assisting them to put on a shoe. Several metrics were used to compare the performance and workload of the users with and without previous robot adaptation.

Contents

List of Figures

Chapter 1

Introduction

Service robotics is a fast-growing field in various application domains. This work focused on assistive robotics in the healthcare domain, in particular, with robots assistants with activities of daily living for people with reduced mobility.

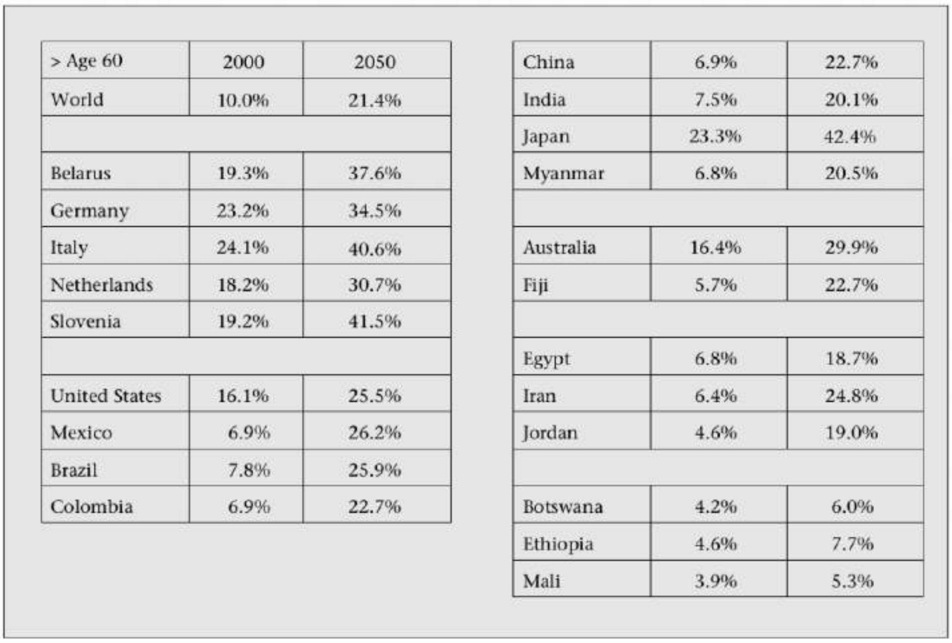
1.1 Motivation

Worldwide, the human population is constantly growing while the opposite trend was reported in the more developed regions. By 2050, the world population is expected to increase by 2 to 4 billion people [1]. This growth will have a profound demographic consequence: while in 2000, 10 percent of the world's population was over 60 years old, by 2050 this proportion will have more than doubled. Fig. 1.1 shows population statistics for some countries on different continents.

Aging societies struggle to maintain the quality of living, especially for their senior members. Due to increased cost of healthcare and lack of professional carers, the highest burden falls on the family members. Older adults often suffer from age-related physical and cognitive impairment. Some studies report that more than half of the patients 75 years or older need assistance with Activities of Daily Living (ADL) [2].

Some other groups of patients, such as the ones that suffered from spinal cord injury (SCI) or stroke, are often in need of assistance with ADL. Worldwide, there are 250-906 patients with SCI per million inhabitants; traffic accidents and aging are reported as two main causes of SCI [3]. The World Health Organisation (WHO) reported that 15 million people suffer from stroke each year, of which one third remains permanently disabled [4]. The result is a reduced physical mobility that affects the ability for independent living and increases the reliance on caregivers.

While human carers cannot and will not be replaced, assistive technologies can support humans, improving the life quality for both older adults



> Age 60	2000	2050
World	10.0%	21.4%
Belarus	19.3%	37.6%
Germany	23.2%	34.5%
Italy	24.1%	40.6%
Netherlands	18.2%	30.7%
Slovenia	19.2%	41.5%
United States	16.1%	25.5%
Mexico	6.9%	26.2%
Brazil	7.8%	25.9%
Colombia	6.9%	22.7%
China	6.9%	22.7%
India	7.5%	20.1%
Japan	23.3%	42.4%
Myanmar	6.8%	20.5%
Australia	16.4%	29.9%
Fiji	5.7%	22.7%
Egypt	6.8%	18.7%
Iran	6.4%	24.8%
Jordan	4.6%	19.0%
Botswana	4.2%	6.0%
Ethiopia	4.6%	7.7%
Mali	3.9%	5.3%

Figure 1.1: Population statistics for typical selected countries worldwide. Source: United Nations Population Division.

and their caregivers [5]. Assistive robots, in particular, can help patients with the recovery and compensate for increased costs of care and lack of nursing staff [6, 7]. The potential to prolong independent living of older adults and improve quality of living of patients with reduced mobility is the main motivation of this work. Assistive robots may possess social skills making them more capable in helping patients [8, 9, 10]. In some cases, users may prefer robot assistance to human assistance [11].

The present work was developed under the framework of the I-DRESS project[1], which aims to develop a robotic system for proactive assistance with dressing to users with reduced mobility. In my work, the final system consisted of a highly dexterous robot manipulator equipped with sensors for multi-modal human-robot interaction. By using speech and gesture recognition, the robot was able to assist the user with a dressing task.

1.2 Use-case scenario

The scenario of this study addressed the daily activity of putting on a shoe to a person seated at a bedside. The user is presumed to have reduced mobility, with partial control over his/her legs: the person can lift the legs and move the feet properly. Recognition of mutual intentions, both user's and robot's,

---

[1] I-DRESS project website: https://www.i-dress-project.eu/

is required for a natural human-robot interaction [12]. Therefore, in this study the interaction is performed using two different modalities: speech and gesture recognition. The dressing assistance was implemented on a Barrett's 7-DOF Whole Arm Manipulator (WAM) robot. Users experienced less workload when interacting with a single-arm robot [13] rather than two robot arms. The WAM robot is equipped with a depth camera and a microphone array to track the user's activity and decide when and how to assist.

The user may choose from a set of different shoes, using either voice commands or gestures, or a combination of both (diectic expressions). The robot's task is to pick up the selected shoe and bring it close to the user, so he or she can reasonably comfortably put the foot inside. People vary significantly in their skills, culture, habits, behaviours, etc., and these factors affect their choices and preferences. In my work, I proposed adaptation to users as a method to improve robot's performance. Through interaction, users are allows to teach the robot their preferences during the dressing task. The final action of inserting the foot inside the shoe is performed by the user, and it marks the end of the task.

1.3 Main contributions

The main goal of this work is to develop and evaluate an autonomous robotic system that assists a person in putting on a shoe. To successfully provide assistance with a dressing task, a robot requires a set of skills. From studying human behaviour of assisting another human to dress, it is obvious that a previous knowledge about the task and the environment is required. Specifically, in order to assists a patient to put on a shoe, the caregiver needs to know the location of the available shoes and the patient's foot. Moreover, the interaction is developed using scenario-specific vocabularies of spoken utterances, gestures, and body postures. The work defined several research and development objectives, which also represent the main contributions:

- To implement algorithms for speech recognition and text-to-speech synthesis. Verbal interaction allows the user to give commands to the robot; it also allows the robot to inform the user about its current state, or suggest a necessary action to the user.

- To develop algorithms for gesture and body posture recognition. For this, I implemented a motion tracking algorithm that provides the position and orientation of the user's body parts, such as the foot or the hand.

- To implement an algorithm for object color segmentation. In the scenario, shoes were marked with different colors to allow their recognition and localization.

- To develop an interaction framework that integrates all above-mentioned algorithms to recognize user's attention and intentions, and provide current robot state with a required dressing action.
- To implement the interaction framework on a commercial Barrett's 7-DOF Whole Arm Manipulator (WAM) robot, equipped with a gripper and sensors. The implementation was done in Robot Operating System (ROS), which became a standard framework for development of open-source robot applications.
- To perform the final system evaluation through experiments with users. The system was evaluated using a set of quantitative and qualitative metrics to measure performance and workload

Chapter 2

State of the art

So far, there are no commercial robot assistant in dressing available on the market. The presented literature overview provides an insight into the service robotics domain, and focuses on research in robots applied to assisted dressing. Finally, a study of the state of the art in multi-modal human-robot interaction is presented.

2.1 Service robotics

A service robot can be defined as a robot that performs useful tasks for humans or equipment excluding industrial automation application [14]. Therefore it is not the hardware that distinguishes an industrial robot from a service robot, but the application domain. According to Silicon Valley Robotics [15], the number of successful service robotics companies is still very limited. The main challenges in the field are associated with the difficulty of implementing autonomous behaviors with the required knowledge of contexts, safety, and compliance. In fact, the current research in this field aims to make them safe for people, easy to teach by non-expert users, able to manipulate both rigid and deformable objects, and with the ability to adapt in non-structured and dynamic environments [16]. Still, one of the greatest challenges of service robotics is the Human-Robot Interaction (HRI).

HRI is an interdisciplinary research field that puts together areas such as robotics, social and cognitive sciences, medicine, and neuroscience, among others. The interaction with humans can provide social capabilities to service robots. More specific within the field is assistive robotics, that provides assistance to human users [17]. Assistive robots are getting much attention nowadays due to the advances in artificial intelligence and manufacturing of more capable robots. A robot's physical embodiment, its appearance, and its shared context with the user, are fundamental for creating a time-extended engaging relationship with the user. The research in SAR has several application domains identified: care of older adults, care of individuals with physical recovery/rehabilitation and training needs, or care of individuals with cognitive and social disabilities.

It is clear that SAR presents a huge potential for development of our society, but it also faces the challenges listed before. The communication between robot and human, such as through speech or gestures, plays a crucial role, making the robot to appear more social and natural [18]. Some recent studies have employed service robots as assistants for ADL such as cleaning [19] or cooking [20].

However, dressing assistance remains a challenging task. Although it is still an open field for robotics, this is one of the basic assistance activities in daily life for disabled and older adults. Indeed, during the assisted dressing, robots must not only interact with the assisted person (whose posture vary during the assistance), but also handle non-rigid clothes. Therefore, the robot is required to adapt to user's motion, as well as to manipulation of soft objects.

2.2 Assisted dressing

Assisted dressing received little attention in robotics research. Early studies proposed assisted dressing on a mannequin: the goal was to pull a T-shirt over the mannequin's head using a dual-robot arm [21]. During the assistance, the state of the T-shirt was recognized in real-time by tracking markers located on its collar and sleeves. The same authors extended this work by using a depth camera to detect and track the T-shirt [22]. Later studies focused on the interaction of the robot and non-rigid garments during dressing. A framework based on reinforcement learning was proposed for learning of the motor skills for cloth manipulation [23]. A robot was required to dress itself by putting its two arms into the corresponding sleeves of the T-shirt. In [24], the reinforcement learning was again applied to dress a T-shirt to a mannequin using topological coordinates, which describe relationships between the mannequin's posture and the estimated T-shirt state. Learning motion tasks in a real environments with non-rigid objects does not only require a reinforcement algorithm, but also a compliant robot controller for safety. In [25] these parts were unified into a framework in order to learn the safety-critical robotic task of wrapping a scarf around a mannequin's neck. The scenario with a mannequin has a limited utility for real world applications because its position is always fixed. The obtained results are difficult to generalize when applied to dynamic human postures. In addition, the reinforcement learning, that was commonly used to teach robot new motions, requires multiple trials and errors, resulting in potential risks for user's safety.

Guided by the fact that dressing tasks should deal with dynamic user's poses, Yixing Gao et al. [26] focused on tracking the user's body and building personalized user models. The dressing application was designed for users with upper-body mobility limitations, who were assisted by a robot to put on a sleeveless jacket. Similarly, Paulo Costeira et al. [27] proposed an approach for personalized dressing assistance that allows the robot and user to take turns when moving, taking into account user's limitations. Although these studies proposed adaptation to the user's needs and limitations, they did not consider any interaction with clothes.

Recent work by Kimitoshi Yamazaki et al. [28], includes both cloth interaction and personalized assistance for users. In their experiments, a humanoid robot assisted users in putting on a pair of trousers; users started the task in a seated position. The state of the trousers was estimated using an optical flow method using image streams [29]. The consideration of manipulation failures during the dressing, and recognition of the type of failure were one of the main contributions of this work. These capabilities play a crucial role in practical applications due to the difficulty inherent in controlling the fluid motions of the clothes.

Finally, Gregorio Chance et al. [30] proposed a multi-modal framework for a compliant robot arm in a dressing jacket task with a mannequin. Robot end-effector trajectory planning used the estimation of the mannequin's arms joints position obtained from reflective markers attached to the joints. A failure-detection method that used torque feedback and sensor tag data was also developed. A vocabulary of speech commands was implemented allowing the user to successfully correct detected dressing errors. A similar approach was used to develop a multi-modal interaction framework in this current work. These HRI modalities, without the implementation of complex learning algorithms, have been applied successfully to a robot for dressing assistance.

Summary of relevant publications is shown in Table 2.1, providing details about the methodology, experiments with users and evaluation metrics.

2.3 Multi-modal human-robot interaction

2.3.1 Speech recognition

Humans commonly use Natural Language (NL) to interact in the social context. This was the main motivation behind the development of a speech interface for the robot assistant in dressing. Speech provides a natural and intuitive way to express intentions or issue commands to a robot, but it also allows receiving robot's feedback.

Speech recognition (SR) has become a valuable industrial tool. Hardware and software system developers, consumer product designers, researchers, and innovative computer users are creating speech-recognition applications that cover many domains [31]. Several SR interfaces have been widely applied to robot systems [32, 33, 34, 35, 36]. Most of these studies focused

Publication	**Application Domain**	**Methodology**	**User Tests**	**Evaluation Metrics**
Tomoya Tamei et. al. 2011 [21]	T-shirt Dressing	-Reinforcement learning -Cloth state recognition	No	-Reward function from learning -Trajectory motion
Nishanth Koganti et. al. 2013 [22]	Cloth state estimation	-Estimation of human-cloth relationship using topology coordinates	No	-Accuracy of the detected state
Takamitsu Matsubara et. al. 2013 [23]	Cloth interaction	-Reinforcement learning -Representation of the robot-cloth relationship using topological coordinates	No	- Reward function from learning - Trajectory motion
Nishanth Koganti et. al. 2014 [24]	T-shirt dressing	- Reinforcement learning - Representation of the human-cloth relationship using topological coordinates - Ellipse fitting algorithm for T-shirt state	No	- Reward function from learning - Robustness of ellipse fitting algorithm - Accuracy of the T-shirt detected state
Yixing Gao et. al 2015 [26]	Jacket dressing	- Real-time human upper-body pose recognition - User modelling	Yes	- Accuracy of the pose estimation
Steven D. Klee1 et. al. 2015 [27]	Personalized assistance in hat dressing	- Algorithm to divide the dressing task and plan robot motion	Yes	- Number of iterations in the algorithm - Execution time
Kimitoshi Yamazaki et. al. 2013 [29]	Cloth state estimation	- Dynamic state matching for cloth state	No	- Success rate of the matching state - Accuracy of the matching state
Kimitoshi Yamazaki 2016 [28]	Trousers dressing	- Dynamic state matching for cloth state - User's legs estimation - End-effector trajectory planning	Yes	- Success rate of the matching state - Success rate of the dressing task
Greg Chance et. al. 2016 [30]	Jacket dressing	- Trajectory planning - Safety and close-proximity manipulation - Failure detection - Human-robot interaction	No	-Time of dressing performance

Table 2.1: Summary of the most relevant studies in assisted dressing

applications for mobile service robots that used predefined vocabularies. Text-to-speech has been applied to robots as feedback to the user [37]. The SR technology has also had a significant impact on the healthcare domain [38], allowing healthcare providers to operate more cost-efficiency while providing a better level of patient care [39]. Some notable applications include replacement of medical transcriptions [40] and patient monitoring [41].

In the assisted-dressing context, SR can provide a natural way to communicate with, teach, and correct the robot from possible failures. However, contributions in this domain have so far been limited to the recognition of predefined spoken commands [30].

In this work, SP was used as a powerful interactive tool allowing two-way human-robot communication, either in the form of spoken utterances or in synergy with gesture interface to create the so-called deictic expressions (see Section 3.2.1)

2.3.2 Motion recognition

Analysis of human behaviour through visual information has captured the attention of several computer science communities. Human motion tracking was initially achieved via images from a conventional camera; however the arrival of depth sensors, such as Microsoft Kinect, have made a new type of data available [42]. Much of the existing work focuses on body part detection and pose estimation, [43, 44, 45, 46], posture recognition [47] and gesture recognition [48, 49, 50]. These methods provide the basis for effective and time-extended HRI. Some relevant application include robot navigation [51], or awareness of user's attention [52].

Human pose recognition has also found its way to applications in healthcare domain. Monitoring of older adults [53] and coaching during physical exercises [54] were some of the target applications.

For assistance in dressing, recognition of human poses is required not only to successfully accomplish the task, but also to ensure safety in the close interaction. Motion recognition can provide useful information about the user's attention and intentions. For instance, if during the shoe-dressing scenario the user pulls back the foot, it may suggest that the the user does not want to be dressed in that moment. Indeed, most of the studies listed in Table 2.1 considered pose recognition as the unique way to interact with the robot. In this work, human motion recognition was used for user tracking, pointing recognition and posture recognition.

2.3.3 Integration of interaction modalities

A multi-modal interface has a distinct advantage over a single interaction modality in that it can effectively reduce recognition uncertainty, and therefore improve the robustness of the system [55]. Various methods for programming of interactive systems have been proposed [56, 57]. Previous studies have shown that the robots capable of interacting through speech and gesture have more flexibility in performing the tasks that require close HRI [58, 59, 60], and are more efficient in recognizing the user's intentions [61]. Multi-modal interfaces in the healthcare domain have already been applied to assistive surgery [62] and assistance with ADL to older adults [63, 64].

In this work, a framework that combines speech and motion recognition was developed and evaluated in the assisted-dressing scenario. It is a unique system-integration effort and as such represents a contribution beyond the state of the art.

Chapter 3

Methodology

The proposed system consists of hardware and software components and their integration required a significant development effort. The presented methodology describes the technical aspects of the system, along with the developed algorithms that provided different robot features. Some of the algorithms were used off the shelf, but still required implementation and integration with other system components. Finally, the details about the integration in Robot Operating System (ROS) are presented.

3.1 Hardware

The development of the robot assistant in dressing required integration of several hardware and software components. The central part of the systems was a Barrett's 7-DOF WAM robotic arm equipped with an in-house developed gripper for shoe grasping. Two Microsoft Kinect sensors, an XBOX 360 and a Kinect One, were used for shoe color recognition and user tracking, respectively. Three personal computers (PC) were used to run the whole system. A PC running Ubuntu 12.04 LTS 64-bit, powered by an Intel quad-core Q9550 CPU @ 2.83GHz x 4 with 8GB of RAM was used to run most of the implemented algorithms and to connect the XBOX 360 Kinect depth camera. The second PC running Ubuntu 12.04 LTS 64-bit powered by an Intel Core i5-2400 CPU @ 3.10GHz 4 and 4 GB of RAM was used to control the WAM robot and the gripper, having all the necessary drivers installed. The third PC running Windows 8.1 Pro 64-bit, powered by an Intel Core i7 X990 @ 3.47GHz and 2.80GHz and 16GB of RAM, processed the data related with the speech recognition and user tracking. The data was obtained from the Kinect One camera using the Kinect for Windows SDK 2.0. The three computers were integrated using the ROS platform and communicated via laboratory Ethernet. A diagram showing all the system components and their communication is shown in Fig. 3.1.

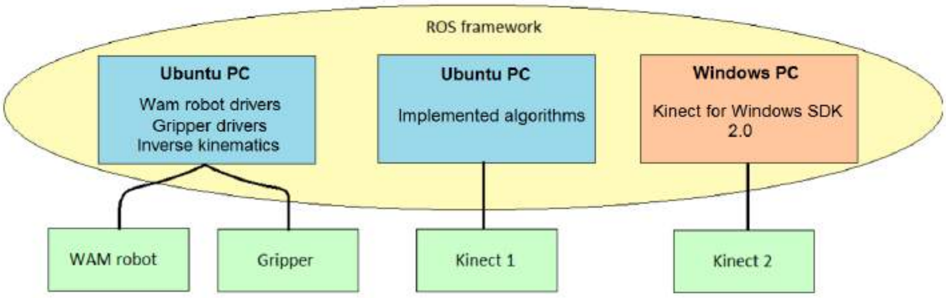


Figure 3.1: Hardware diagram

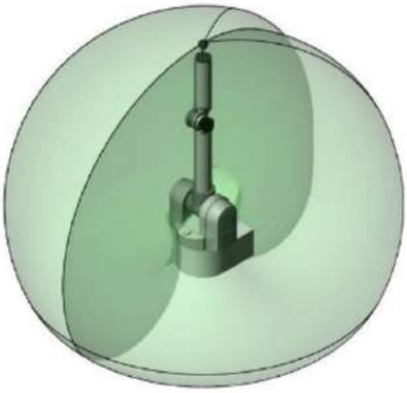


Figure 3.2: Workspace of the Barrett's 7-DOF WAM, isometric view

3.1.1 Robot manipulator arm

The manipulation of the shoes and the dressing assistance were performed by a WAM robot[1], which has a generally spherical workspace of approximately $2m$ in diameter, as seen in Fig. 3.2. Its seven degrees of freedom allow a high flexibility in reaching any point inside its workspace.

3.1.2 RGB-D cameras and microphone array

A first-generation XBOX 360 Microsoft Kinect sensor, from now referred as Kinect 1, was used for the shoe color recognition and location. Its detection range is from $0.4m$ to $4m$, with a vertical viewing angle of 43°and the horizontal viewing angle of 57°. The image resolution in both the depth stream and the color stream are respectively 320x240 and 640x480 with a rate of $30fps$.

User tracking, speech, pose and gesture recognition were performed by

---

[1] WAM specifications: http://www.barrett.com/DS_WAM.pdf

	Version 1	**Version 2**
Depth Range	0.4m - 4.0m	0.4m - 4.5m
Detection Range	0.8m - 4m	0.5m - 4.5m
Color Stream	640x480	1920x1080
Depth Stream	320x240	512x424
Infrared Stream	None	512x424
Audio Stream	4-mic array 16kHz	4-mic array 48 kHz
USB	2.0	3.0
FOV	57 °H, 43 °V	70°H, 60°V
Tilt Motor	Yes	No

Table 3.1: Specifications of first and second-generation of Microsoft Kinect

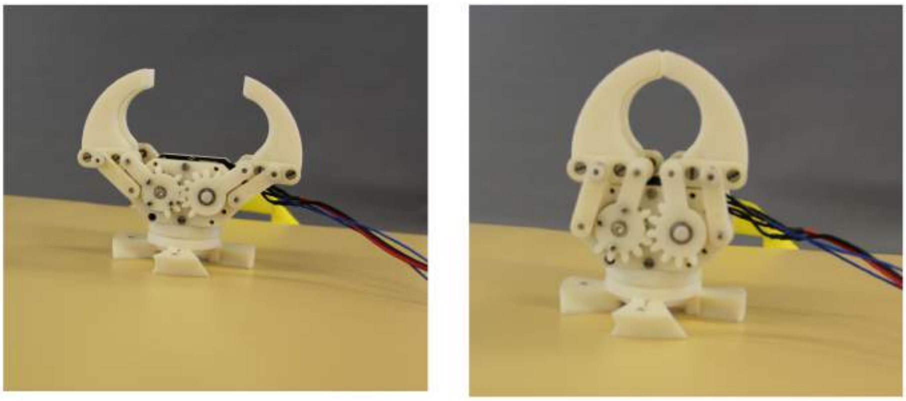


Figure 3.3: Gripper in different states: open (left) and closed (right)

a second-generation Microsoft Kinect sensor, from now referred as Kinect 2. The user detection range is from $0.5m$ to $4.5m$, improved from the first generation (which needs more than $0.8m$), with a vertical and horizontal viewing angle of 60°and 70°respectively. The color and depth resolutions are also improved with respect to its predecessor, which improves the overall user-tracking performance.

The Kinect 2 has integrated four microphones as a microphone array that operates with a frequency of $40kHz$ for the speech recognition and sound localization.

The main specifications for both cameras are given in Table 3.1.

3.1.3 Robot gripper

A 3D printed in-house made gripper was attached to the last joint of the WAM robot for object grasping, i.e. grasping of the shoes. The gripper has four fingers, two on each side, which are opened and closed with a servo motor. When closed, the gripper's fingers make a physical contact, and when open, they are separated $5.5cm$ as shown in Fig. 3.3

The separation of the fingers leaves enough room to securely grasp a

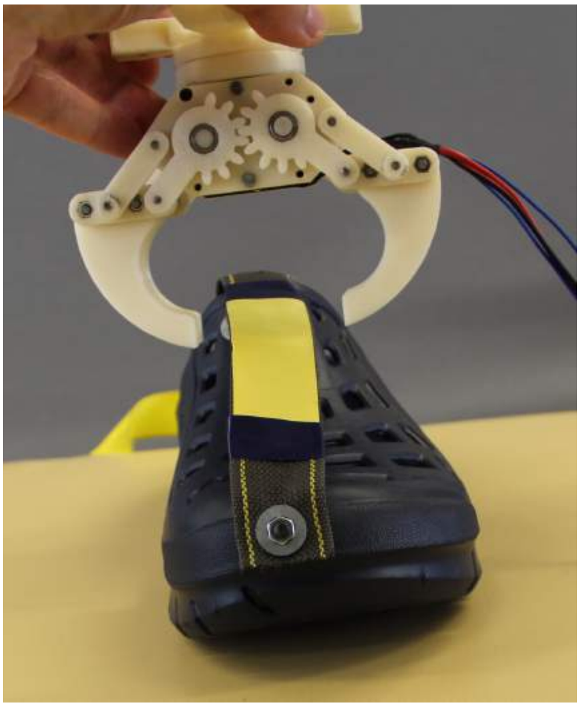


Figure 3.4: Separation distance of the gripper's fingers for picking a shoe

ribbon attached to each shoe, as shown in Fig. 3.4. More details about the ribbons attached to the shoes are provided in Section 3.2.4.

3.2 Algorithms

3.2.1 Speech recognition

Speech recognition allowed the robot to receive spoken commands to either start or finish the task, be corrected by the user or learn user preferences. The implementation of the speech-recognition interface was made through the Microsoft Speech Platform SDK 11 engine, a speech recognizer in Windows, which provides text transcriptions from spoken utterances. The microphone array from Kinect 2 acted as the input for the audio stream. It provided better sound quality and a larger detection range than a single microphone of similar technical features.

The speech-recognition process can be thought of as having a front-end and a back-end part. The front-end part processes the audio stream, isolating sound segments, and converting them into a series of numeric values that characterize the vocal sounds in the signal. The back-end part is a specialized search engine that takes the output from the front-end part and perform a search across three databases [65]:

- **Acustic model**: The acoustic model is represented by the acoustic sounds of a specific language. Kinect for Windows SDK includes a custom acoustical model that is optimized for the microphone array inside the Kinect 2.

- **Lexicon**: The lexicon is a large set of words of a given language, which provides information on how they are pronounced.

- **Grammar model**: The grammar model represents the rules by which the words are combined into specific domains.

A grammar is usually customized to recognize the utterances specific for the given application, rather than a general dialog manager. Grammars are at the core of speech recognition and can affect the recognition accuracy. Microsoft Speech Platform SDK 11 supports the option of including grammars defined in XML-format[2].

In the present work, a grammar model was created in XML-format and loaded by the recognizer for the specific task of assisted dressing. In grammars, it is convenient to link each predefined utterance with a semantic tag, which can be retrieved when the utterance is recognized. An example of the utterances and their associated semantic tags used for the assisted-dressing task is given in Table 3.2 as well as their semantic tags. Several utterances may be labeled using the same semantic tag: e.g. the words *"begin"* and *"start"* can be recognized as the same tag "start". These tags represent the actual output of the speech-recognition algorithm when an utterance is recognized, and they will be used in Section 3.2.9.

The speech-recognition algorithm is constantly running: when an utterance is recognized, the algorithm returns its semantic tag in the form of a text string.

3.2.2 Speech synthesis

Robot feedback is an important aspect of human-robot interaction as it allows the user to understand the robot current state and actions. This contributes to the user's safety, but also allows the user to correct the robot's behaviour. In this work, a speech-synthesis algorithm was implemented to inform the user about the progress of the dressing task and allow more natural interaction.

The speech-synthesis algorithm allows to know when a certain stage of the task is completed or what voice command was actually recognized by the speech-recognition algorithm. It is also required to inform the user about the necessary actions during the dressing assistance, for instance, to extend the foot towards the robot.

---

[2] Grammar models documentation: https://www.w3.org/TR/speech-grammar/

Spoken command	**Semantic tag**
"begin"	start
"fit pointing"	pointing
"take this shoe"	take
"take the blue shoe"	blue
"take the red shoe"	red
"take the green shoe"	green
"take the yellow shoe"	yellow
"dress me"	dress
"move back"	back
"move forward"	forward
"move up"	up
"move down"	down
"stop"	stop
"that's ok"	ok
"abort"	abort

Table 3.2: Spoken commands with associated semantic tags

The speech synthesis was integrated using Python and the $gTTS$ package that uses Google's Text-to-Speech API. A text string is provided as input, and the gTTS package converts it into speech transcription in $mp3$ format. The retrieved $mp3$ file containing the speech is played by speakers connected to one of the Ubuntu PCs with the *pygame* package installed.

Overall, the messages used for robot feedback can be grouped in three categories. The first type of messages was used to inform the user about the current robot state or required user action. Examples of messages of this type are given in Table 3.3.

Some messages were implemented for other reasons, for example when a recognition problem occurs. If the user requests the red shoe and the robot cannot recognize it, the message *"cannot recognize the red shoe"* informs the user about the problem. These kind of messages are meant to inform about technical problems. Other messages suggest that an unexpected user action confronts with the system logic: for instance, if the user says *"dress me"* without previously selection a shoe; the robot's response is *"I haven't picked up a shoe"*. This type of messages is used to inform the user that the predefined procedure has not been properly followed. For the sake of clarity and concise presentation, not all used messages are listed here.

3.2.3 User tracking and following

User-motion recognition was the second component of the proposed human-robot interface. The ability to track and follow user's body parts, such as a foot or a hand, was a requirement for the assisted dressing task. Specifically, the location and orientation of the foot were necessary for a proper

Fixed feedback messages
"ready to help"
"taking the #color *shoe"*
"changing to the #color *shoe"*
"the #color *shoe is picked up"*
"please, approach your foot"
"approaching the foot"
"you moved away your foot"
"moving away"
"approaching more"
"moving up"
"moving down"
"stopped, waiting for the user to finish the task"
"the task is finished, thank you"
"please point to the #color *shoe and say take this shoe*

Table 3.3: Example of messages used in robot feedback. The keyword #color can be substituted by *"blue"*, *"red"*, *"green"* and *"yellow"*.

positioning of the shoe, but also for the collision avoidance in order to keep the interaction safe. On the other hand, the location of the arm joints was necessary to recognize pointing gestures, which was performed by the user when selecting a shoe.

User tracking was performed using the Kinect for Windows SDK 2.0, a library in Windows, developed for the Kinect 2. It is a two stage process: first, a depth map is computed using the infrared sensor and the time-of-flight analysis; secondly, the user joints are segmented using a trained randomized decision forest algorithm [66], mapping depth images to body parts, as shown in Fig. 3.5.

The algorithm is able to recognize up to 6 users in the field of view of the Kinect 2, from which up to 2 users can be tracked in detail, tracking their movements in real time. The data obtained from the algorithm is a set of 25 joints per user [67], as illustrated in Fig. 3.6. Each joint is represented by its location and orientation with a quaternion, respect to the Kinect 2 reference system shown in Fig. 3.7.

Subsequently, the user's joints were transformed to the WAM robot reference system for the final integration. The calibration of the Kinect 2 and the WAM robot was important for accurate tracking and robot motion planning (for details see Section 4.1). The pointing-recognition and posture-recognition algorithms that rely on accurate user tracking are described in Sections 3.2.5 and 3.13.

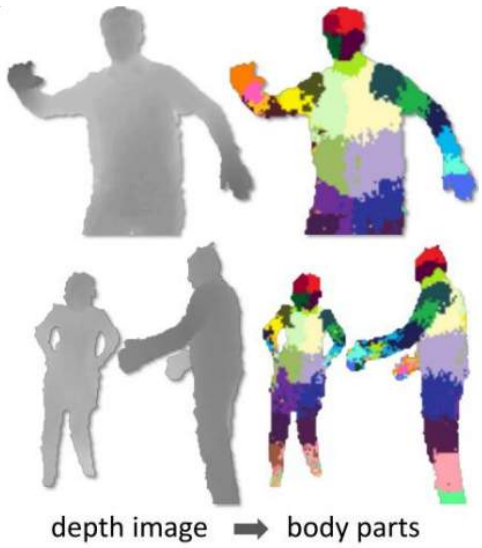


Figure 3.5: User-tracking process

3.2.4 Object color segmentation

Object recognition and manipulation are challenging problems, that require finding not only the location and the orientation of the object but also its gripping points. For example, in the assisted-dressing scenario, the gripper should not cover the opening of the shoe preventing the user to put the foot inside; instead, the gripping strategy should allow comfortable dressing.

Object manipulation was not among the objectives of this work. Therefore, for the proposed assisted-dressing scenario, shoe manipulation was simplified by attaching a ribbon to the top a shoe, as shown in Fig. 3.8, such that the gripper can grasp the shoe from above. All ribbons were $3cm$x$17cm$ in dimensions, and they had rectangular $3cm$x$6cm$ color markers placed in the central segment of the ribbon. The markers allowed recognition and location of the shoes by Kinect 1. The shoe considered for this work is a crocs shoe commonly used by patients in hospitals.

A user was able to choose from a set of four crocs shoes that were used in the experiments, and marked with blue, green, red and yellow markers, and which are shown in Fig. 3.9. A color-segmentation algorithm provided the location and the color of the recognized marker, which was used to find the position of the shoe and its gripping point.

Streaming images obtained from the Kinect 1, as the one shown in Fig. 3.10, were processed by the OpenCV image-processing library to recognize the markers in the image. The images obtained in RGB format were converted to HSV format with the following range of values:

- Hue [0,179]

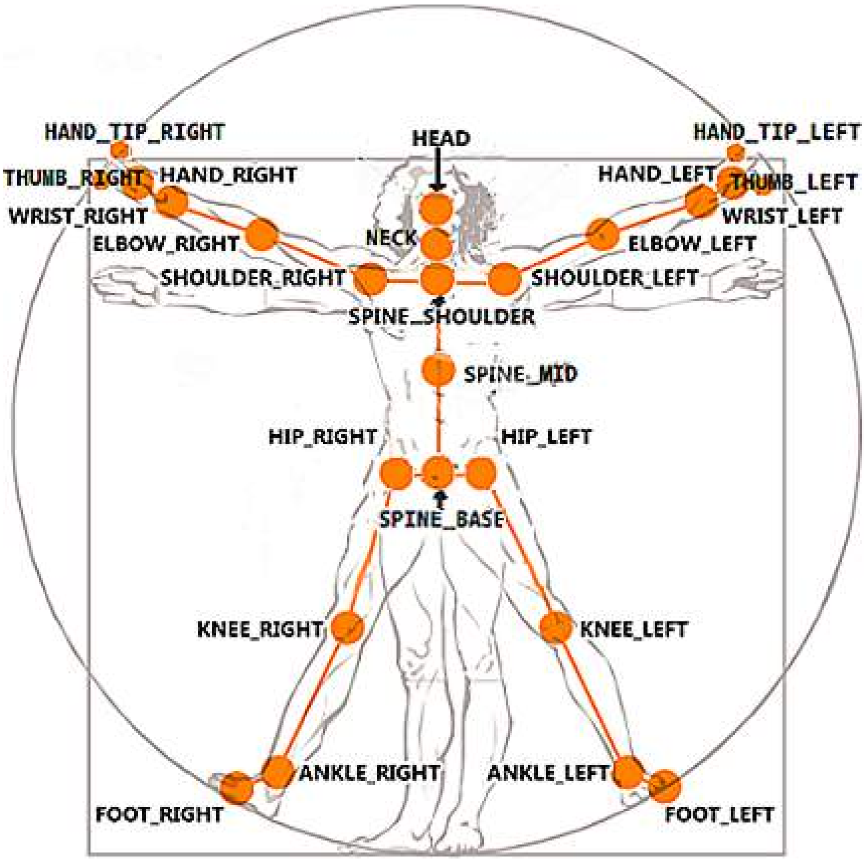


Figure 3.6: Joints provided by the user-tracking algorithm

- Saturation [0, 255]
- Value [0, 255]

Since there are four markers, one for each shoe, four different colors must be recognized. The chosen colors: blue, red, green and yellow, are defined in Table 3.4 by a range of HSV values to allow the recognition. The colors in the image were clustered according to their HSV values and their centroids

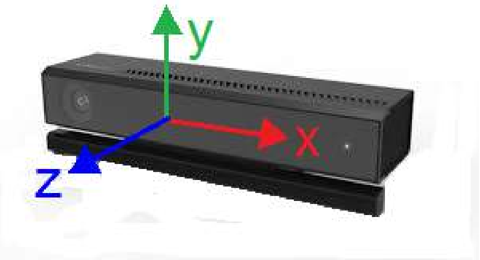


Figure 3.7: Reference system of the Kinect 2

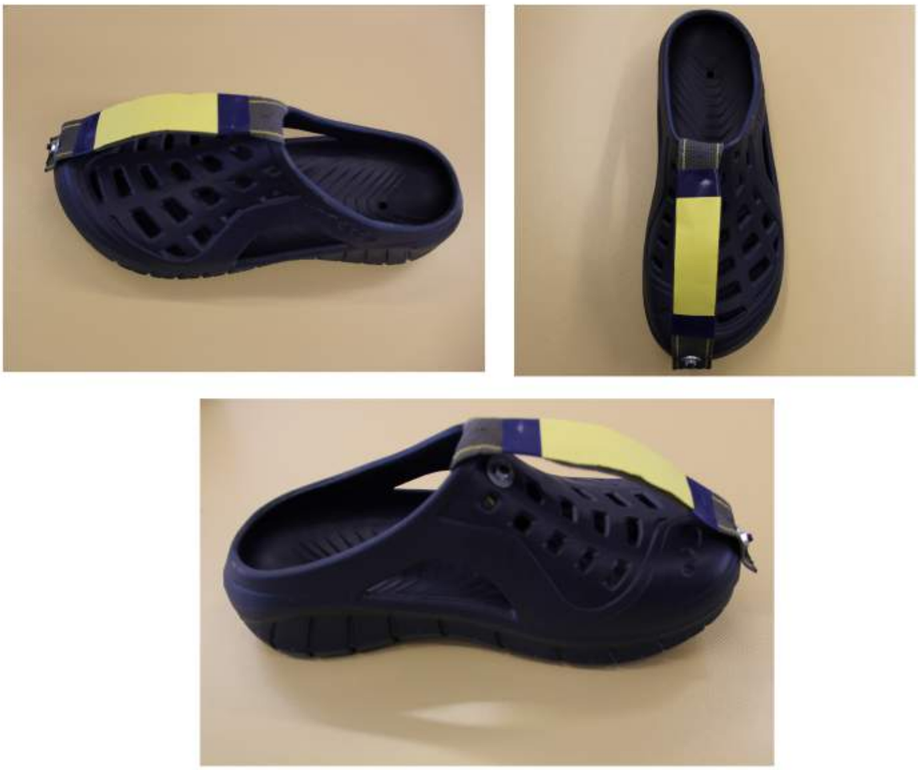


Figure 3.8: Ribbon attached to the foot with yellow color marker

	Hue	Saturation	Value
blue	[97, 110]	[70, 150]	[100, 255]
red	[167, 179]	[150 ,220]	[100, 255]
green	[48, 58]	[70, 130]	[100, 255]
yellow	[20, 30]	[110, 170]	[100, 255]

Table 3.4: HSV values for the recognition of the

were computed. Finally, the point cloud provided by Kinect 1 was used to match the coordinates (x, y, z) of the computed centroid in the 3D space in the Kinect 1 reference system, as shown in Fig. 3.11.

Every time that the user requested a shoe from the robot, the object-color-segmentation algorithm was executed returning the set of the recognized marker's positions. These positions are later transformed into the WAM robot reference system before performing the robot motion planning. The calibration of the Kinect 1 and the WAM robot was important for accurate shoe grasping. More details are in Section 4.1.

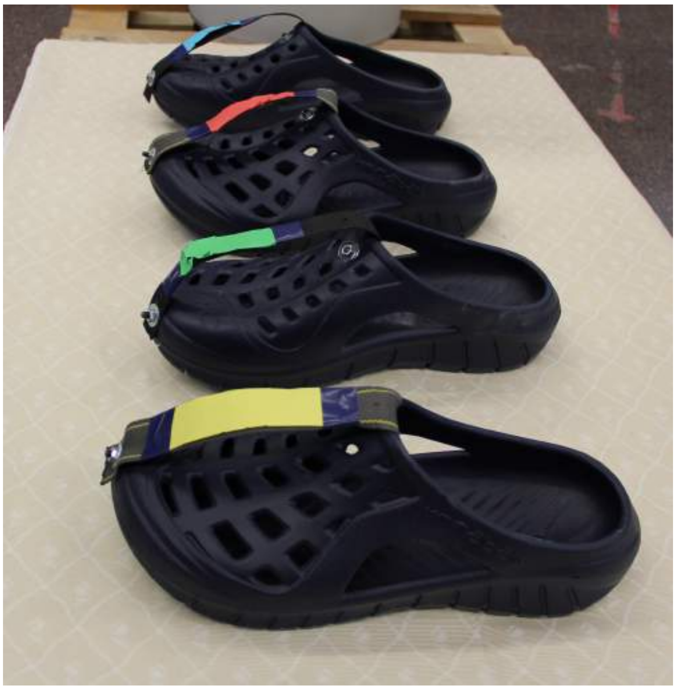


Figure 3.9: The 4 shoes used in the defined dressing assistance

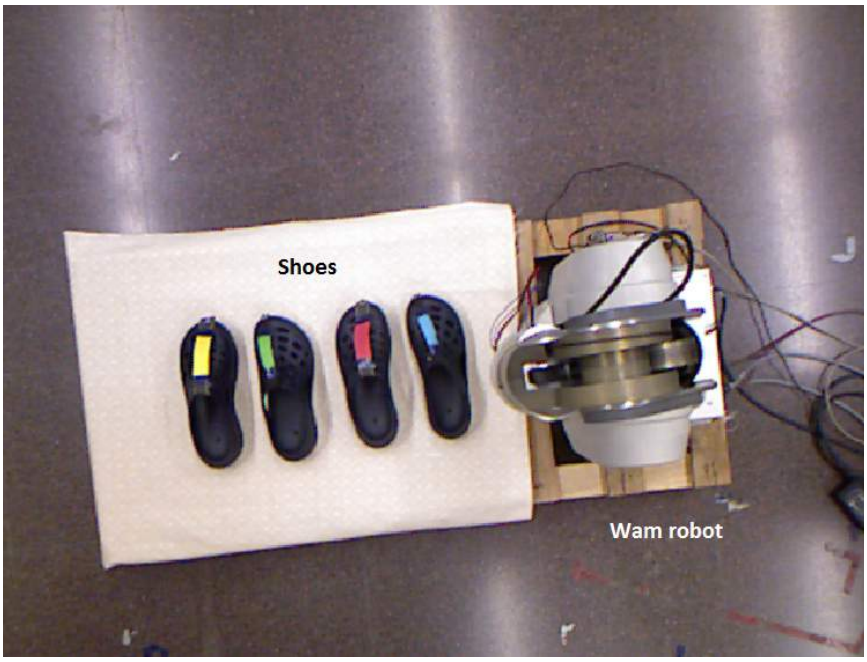


Figure 3.10: Kinect sensor 1 view for the recognition of the shoes

3.2.5 Pointing recognition

Pointing recognition was a second modality used for shoe selection and it required a combination of speech and pointing gesture to form the so-called deictic expressions. The reason for implementing two different modalities, i.e. speech and pointing recognition, was to offer more flexibility to the user. In fact, the reference to the color using speech seems to be easier and simpler when distinguishing the shoes. However, the pointing gesture provides an alternative solution in real life situations when the colors might not be enough to discriminate different objects, e.g. when there is more than one blue shoe. In this case, the user should point to the desired shoe with the right arm and say *"take this shoe"*. Once the voice command is recognized, the algorithm computes the pointing target in the intersection with the shoes plane. The selected shoe is the closest one to the pointing target.

The computation of the target from the pointing gesture can be done with some combinations of joints such as the wrist and the elbow, the wrist and the shoulder, or the wrist and the head, providing good results [68]. The pointing-control interaction modality uses the location of the wrist and the elbow joints to calculate the pointing target, as in [69, 70].

All following computations are performed in the robot reference system, which is defined in detail in Section 4.1. Hence, let $p_e = (x_e, y_e, z_e)$ be the position of the user's elbow and $p_w = (x_w, y_w, z_w)$ the position of the user's wrist, both locations obtained from the user-tracking algorithm (see Section 3.2.3). A straight line is computed using these two positions:

$$\begin{cases} x = x_e + \lambda(x_w - x_e) \\ y = y_e + \lambda(y_w - y_e) \quad \text{where } \lambda \in \Re \\ z = z_e + \lambda(z_w - z_e) \end{cases} \tag{3.1}$$

where λ is the straight line parameter and can take any value. A general definition of a plane in the 3-dimensional space is given by:

$$Ax + By + Cz + D = 0 \tag{3.2}$$

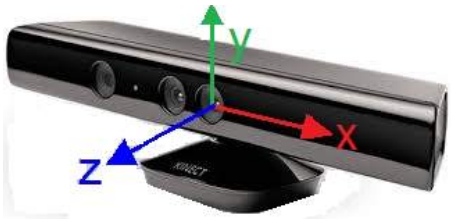


Figure 3.11: Reference system of the Kinect sensor 1

In the proposed dressing scenario, the shoes are placed on a platform that is parallel to the ground floor ($z = 0$), so in this case the equation 3.2 can be reduced to $z = h$, where h is the height of the plane from the origin.

The intersection point between the straight line and the shoe plane provides the pointing target of the user $p_t = (x_t, y_t, z_t)$, given by:

$$\begin{cases} x_t = x_e + \frac{h-z_e}{z_w-z_e}(x_w - x_e) \\ y_t = y_e + \frac{h-z_e}{z_w-z_e}(y_w - y_e) \\ z_t = h \end{cases} \tag{3.3}$$

Finally the closest shoe to the pointing target is selected. Let $S = \{\text{blue, red, green, yellow}\}$ be the set of the different shoes in the plane, and p_i, where $i \in S$, their location in the plane. These locations are obtained from the object-color-segmentation algorithm described in Section 3.2.4. The closest shoe $s \in S$ from the target point is given by:

$$s = \arg\min_{s \in S}[\text{dis}(p_t, p_s)] \tag{3.4}$$

An illustration of shoe selection through pointing recognition is given in Fig. 3.12. In the given example the closest shoe is the blue one, which is the one selected by the user.

3.2.6 Posture recognition

The posture-recognition algorithm was developed to recognize the user's extended foot, and its contribution was twofold. First the algorithm contributes to user's safety. If the robot makes a forceful physical contact with the user while approaching, the user is able to withdraw the foot, so the robot can stop the task. The second reason is to know the user's intention. If the user wants to stop the dressing task for a moment, or does not want to be dressed anymore, withdrawing the foot means that the intention of the user to be dressed has hanged. The robot should not try to put on the shoe when the user is not focused on the dressing task. This awareness is essential in robot assistance, and for this specific task, the extended foot plays a crucial role in the user's intentions.

The posture recognition takes place after the robot picks up a selected shoe and the user says *"dress me"* to start the dressing part. From this moment, the robot will wait for the user to extend the right foot in order to start the approach and assist with dressing.

Some algorithms such as Dynamic Time Warping (DTW) [71] were considered at the beginning to recognize dynamic postures with time independence. However, the final posture considered is static and therefore a more simple algorithm could be applied. The recognition of the extended foot posture was only performed on the right foot, and not the left one. Although the same methodology could be applied to the recognition of the left

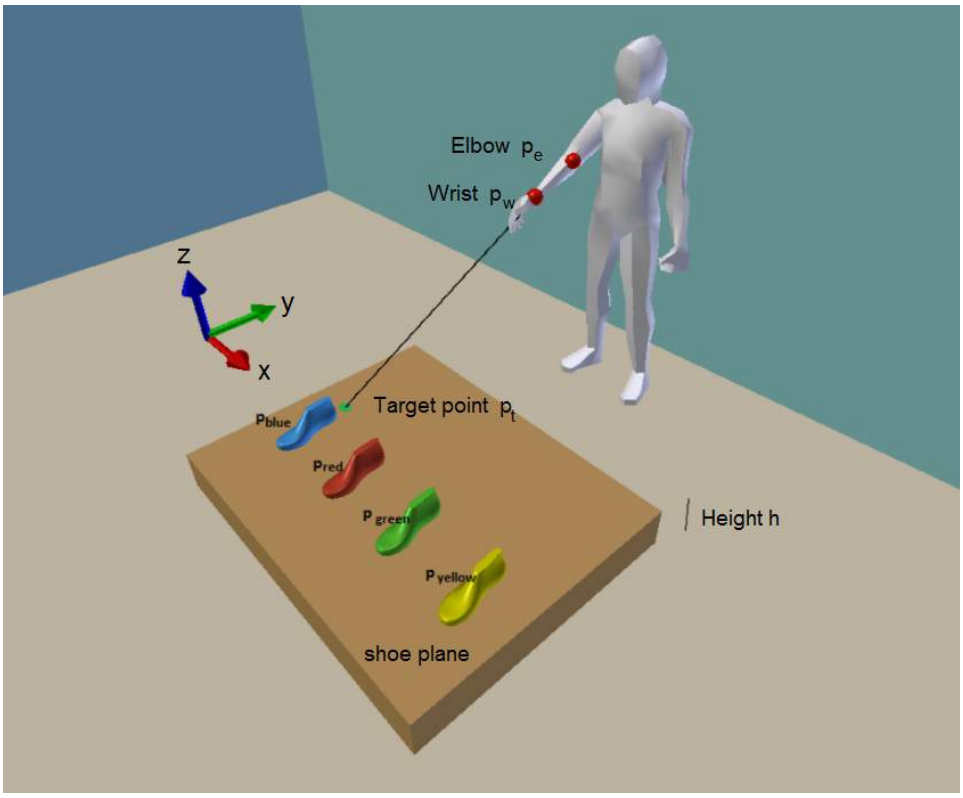


Figure 3.12: Representation of the target point calculation

foot, the dressing task was limited to putting on the right shoe only. Further details are given in Section 3.2.9.

The recognition is considered as a binary problem: the foot can be either extended or not. This is done by computing the ankle position in the knee frame reference using a transformation with the tf package in ROS[3]. To determine whether the posture is recognized, the y coordinate of the ankle position from the knee frame needs to be larger than the threshold $0.05m$, as shown in Fig. 3.13. For the opposite, i.e. when the foot is folded, the posture is not recognized.

The algorithm is running all the time during the assistance, returning whether the posture is recognized or not. This allows the robot to know in any moment if the user's foot is extended, allowing it to decide what to do in each situation.

---

[3] tf transformations source:
http://wiki.ros.org/tf/Overview/Transformations

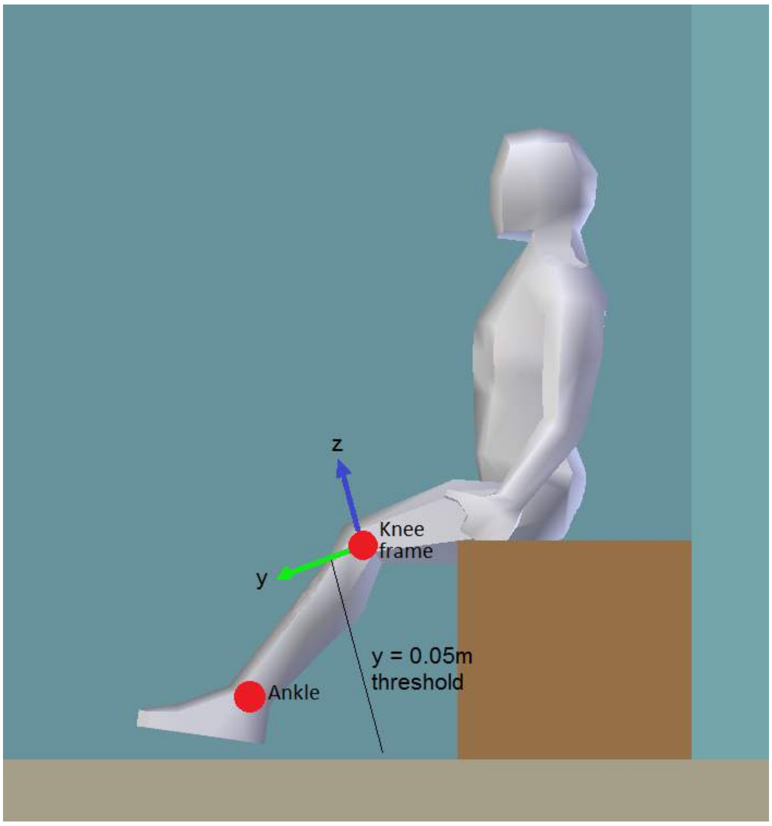


Figure 3.13: Threshold of the posture recognition

3.2.7 Robot motion planning

The WAM robot movements were computing by the inverse kinematics algorithm operating in the compliant mode, and contributed to the user's safety. In case of physical contact, the robot reduces the force applied against the user. On the other hand, the inverse kinematics allows to move the last joint WAM robot (where the gripper is attached) to a certain point in space, without worrying about the positions of the WAM robot's joints.

Some fixed positions of the robot were defined for the dressing task. All positions are represented in the WAM robot reference system, which is described in detail in the Section 4.1. The *home position* of the robot end-effector was set to $(x, y, z) = (0.1m, 0m, 0.2m)$ with the initial orientation defined by the quaternion $q = (0, 0, 1, 0)$, as shown in Fig. 3.14. With this orientation, the gripper is facing down, ready to pick up a shoe. The *home position* was chosen to avoid possible occlusions of the shoes in the Kinect 1 view.

When the user requests a shoe, the robot checks whether a shoe was already taken or not. If no shoe was picked up, the robot moves to the

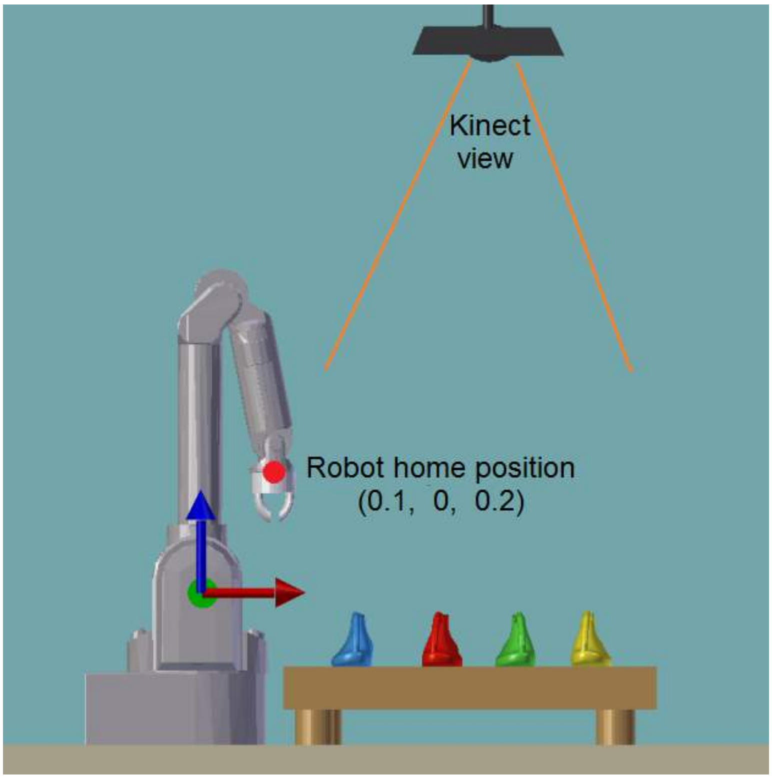


Figure 3.14: Robot home position and kinect view from a front view of the scenario

position of the requested shoe making a trajectory through 3 waypoints. These positions have the same location of the requested shoe in the xy plane, but different heights: the first one at $0.20m$, the second one at $0.10m$ and the third one at $0.05m$ from the shoe, as shown in Fig. 3.15.

The first waypoint was chosen to avoid any possible collision of the WAM robot arm with other shoes during the movement. The inverse kinematics algorithm in the compliant mode achieves safer operation at the expense of lower accuracy. Subsequently, the second waypoint was used for the WAM robot to go down in the z direction with no deviation. The third waypoint is where the gripper's fingers are below the ribbon such that the gripper may close and grasp the shoe. Once the shoe is picked up, the robot moves through the same three waypoints to return to the *home position*, avoiding to hit any other shoe during the movement. The initial location of the selected shoe is stored, so that if the user requests another shoe, the robot can return the picked-up shoe to its original location. When the robot places the shoe by opening the gripper, it moves for the new requested shoe following the three waypoints defined for the new location. The orientation of the gripper was set to be perpendicular with the shoes in order to perform flawless grasping.

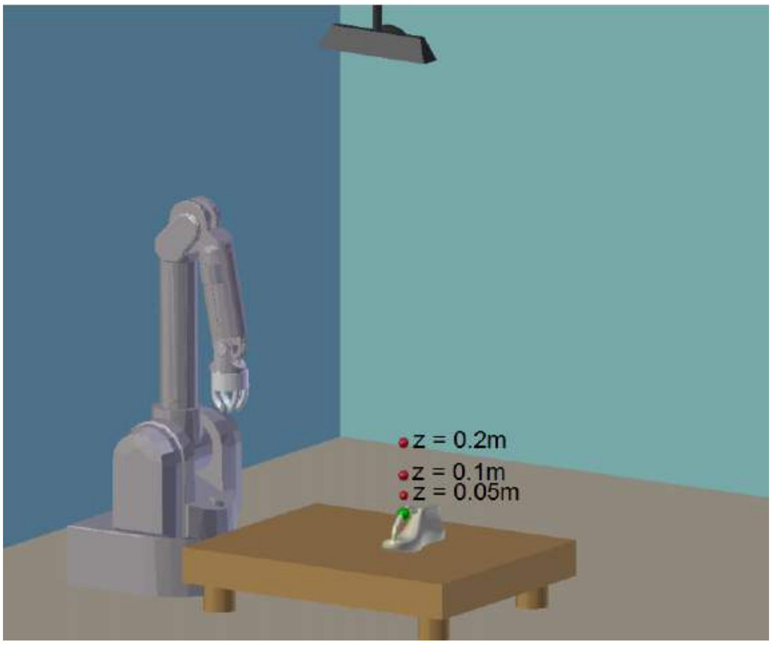


Figure 3.15: Three waypoints of the robot motion when picking and place a shoe

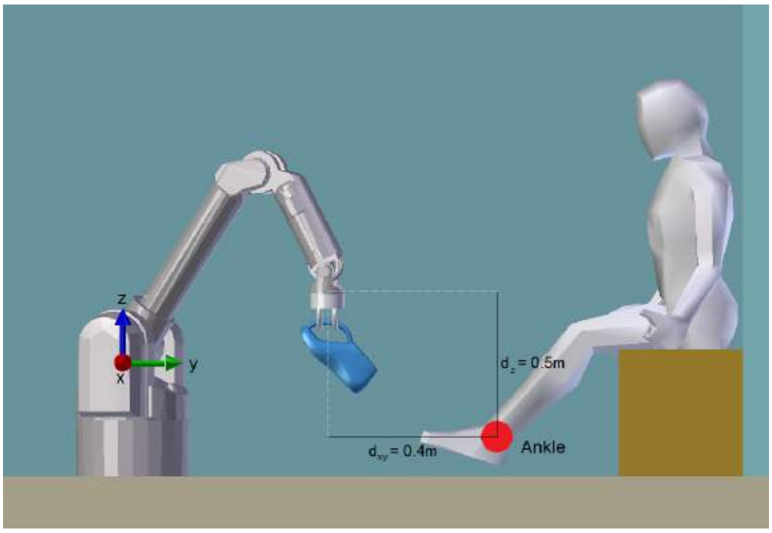


Figure 3.16: Way points for the picking phase

Finally, when the robot picks up the shoe selected by the user and after recognizing a *"dress me"* command, it approaches the user's foot. From the home position the robot moves to a prudent distance from the ankle, which is $d_{xy} = 0.4m$ in the xy plane (taking into account the orientation of the foot) and $d_z = 0.5m$ in the z axis, as shown in Fig. 3.16.

Computing the orientation of the extended foot in the xy plane is re-

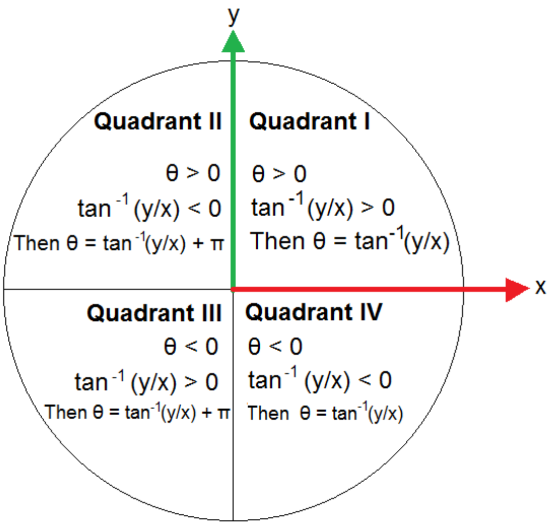


Figure 3.17: Convention used to compute knee-ankle angle in the robot frame of reference

quired for the motion planning of the WAM robot. This is done using the ankle and knee joints provided by the user-tracking algorithm (Section 3.2.3). Although the foot joint is also provided, in practice its recognition was unreliable.

The orientation of the extended foot is described by the angle formed by the x axis and the straight line given by the ankle and the knee joints. Before getting into more details, the convention followed for angles must be defined: all angles were measured from the x axis, in the positive direction. In the quadrants I and II, angles take positive values, and for the quadrants III and IV, negative. Angles are computed from coordinates applying trigonometric functions, however these functions are dependent on the coordinates signs. Using the tan^{-1} trigonometric function, wrong values are obtained in quadrants II and III, as summarized in Fig. 3.17. Then, different expressions are required for these quadrants.

Let $p_a = (x_a, y_a, z_a)$ and $p_k = (x_k, y_k, z_k)$ be the positions of the user's ankle and knee respectively. The position of the ankle with respect to the knee $p_a^{(k)}$ is given as follows,

$$p_a^{(k)} = p_a - p_k \tag{3.5}$$

The angle formed between the knee and the ankle with the x axis is

$$\beta = tan^{-1}\left(\frac{y_a^{(k)}}{x_a^{(k)}}\right) + \begin{cases} \pi & \text{if } (x_a^{(k)} < 0) \text{ and } (y_a^{(k)} > 0) \text{ (quadrant II)} \\ -\pi & \text{if } (x_a^{(k)} < 0) \text{ and } (y_a^{(k)} < 0) \text{ (quadrant III)} \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

The cases when $x_a^{(k)} = 0$ are not represented in this equation, but are considered in the same way by looking at the sign of the $y_a^{(k)}$ coordinate. With this angle and knowing the distance at which the gripper must be in both the xy plane and the z axis, the robot position $p_r = (x_r, y_r, z_r)$ is given by equation 3.7

$$p_r = \begin{cases} x_r = x_a + d_{xy} \cos\beta \\ y_r = y_a + d_{xy} \sin\beta \\ z_r = z_a + d_z \end{cases} \tag{3.7}$$

The WAM robot position is updating at all times, meaning that if the user moves the foot, the robot will follow it from the predefined distance.

3.2.8 User adaptation

User adaptation was developed to provide personalized assistance to each user. It consists of two algorithms: pointing calibration, and robot position adjustment. Pointing calibration was developed to improve the accuracy of pointing. On the other hand, the robot position adjustment was implemented to improve the comfort of the users by placing the shoe at a chosen distance from the foot. The use of adaptation allowed a personalized robot setup for each user.

A Pointing calibration

Regarding the pointing-recognition algorithm described in Section 3.2.5, problems can appear when the user aims to point a shoe that is surrounded by others: for instance the green shoe in Fig. 3.9. Due to the fact that the shoes might not be separated by a large distance, and that the user is not presumed to be very accurate with the pointing gesture, the pointing recognition algorithm can select a wrong shoe. In addition, the way to point may differ from one person to another. For this reason, a pointing calibration method is proposed in order to adapt the pointing recognition for a particular user.

The pointing calibration corrects the angle of the user's pointing direction, seen in the shoe plane $z = h$ (defined in Section 3.2.5), and formed by the x axis and the straight line passing through user's elbow and wrist, with the origin in the elbow joint. The correction was meant to fit the pointing target closer to the desired shoe. The convention followed for angles was the same as followed in Section 3.2.7, shown in Fig. 3.17.

An specific case from a top view is shown in shown in Fig. 3.18. In the given example, the user is trying to pick up the blue shoe but with out low accuracy, pointing closer to the red shoe. The two black points represent the user's elbow and wrist locations. The pointing target p_t is closer to

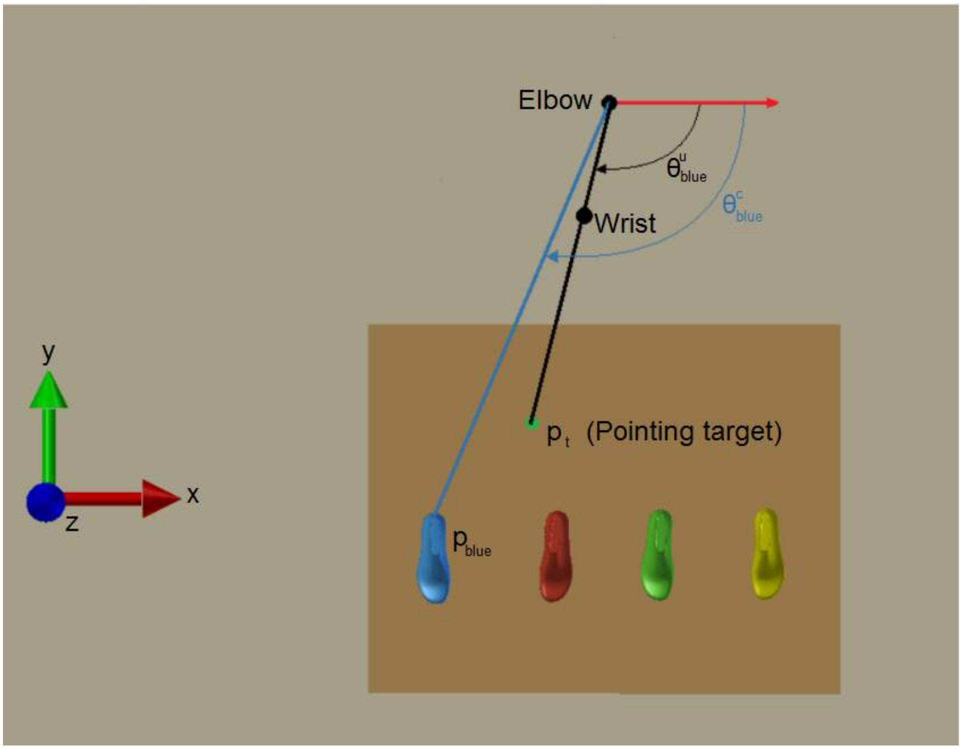


Figure 3.18: Representation of the target point calculation

the red one in this case. The angle θ^u_{blue} is formed by the elbow and the wrist, taking the origin from the elbow. The angle θ^c_{blue} is given by the line which connects the elbow and the blue shoe positions, from the elbow. In the followed convention, these angles are negative. The pointing-calibration algorithm aims to correct the pointing gesture of the user, and later to make a correction from the θ^u_{blue} to the θ^c_{blue} angle to correct the position of the target point p_t.

Before correcting the user angle in the pointing recognition, a calibration by the user must be performed. The pointing-calibration algorithm assumed that the user's elbow was in a fixed position, or at least almost static during the dressing assistance. First, a calibration must be performed by the user in order to adjust the pointing gesture to the pointing recognition. The procedure is the following: the user has to say first the voice command *"fit pointing"*. Then the robot requests the user to point out all the different available shoes (saying the command *"take this shoe"* for each shoe), in this case the blue, red, green and yellow shoes, in this order. The method computes two angles for each shoe: the user angle and the corrected angle of the shoe, both from the user's elbow: θ^u_s and θ^c_s where $s \in \{\text{blue, red, green, yellow}\}$.

For simplification, let $p_{user} = (x_{user}, y_{user}, z_{user})$ be the difference between the wrist and elbow positions: $p_{user} = p_w - p_e$, and $p_{cor} = (x_{cor}, y_{cor}, z_{cor})$ the difference between the shoe position and the elbow $p_{cor} = p_s - p_e$. Thus, the computation of the user angle θ_s^u and the shoe angle θ_s^c is given by equations 3.8 and 3.9 respectively.

$$\theta_s^u = tan^{-1}\Big(\frac{y_{user}}{x_{user}}\Big) + \begin{cases} \pi & \text{if } (x_{user} < 0) \text{ and } (y_{user} > 0) \text{ (quadrant II)} \\ -\pi & \text{if } (x_{user} < 0) \text{ and } (y_{user} < 0) \text{ (quadrant III)} \\ 0 & \text{otherwise} \end{cases} \tag{3.8}$$

$$\theta_s^c = tan^{-1}\Big(\frac{y_{cor}}{x_{cor}}\Big) + \begin{cases} \pi & \text{if } (x_{cor} < 0) \text{ and } (y_{cor} > 0) \text{ (quadrant II)} \\ tan^{-1}\Big(\frac{y_{cor}}{x_{cor}}\Big) - \pi & \text{if } (x_{cor} < 0) \text{ and } (y_{cor} < 0) \text{ (quadrant III)} \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

where s represents the shoe color ($s \in \{\text{blue, red, green, yellow}\}$). Now it is assumed a linear relationship between the two set of angles θ_s^c and θ_s^u. The reason of this assumption is that the pointing gesture of the user is presumed to be the same for all the shoes. A linear fitting is computed between the two sets of angles in order to extract the linear parameters A and B, given by:

$$\theta^c = A\theta^u + B \tag{3.10}$$

The parameters A and B are stored and used to correct the pointing target in future pointing recognitions. Every time the user uses the pointing gesture to select a shoe, the user angle θ^u is computed using equation 3.8. Then the corrected angle θ^c is retrieved applying the equation 3.10. The pointing target must be shifted to the corrected angle: basically, polar coordinates in the shoe plane $z = h$ are applied, taking the user's elbow as the origin. First the distance of the pointing target is computed as follows,

$$d = \sqrt{(x_t - x_e)^2 + (y_t - y_e)^2} \tag{3.11}$$

Finally the corrected target point $p_t^c = (x_t^c, y_t^c, h)$ is computed:

$$p_t^c = \begin{cases} x_t^c = d\cos\theta^c \\ y_t^c = d\sin\theta^c \end{cases} \quad z_t^c = h \tag{3.12}$$

The process of correction the pointing recognition is summarized in Fig. 3.19.

The calibration method was optional, therefore if no calibration was performed by the user, the fitting parameters were set to their default values $A = 1$ and $B = 0$, resulting in $\theta^c = \theta^u$. In that case, no correction was done to the pointing target.

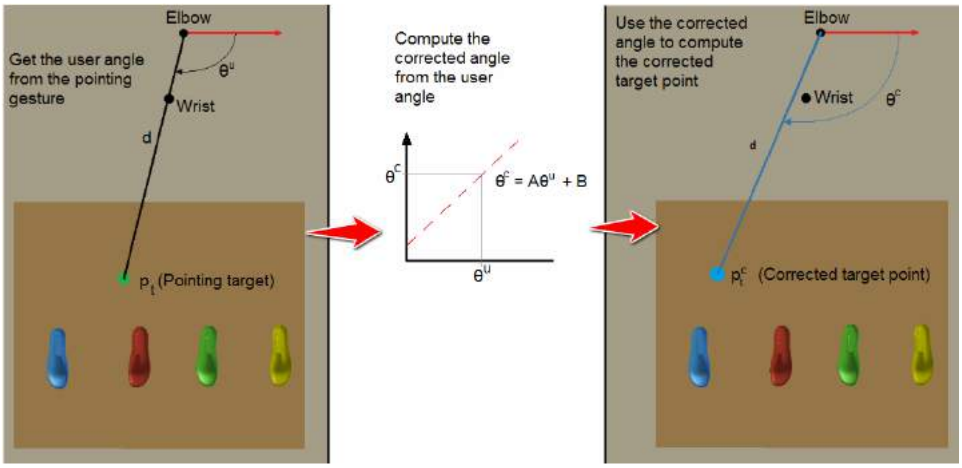

Elbow
Get the user angle from the pointing gesture
θ^u
Wrist
d
p_t (Pointing target)
Compute the corrected angle from the user angle
θ^c
$\theta^c = A\theta^u + B$
θ^u
Use the corrected angle to compute the corrected target point
Elbow
Wrist
θ^c
d
p_t^c (Corrected target point)

Figure 3.19: Proccess of the pointing-recognition correction. First, the user angle θ^u is computed from the pointing gesture (left). Second, the corrected angle θ^c is retrieved from the linear fitting performed in the calibration (middle). Finally, the corrected angle is used to shit the pointing target.

B Robot position adjustment

A predefined position of the robot when delivering the shoe may not be comfortable all the users. The user tends to perform as less physical effort as possible during the assistance, and the position of the shoe may not be adequate. For this reason, an adjustment algorithm for the distance between the WAM robot position and the foot was developed. The user can modify the robot position using the commands *"move forward"*, *"move back"*, *"move up"* and *"move down"*, recognized by the speech-recognition algorithm described in Section 3.2.1. Once recognized, the robot continuously moves in the selected direction until the user says *"stop"*. In this way, both distances in the xy plane and z axis can be adjusted, making the autonomous robotic system able to personalize the assistance for a given person. The adjustment is performed once, such that the new distance to the foot is stored and used for next assistance.

The process of the robot position adjustment is the following: let v_{xy} be the velocity at which the robot moves during the adaptation in the xy plane and v_z the velocities in the z axis. The position of the robot during the adjustment is given by:

$$p_r(t) = \begin{cases} x_r(t) = x_a + (d_{xy} + v_{xy}t)\cos\theta \\ y_r(t) = y_a + (d_{xy} + v_{xy}t)\sin\theta \\ z_r(t) = z_a + (d_z + v_z t) \end{cases} \tag{3.13}$$

When no adjustment is performed, the speeds values are set to zero by default, such that the robot position p_r returns the same position as in equation 3.7

When the command *"move forward"* is recognized, the velocity in the xy plane is set to $v_{xy} = 0.01m/s$ and the velocity v_z set to zero. The $0.01m/s$ value is meant to be slow enough taking into account the delay in the speech recognition of the command *"stop"*, which take approximately one second. According to the equation 3.13 the robot moves away from the foot along the direction formed by the knee and the ankle, following the foot at the same time. This happens until the *"stop"* command is recognized and both velocities v_{xy} and v_z are set to zero. The same process is followed with the command *"move back"*, but now the speeds values are set to $v_{xy} = -0.01m/s$ and $v_z = 0$. In this case the robot approaches the foot until the *"stop"* command sets both velocities to zero.

The *"move up"* voice command set the following speeds values: $v_{xy} = 0$ and $v_z = 0.01\text{m/s}$. The robot moves up along the z axis, until the *"stop"* command is recognized and set both velocities to zero. The same procedure is performed with the *"move down"* voice command, but setting both velocities to $v_{xy} = 0$ and $v_z = -0.01\text{m/s}$. In this way the robot moves down along the z direction, until the *"stop"* command set the speeds values to zero.

Taking into account the above description, if the assisted person first says *"move down"*, and then *"move forward"*, the first command is canceled by the second command, and the robot moves forward. This behaviour is the same for all combinations of the four voice commands.

The position-adjustment algorithm offers the adaptability of the robot position with respect to to the foot, following equation 3.13. The adjustment needs to be performed only once per user, such that adjusted position of the robot is stored to be used during next assistance.

3.2.9 Decision module

To process all the information coming from previous algorithms, a Finite State Machine (FSM) was implemented to deal with all the actions and decisions made by the system. A diagram of the FSM is shown in Fig.3.20. The FSM has a total of 8 states, which are: Aborted, Stopped, Pick phase, Wait for posture, Follow foot, Wait to finish, Finish and Pointing. The transitions between the states are deterministic evoked by events such as a voice command or a recognized posture. The voice commands are interpreted by their semantic meaning tags as described in Section 3.2.1, shown in Table 3.2. These tags are the actual data that the FSM receives from the speech recognition, and it is used from now. On the other hand the posture recognition of the extended foot explained in Section 3.13 is binary, which means that it is either recognized or not.

A description of each FSM state is given below.

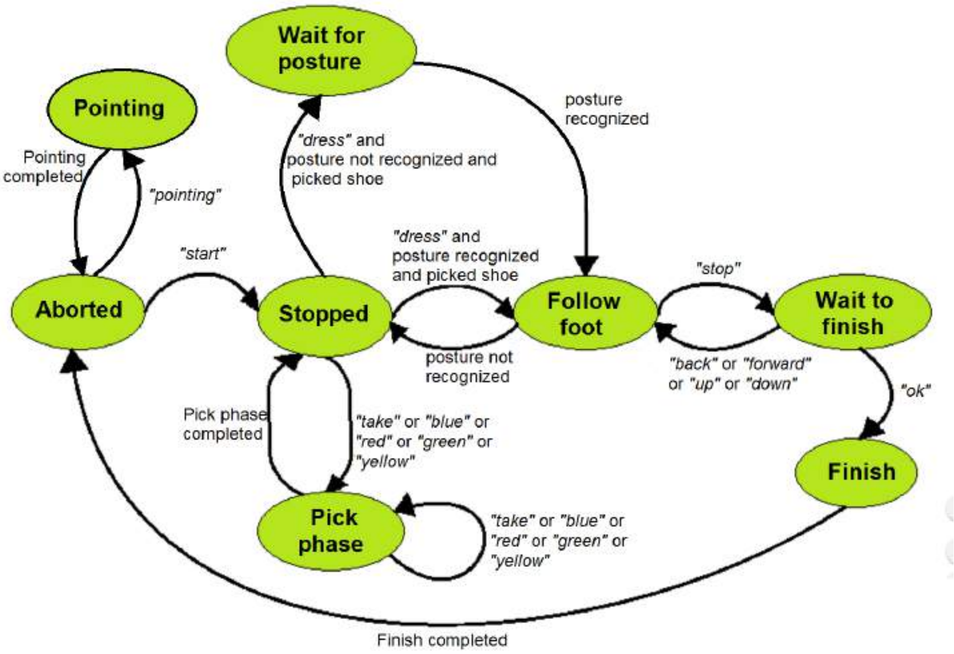

Wait for posture
Pointing
posture recognized
Pointing completed
"pointing"
"dress" and posture not recognized and picked shoe
"start"
"dress" and posture recognized and picked shoe
"stop"
Aborted
Stopped
Follow foot
Wait to finish
posture not recognized
"back" or "forward" or "up" or "down"
"ok"
Pick phase completed
"take" or "blue" or "red" or "green" or "yellow"
Finish
Pick phase
"take" or "blue" or "red" or "green" or "yellow"
Finish completed

Figure 3.20: Implemented FSM of the dressing assistance

- **Aborted:** Initially, the autonomous robotic system is in the aborted state. In this state only the *"start"* and *"pointing"* commands can perform a transition. Meanwhile, the robot does not perform any task, being stopped in its *home position.* The aborted state is also thought as a prevention in case something goes wrong: although it is not indicated in Fig. 3.20, there is a transition from any state to the aborted state, using the voice command *"abort"*.

- **Pointing**: The pointing state makes a transition from aborted with the command *"pointing"*. In this state the calibration algorithm of the pointing recognition, described in Section 3.2.8 part A, is performed. The robot requests the user: *"point out to the blue shoe and say take this shoe"* with the speech-synthesis algorithm explained in Section 3.2.2. The user must point to the blue shoe and say *"take this shoe"*, repeating the same proceed for the red, green and yellow shoes. Once the calibration is completed the FSM returns to the aborted state, storing the pointing calibration of the user for future assistance.

- **Stopped** When the *"start"* command is recognized during the aborted state, a transition to stopped state is made. The robot lets know the user that it is ready to perform the assistance with the message *"ready to help"*. If the robot has not picked any shoe, therefore the only state available is the pick phase. The user is able to choose between the different shoes by saying one of the five following commands: *"take"*, *"blue"*, *"red"*, *"green"* and *"yellow"*. If the user uses *"take"*, then the system computes the closest shoe from the pointing gesture as explained in Section 3.2.5. Any other voice command of these five request the shoe with the related color, e.g. *"blue"* for the blue shoe. When the robot has picked a shoe, two more states are available from the stopped state: wait for posture and follow foot. For both states the voice command *"dress"* has to be said, but the recognition of the extended foot posture decides whether going to one state or to the other.

- **Pick phase:** Any of the five commands *"take"*, *"blue"*, *"red"*, *"green"* and *"yellow"* sets the state of the FSM to pick phase from the stopped state. In this state the system tries to recognize the requested shoe using the object-color-segmentation algorithm described in Section 3.2.4. Once the location of the color marker is found, the robot gives the feedback *"taking the "* #color *shoe"* and moves to the requested shoe's position through the 3 waypoints defined in Section 3.2.7. When the gripper grasps the shoe, the robot returns to its *home position* through the 3 waypoints. Then the object color-segmentation algorithm tries to recognize the same color again to ensure that the shoe was actually taken: if the color is recognized, then the shoe was not picked up; otherwise the system stores the previous location of the shoe and considers that it was picked up. Finally the FSM returns to the stopped state again, informing the user that the shoe was picked successfully with the message *"the* #color *shoe is picked"*.

  While the robot is in the pick phase the user is able to correct the robot if a wrong shoe is being picked up, just saying one of the five previous commands. In that case the robot re-instantiates the pick phase with the new requested shoe. This feature is represented as a self transition in the pick phase.

  As described in Section 3.2.7, if the user requests a new shoe and a shoe is already taken, the robot places the grabsped shoe to its previous stored location. Once the shoe is placed, the robot starts the picking process for the new shoe.

- **Wait for posture:** If the user says *"dress me"* during the stopped state and the extended foot posture is not recognized, the state of the FSM is set to wait for posture, followed by the feedback *"please, approach your foot"*. As its name indicates, this state is meant to wait until the user extends the foot towards the robot. When the posture is recognized, the state changes to the follow foot state.

- **Follow foot:** Once the extended foot posture is recognized, either during the stopped or the wait for posture states, the current state is set to follow foot. The robot approaches the user's foot, as explained in Section 3.2.7, with the message *"approaching the foot"*. Then the FSM remains in this state to allow the user-guided. The adjustment of the robot position, described in Section 3.2.8 part B, by using the voice commands *"back", "forward", "up"* and *"down"*. Every time one of these commands are recognized, the robot gives the corresponding feedback:

  - *"moving away"* for the command *"back"*
  - *"approaching more"* for the command *"forward"*
  - *"moving up"* for the command *"up"*
  - *"moving down"* for the command *"down"*

  The WAM robot follows the foot until either the user withdraws his or her foot or the voice command *"stop"* is recognized. If the user withdraws the foot, the extended foot posture is not recognized, coming back to the stopped state while the robot returns to its *home position* with the feedback *"you moved away your foot"*. If this happens, the user is able to use the command *"dress"* to set the follow foot state again.

- **Wait to finish:** If the command *"stop"* is recognized, the FSM state is set to wait to finish. At this moment, the robot stops following the foot and stops in the current position providing the message *"stopped, waiting for the user to finish the task"*. This state is created to wait until the user puts the foot inside the shoe. The command *"ok"* makes a transition to the last finish state. However, the user might not be satisfied with the adjustment of the robot position from the foot. The voice commands *"back", "forward", "up"* and *"down"* can be used again to return to the follow foot state.

- **Finish:** Once the user introduces the foot inside the shoe and the voice command *"ok"* is recognized, the finish state is set in the FSM. The gripper opens to release the shoe, and the robot returns to its *home position*, while providing the feedback *"the task is finished, thank you"*. When the robot reaches the *home position*, the state is again set to abort, allowing the robot to be ready for a new dressing task.

3.3 Integration in ROS

The current project was implemented using the Robot Operating System (ROS). ROS is a widely-used platform that provides libraries and tools for development of robot applications. It has become a standard development platform in the robotics academic community, and rapidly adopted by the representatives from industry. The major benefit from using ROS is the easy integration and data transfer between different components of a system, which are structured in so-called ROS nodes. ROS is licensed under an open source BSD license[4], providing hardware abstraction, visualizers, message-passing and package management, among others capabilities. Several distributions were developed for Linux operating system. In this work, ROS Hydro[5] was installed in both Ubuntu PC

Data from the Kinect 1 is obtained using the OpenNI library[6], and open-source SDK used for development of applications based on RGB and depth images. The integration with ROS is done using the Openni_kinect [7], an open-source library used for the integration of PrimeSense sensors with robotic systems, which are used for the recognition and localization of the shoes in the assisted-dressing scenario.

The Kinect sensor 2 can be integrated with ROS using the newer OpenNI2 framework[8]. However, at the time of this project, user-tracking was not provided. For this reason, Kinect for Windows SDK 2.0, running on Windows operating system was used. The application providing the user skeleton tracking was written in C++ using Visual Studio 12 IDE. The integration with ROS was enabled using the rosserial_windows package[9], which allowed sending the Kinect 2 data from a Windows-based PC to a PC running on Linux through a communication socket. A ROS node running on Linux-based PC received the data from the socket and published them to the ROS framework for other nodes to use them.

The WAM robot and the gripper drivers were compatible with ROS, which allowed their easy integration and customization. ROS offers an easy integration of both software and hardware, but also the multi-threading capability required in robotic applications, where different threads are used to wait for different events to occur.

The implemented algorithms distributed along different ROS nodes are represented in Fig. 3.21. The red color indicates the implementation of the node in the Windows-based PC, and the blue color in the Linux-based PCs.

The different ROS nodes are described bellow, while detailed ROS specifications and developed code can be found in Appendix A.

- **Kinect 2 data socket**: On the Windows-based PC, a data socket that was developed provides the communication between the Kinect 2 and ROS framework. Both speech-recognition (Section 3.2.1) and user-

---

[4] ROS: http://wiki.ros.org/

[5] ROS Hydro: http://wiki.ros.org/hydro

[6] OpenNI: http://openni.ru/index.html

[7] Openni_kinect package: http://wiki.ros.org/openni_kinect

[8] OpenNI2: http://structure.io/openni

[9] Rosserial package: http://wiki.ros.org/rosserial_windows

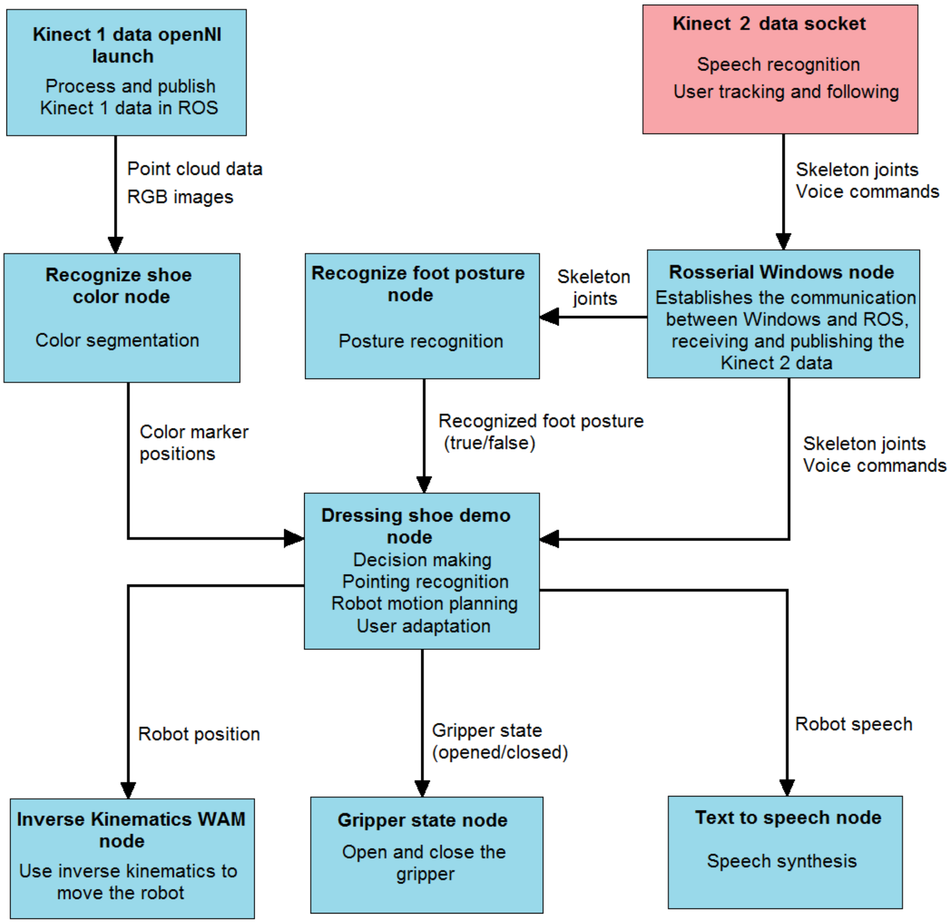

Kinect 1 data openNI launch
Process and publish Kinect 1 data in ROS
Kinect 2 data socket
Speech recognition
User tracking and following
Point cloud data
RGB images
Skeleton joints
Voice commands
Recognize shoe color node
Color segmentation
Recognize foot posture node
Posture recognition
Skeleton joints
Rosserial Windows node
Establishes the communication between Windows and ROS, receiving and publishing the Kinect 2 data
Color marker positions
Recognized foot posture (true/false)
Skeleton joints
Voice commands
Dressing shoe demo node
Decision making
Pointing recognition
Robot motion planning
User adaptation
Robot position
Gripper state (opened/closed)
Robot speech
Inverse Kinematics WAM node
Use inverse kinematics to move the robot
Gripper state node
Open and close the gripper
Text to speech node
Speech synthesis

Figure 3.21: Diagram about the integration of the algorithms in ROS. Red color indicates the implementation in Windows, blue in Ubuntu.

tracking (Section 3.2.3) algorithms were integrated using this socket, providing the user's joints and the semantic meaning tags from speech.

- **Rosserial Windows node**: From one of the Linux-based PC, the Rosserial Windows node published the data coming from the Kinect 2 data socket.
- **Kinect 1 data openNI launch**: The Kinect 1 data openNI launch provided the RGB images and point clouds coming from the Kinect 1.
- **Recognize shoe color node**: The Recognize shoe color node used the RGB images and point clouds from the Kinect 1 to execute the object-color-segmentation algorithm, described in Section 3.2.4. Whenever it was requested, the node executed this algorithm and published the positions of the recognized color markers in ROS.

- **Recognize foot posture node**: The user's joints are required for the Recognize foot posture node, which executes the posture recognition described in Section 3.13. A boolean is published in the ROS framework, meaning by true that the extended right foot is recognized, and by false the inverse.

- **Dressing shoe demo node**: This is the main node, which integrated the pointing-recognition, decision-module, position-adjustment and robot-motion-planning algorithms. It required the color markers positions, the recognized voice commands and the user's joints to execute the four integrated algorithms. As output, this node publishes the position of the WAM robot, the state of the gripper and feedback messages.

- **Inverse kinematics WAM node**: The inverse kinematics WAM node had integrated the inverse kinematics algorithm of the WAM robot. The published WAM robot position is converted to the robot's joints poses.

- **State gripper node**: The state gripper node was responsible for opening and closing the gripper.

- **Text to speech node**: The feedback of the robot was sent as a text string in order to be converted to speech by the speech-synthesis algorithm described in Section 3.2.2.

The real-time visualization of the dressing task in ROS was produced in Rviz[10], as shown in Fig. 3.22, with representations of the WAM robot, platform with shoes, and the user's joints. Everything integrated in the ROS framework was referred to the WAM robot reference system, described in Section 4.1

---

[10] Rviz: http://wiki.ros.org/rviz

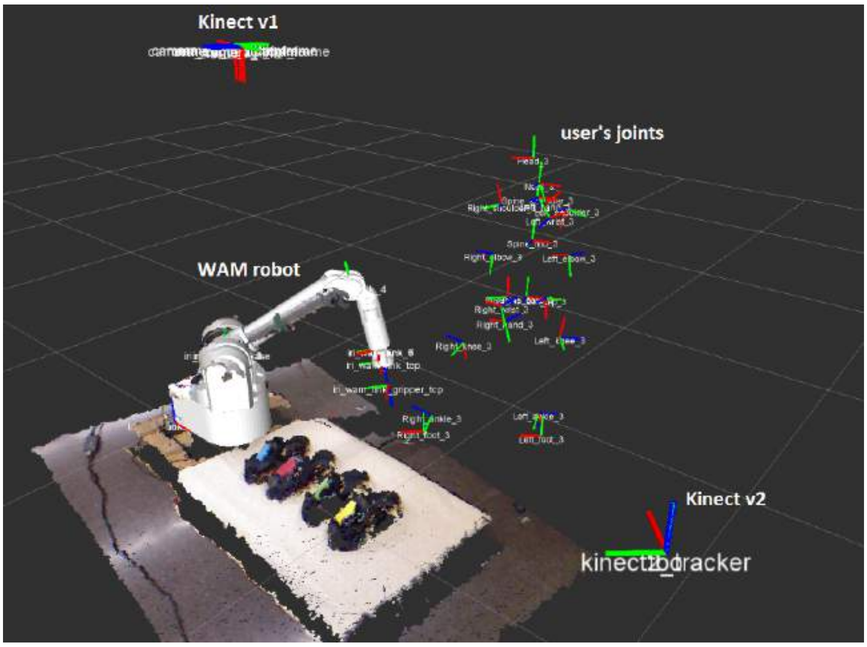


Figure 3.22: Visualization of the system in Rviz

Chapter 4

Experiments

The implemented autonomous robotic system for dressing assistance was tested in a serie of experiments with users with no previous experience in robotics. The experiments were designed to evaluate performance and user workload under different conditions. The following sections describe the experimental setup, tasks, user profiles and evaluation metrics.

4.1 Experimental setup

The proposed experimental setup consisted of one WAM robot, Kinect 1, Kinect 2, and a platform holding four different shoes, two rights (blue and green) and two lefts (red and yellow). An illustration of the scenario is shown in Fig. 4.1

The WAM robot performed the dressing task. All the following measurements are given in the WAM robot reference system. This origin was parallel to the floor situated at $0.45m$ upper, and in the center of the WAM robot's base. The Kinect 1 was placed on the top and it was used for the recognition of the shoes. Its location from the robot reference system was $(x, y, z) = (0.38m, -0.07m, 1.16m)$, and its orientation given by its Euler angles was $(\alpha, \beta, \gamma) = (-139^\circ, 80^\circ, -37^\circ)$. The Kinect 2 was used to recognize speech, and track the user movements. Its position was in front of the user was chosen to avoid the occlusion by the WAM robot when dressing the shoe. The position in the robot reference system was set to $(x, y, z) = (2.03m, -0.57m, 0.53m)$ and its orientation in the Euler angles was $(\alpha, \beta, \gamma) = (0^\circ, 0^\circ, 121^\circ)$.

The process of calibrating the cameras with respect to the robot's position was very important for the success of the dressing task. It was made manually, measuring the distances to each axis in the WAM robot reference system and with the visualization of the scenario in the ROS framework using rviz.

In front of the robot there was a small platform from where the robot

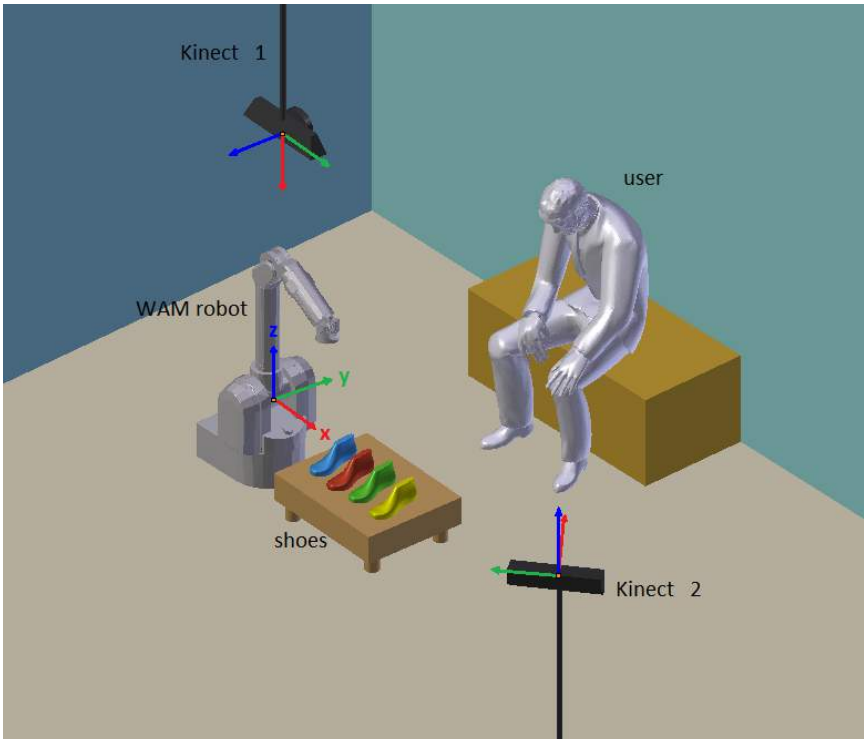


Figure 4.1: Dressing scenario

was able to pick up the shoes. The platform was parallel to the floor, placed at $-0.33m$ from the origin in the z direction. This platform is the so-called shoe plane in Section 3.2.5. The shoes in the scenario were crocs shoes with a attached ribbon in which different colored markers were placed: yellow, red, blue and green. The shoes were placed inside the WAM workspace, being equally distant one with another. The blue shoe was the closest to the WAM robot at a distance of $0.3m$, and the yellow shoe was the furthest at a distance of $0.85m$, both distances measured from the WAM robot reference system.

The user was seated on one side of the WAM robot and in front of the Kinect 2. The distance of the user from the robot was flexible however, two constraints were considered: the user was located inside the visual field of the Kinect 2 to allow the motion tracking, and the extended foot of the user had to be at a distance between $0.50m$ and $0.70m$ from the WAM robot base. In this way the extended foot was inside the workspace of the robot. The user's chair was a wheeled platform, allowing the user to adjust the distance from the robot.

The dressing task performed by the robot was performed in several steps:

- (i) The user selected a one of the available shoes, either by pointing the target shoe with the right arm saying *"take this shoe"* or using only speech, for instance *"take the green shoe"*.

- (ii) The user was able to correct the robot in order to take other shoe following the same procedure as described in (i).
- (iii) The dressing part started when the user said *"dress me"*.
- (iv) The robot waited until the user extended the right foot. Then the robot holding the shoe approached the user's foot at a distance of $0.50m$ in the z axis and $0.40m$ in the xy plane, taking into account the foot's orientation.
- (v) The robot followed the user's foot. Now the user could adjust the position of the WAM robot saying *"move up", "move down", "move forward"* and *"move back"*.
- (vi) When a comfortable position was found, the user said *"stop"* in order to stop the robot in the current position.
- (vii) The task was finished when the user put the foot inside the crocs shoe and said *"that's ok"*.

4.2 User tests

The robot was evaluated in experiments with real users. The autonomous robotic system was designed to dress only the right foot. In order to simulate the dressing process of putting on both shoes, each trial performed with participants consisted in dressing two right shoes such that the complexity of the task was similar to dressing both shoes.

Since two different modalities were implemented to select a shoe, speech and pointing, two experiments were defined, one for each modality.

- **Experiment 1**: The experiment 1 was designed to evaluate the use of speech modality when choosing a shoe. The participant started the trial by saying *"begin"*, and selected the blue shoe with the command *"take the blue shoe"*. If the robot recognized a wrong command, taking any other shoe, the user corrected the robot by repeating the same command until the blue shoe was picked up. Then the participant said *"dress me"*, extending the right foot towards the robot to be assisted. When the extended foot posture was recognized, the robot approached to the user's foot, following it until the command *"stop"* was recognized. Finally the robot waited for the participant to put the foot inside the shoe. Finally the user said *"that's ok"*. The robot leaved the crocs shoe and came back to its home position. The same task was repeated with the green shoe by saying *"take the green shoe"*.

- **Experiment 2**: This experiment was designed to evaluate the use of pointing modality when choosing a shoe. In the experiment 2, the participant started the trial saying *"begin"*, but now the selection of the shoe was made through the pointing gesture. The user had to point the blue shoe, and then say *"take this shoe"*. If the robot made a mistake picking up any other shoe, the user had to insist on taking the blue shoe using pointing again. Then, the participant had to say *"dress me"* and extend the foot. The robot approached and followed the foot until the user said *"stop"*. The dressing assistance was completed when the user put the right foot into the crocs shoe and said *"that's ok"*. The same task was repeated for the green shoe using pointing.

In addition, two groups of users were created to evaluate the user adaptation:

- **Group 1**: The group 1 was created to evaluate the autonomous robotic system without user adaptation. This group consisted of 6 people. The order of the experiments was changed in subgroups of 3 for counter-balancing, in order to reduce the learning effect.
- **Group 2**: The group 2 was created to evaluate the user adaptation, which included the pointing calibration and the robot position adjustment. The group consisted of 6 people, with the same procedure used as before: 3 participants performed the experiment 1 first followed by the experiment 2 and the other 3 in the inverse order.

  Before performing the experiments, each participant in this group had to calibrate the pointing recognition by saying *"fit pointing"*. The system requested the user to point the blue shoe and say *"take this shoe"*, and the same for the red, green and yellow shoes. The calibration finished after pointing the four shoes and was taken into account in the experiment 2.

  In addition, the robot position was adjusted. Each participant had to say *"begin"* and select the blue shoe with the command *"take the blue shoe"*. Then the user said *"dress me"* and extended the right foot for the assistance. In that moment the robot approached the foot to a prudent distant. The participant adjusted this distance using the voice commands *"move up"*, *"move down"*, *"move forward"* and *"move back"*. When one of these commands was recognized, the robot continuously moved in the selected direction until the user said *"stop"*, stopping the WAM robot in the current position. If this position was not still comfortable for the user, the last four commands were used again to re-adjust the distance. Once the position was comfortable,

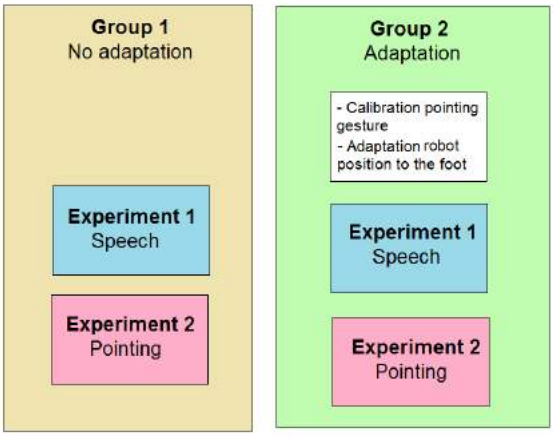


Figure 4.2: Representation of the distributed tasks and experiments for the two groups of participants. The group 1, with no adaptation, performed the experiment 1 and 2. The group 2, with adaptation, first calibrated the pointing recognition and adjusted the robot position. Then the experiment 1 and 2 were performed.

the participant finished the adjustment saying *"that's ok"*. This adjustment was stored and used both experiments for each participant.

The distributed tasks and experiments for each group of participants are summarized in Fig. 4.2

A total of 12 participants were requested to perform 5 trials of each experiment. They were equally distributed between the two groups, each group containing 2 female and 4 male adults of age between 22 and 29. All participants had a university degree (6 engineers, 3 computer scientist, 2 chemist and 1 biologist) and no previous experience in robotics.

The procedure of each experiment with the participants consisted of five steps:

- 1) The given experiment was explained to each user using written instructions. In this way all the participant received the same information and hence the same conditions.
- 2) I performed a demo showing how the assisted robot works.
- 3) The participant was invited to try the system for a few minutes before doing the first experiment. The distance between the user and the robot was adjusted such that the extended foot was within the workspace of the WAM robot.
- 4) The user performed 5 trials for the determined experiment, where each trial consisted on putting on the blue and the green shoe.
- 5) The participant filled in a NASA-TLX questionnaire for each experiment.

All the participants in this work were assured that the robot was in the compliant mode, making any contact with the robot safe during the experiments. I was present during the experiments to monitor the correct functionality of the robot, without interfering on providing any feedback to the participant.

4.3 Evaluation metrics

The following metrics were used to measure the performance of the autonomous robotic system and the workload of the participants. These metrics are divided into quantitative and qualitative:

- **Quantitative metrics**
  - **Completion Time:** The completion time was the total time used by the participant to be assisted in dressing two shoes. A timer started when the voice command *"begin"* was recognized and stopped with the command *"that's ok"*. Since there were two separated tasks per trial (one for the blue shoe and other for the green shoe), two different times were obtained. The sum of these two resulted in the total completion time, which was the used metric for the experiments. Lower competition times indicated better performance.
  - **Success:** The success indicated the number of successfully delivered shoes. A perfect performance was when both shoes were successfully put on the user's foot.
  - **Number of Corrections:** This was the number of times that the user had to correct the robot in order to pick the desired shoe. If the number of corrections is zero, the robot correctly picked up the chosen shoe in the first attempt. Since there were two dressing task per trial, the total number of corrections was the sum of the corrections from both tasks. This metric was used to evaluate the performance of the pointing algorithm.
  - **User angle θ^u and corrected angle θ^c:** These two angles were computed from the calibration performed by the group 2. The computation of these angles was described in Section 3.2.8 part A.
  - **Position of the robot:** The position of the robot adjusted by users in group 2 refers to the distance of the robot respect to the foot during the dressing task. This metric was also used to compute the average position, i.e. the most comfortable position chosen by the users.
- **Quantitative metrics**

- **Raw NASA-TLX Questionnaire:** Participants were asked to complete a computerized version of the questionnaire for each experiment. The raw NASA-TLX enables the collection of six dimensions of workload: mental demand, physical demand, temporal demand, performance, effort and frustration, all of them ranging from 0 to 100 [72]. This questionnaire was used to assess the participant workload when interacting with the robot for each experiment, similarly as in [73, 69, 13].

Chapter 5

Results and Discussion

In this chapter, the quantitative and qualitative analysis of the results from the experiments is presented. The qualitative analysis was concerned with the metrics that evaluated the accuracy of the system and the overall task performance. The qualitative analysis provided the insight into the user workload, which was evaluated using the NASA-TLX questionnaire.

5.1 Quantitative analysis

The results obtained with the quantitative metrics are described for each experiment and each group of users. The experiments were designed to evaluate the two different modalities when selecting a shoe: speech and pointing. The two groups of participants were used to evaluate the user adaptation algorithms. The group 1 performed without any adaptation and the group 2 performed with adaptation.

The total success rate did not show any difference between the user groups. For a total of 120 trials performed by 12 participants, the 97.7% were successfully accomplished. Cases of failure appeared when the shoe fell down to the floor after the user said the command *that's ok*, without placing the foot into the shoe properly.

The average and standard deviation of the completion time is shown in Fig. 5.1. The completion time was approximately the same using the speech modality for both groups. However, in the pointing modality, this time was in average 23.3% longer in group 1 than in group 2. In fact, for the group 2 the completion time was similar using both modalities. Users in group 1 performed the task 24.2% slower with the pointing respect to speech, indicating that the pointing was less precise without calibration.

The average number of corrections and its standard deviation for each modality and group is represented in Fig. 5.2. The corrections using speech modality was again similar in both groups. This behaviour was expected since both groups had the same conditions using speech. On the other

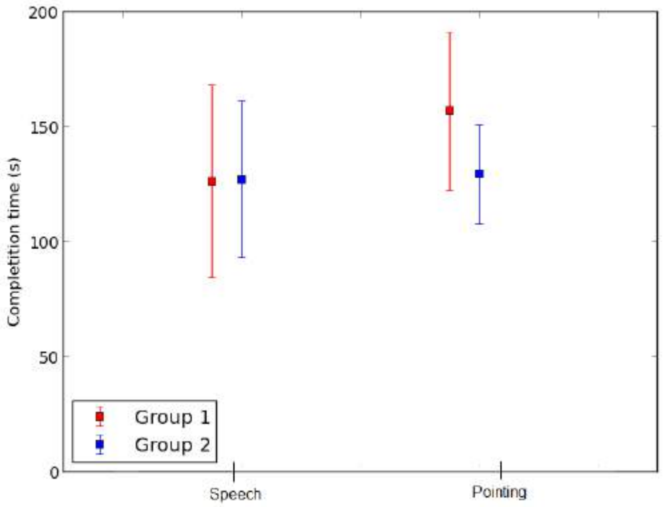


Figure 5.1: Effect of the interaction modality and adaptation of the autonomous system on the task completion time. Error bars represent the standard deviation of the mean.

hand, the corrections performed by group 1 with respect to the group 2 using pointing was on average higher. The standard deviation is also high in group 2, which indicated that the way of pointing depended of each participant: some pointed better than others. The pointing calibration allowed reduce the number of corrections by 79.2% compared to the performance of the group 1. In fact, the group 2 had similar number of corrections in both modalities, meaning that with the calibration, the pointing modality was as accurate as speech.

Both the corrected and user angles were measured for the group 2. The average and the standard deviation of both angles is shown in Fig. 5.3. Using the convention described in Fig. 3.17, both angles were negative from the WAM robot reference system. In general, participants pointed with a deviation to the left from the user point of view. No clear conclusions can be extracted about this behaviour, but it could be caused because all users pointed with the right arm.

The difference between the corrected and the user angles ($\theta^c - \theta^u$) with mean and standard deviation values is shown in Fig. 3.2.5. The assumption for the pointing-calibration algorithm in Section 3.2.5 was to consider a linear relationship between these two angles. A linear trend can be noted for the difference of these two angles, although the standard deviations are too high for only 6 participants.

Finally, the mean value of the initial and final position of the robot with respect to the ankle is shown in Fig. 5.5. This measurement provides the point from where the users encounter more comfortable the dressing

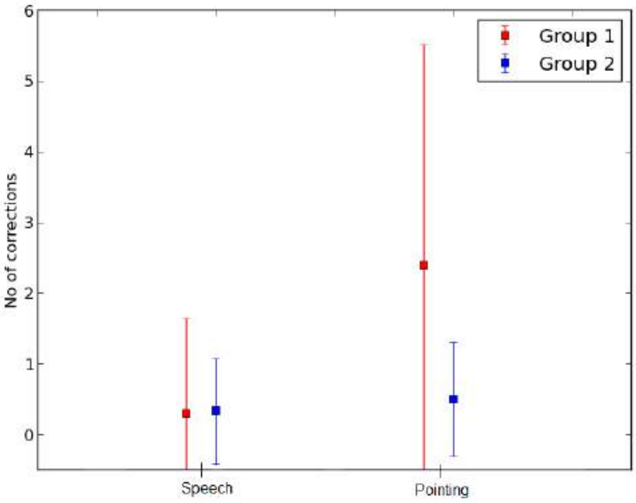


Figure 5.2: Effect of the interaction modality and adaptation of the autonomous system on the number of corrections. Error bars represent the standard deviation of the mean

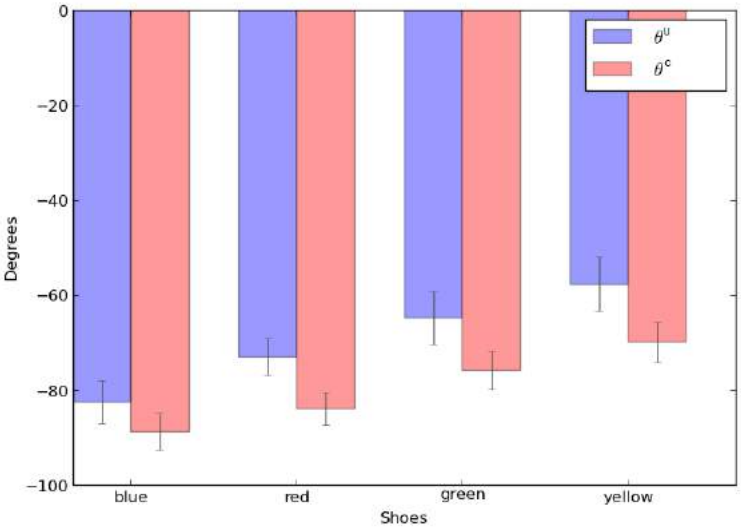


Figure 5.3: Mean of both corrected (θ^c) and user angles (θ^u) for each shoe. Error bars represent the standard deviation of the mean

assistance in average. able the dressing assistance.

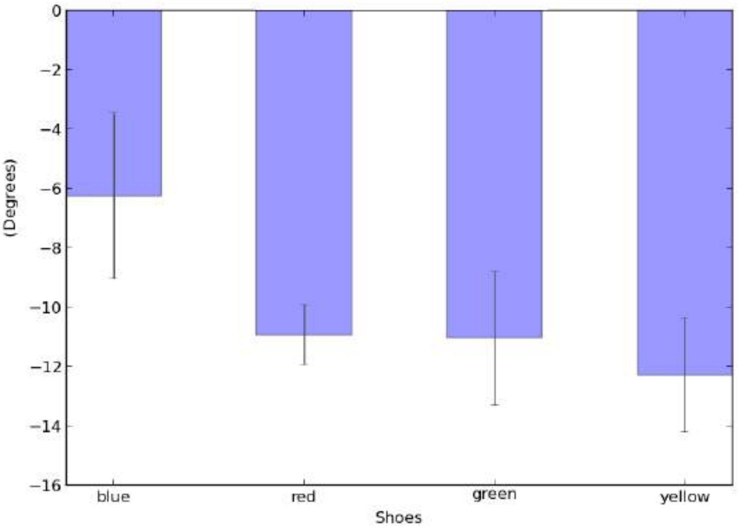


Figure 5.4: Difference between the corrected and the user angle ($\theta^c - \theta^u$) for each shoe. Error bars represent the standard deviation of the mean

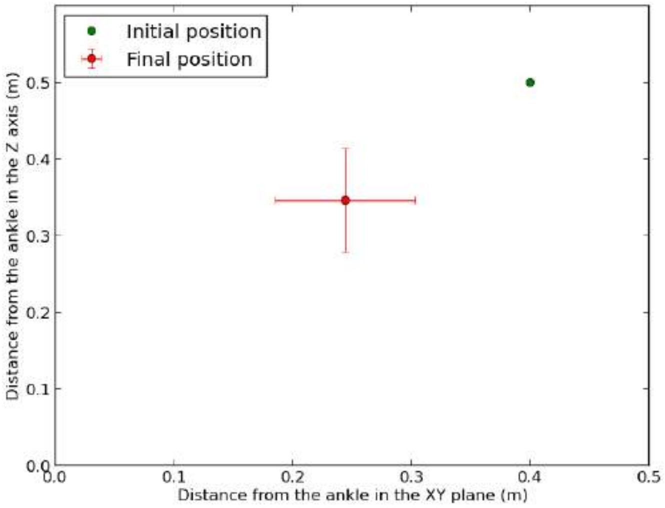


Figure 5.5: Initial and final position of the gripper respect to the ankle. Error bars represent the standard deviation of the mean

5.2 Qualitative analysis

The NASA-TLS questionnaires were used for a qualitative analysis of the experiments. The averages and standard deviations of the six dimensions of workload for each group and each modality are shown in Fig. 5.6.

On average, the mental demand was higher in group 2 than in group 1 using both modalities. This might be caused by the calibration and adjustment procedure performed by participants in group 2. Indeed, these users performed a more complex test with the robot taking into account the adaptation tasks.

The physical demand was approximately the same in both groups using speech. Although a lower physical demand was expected for group 2, which adjusted the robot position, the physical effort was rated very low in both groups. Therefore the difference between the two groups in the physical demand might not be appreciated using in this scale. Regarding the pointing modality, the physical demand in group 2 remained at the same level as for speech, but it increased 5.0% in group 1. Since the number of corrections in group 2 was higher, it is expected that a larger number of corrections implied more physical effort.

The temporal demand was evaluated lower in group 2 for both modalities. The user adaptation made participants in group 2 to evaluate the temporal demand 5.0% lower using speech and 8.3% lower using pointing than group 1. This result is compatible with the completion time measurement.

The performance was also higher for the participants in group 1, meaning that they performed worse the two experiments than the group 2. The performance using speech decreased 19.1% in group 2 while in pointing decreased 8.3%

Regarding the total effort, the results were very similar for both groups, but different between modalities. The effort was higher using the pointing rather than speech in both groups. Group 1 experienced 10.8% higher effort using pointing than speech. Indeed pointing was a combination of speech and pointing gesture to form the so-called deictic expressions, therefore the final effort of the pointing modality is presumed to be higher as the results shows.

Finally, the frustration was similar for both groups using the speech. From this result it is extracted that the adjustment of the robot position to the foot was not a source of frustration. About the pointing modality, group 1 experienced 10.0% more frustration than group 2. This result is has the same behaviour as the number of corrections, being both results compatible. The participants in group 1 experienced more frustration when the pointing calibration was not performed.

The overall workload was the average of the 6 dimensions, and it is represented in Fig. 5.7. Group 2 experienced less overall workload in both modalities: 3.2% for speech and 5.4% for pointing. On the other hand, group 1 rated 5.8% higher the workload using pointing rather than speech. Group 1 only rated 3.6% higher the pointing modality.

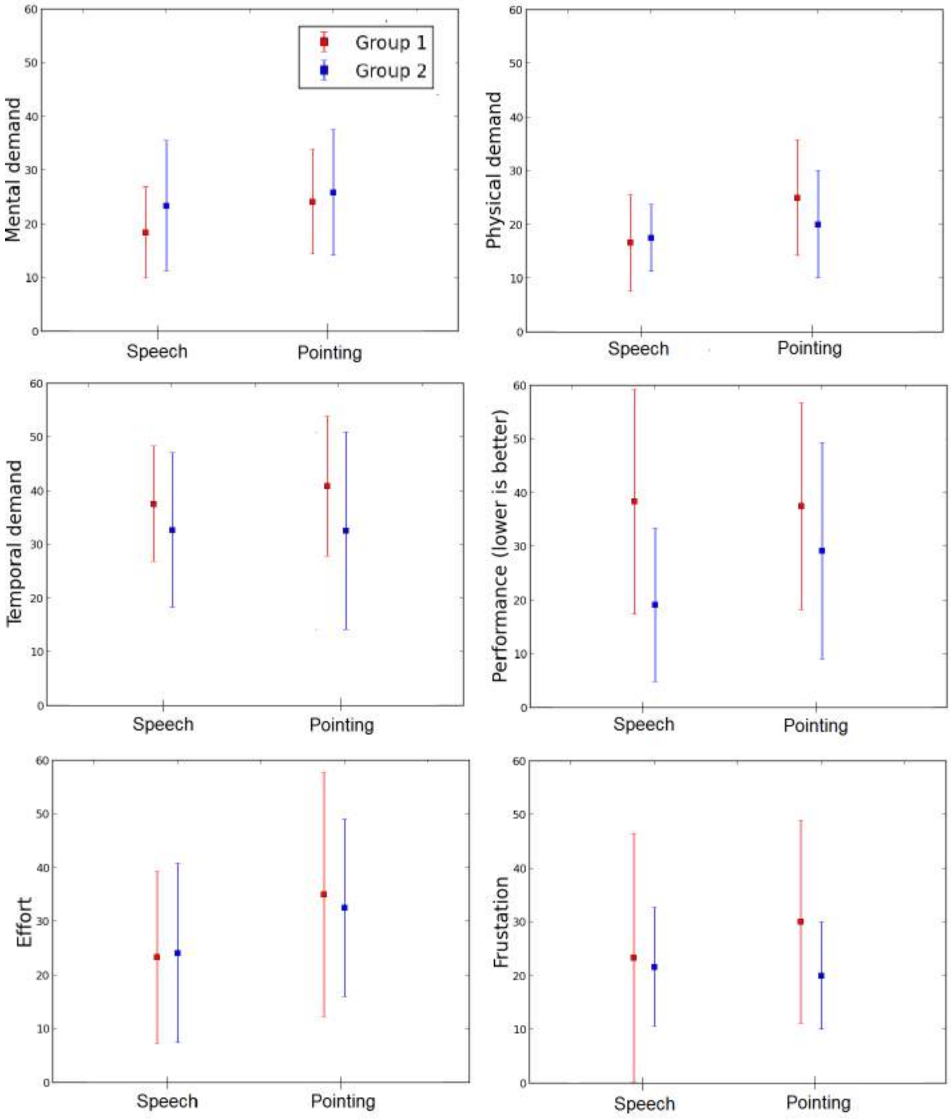


Figure 5.6: Effect of the interaction modality and adaptation of the autonomous system on the six dimensions of workload. Error bars represent the standard deviation of the mean

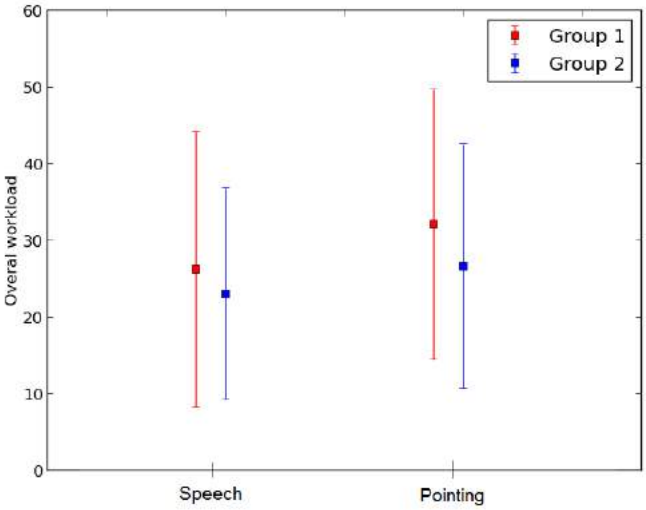


Figure 5.7: Effect of the interaction modality and adaptation of the autonomous system on overall workload. Error bars represent the standard deviation of the mean

Chapter 6

Conclusions

Development of a robot assistant with activities of daily living is expected to prolong independent living and improve life quality of patients with reduced mobility. The work presented in this dissertation is an attempt to move a step toward this ambitious goal. Contributions were made on several levels of robotic system development including algorithm development, system integration and evaluation with users. The final system was able to assist the users with a dressing task using speech and motion recognition, which was the main goal of this work.

The main result of the project is a multi-modal human-robot interaction framework that combines modalities such as voice command, gesture and body posture, which are based on speech and motion recognition. The potential of the proposed framework is not limited to development of speech and motion recognition interfaces; its strongest feature is that it can combine both speech and motion to produce so-called deictic expressions. For example, in case of ambiguity, where there are several similar shoes to choose from, the user may select a shoe by saying *"take that shoe"* while pointing in the direction of the desired shoe. The experimental results showed that this modality is less precise and requires more effort from the users than using the voice commands only (task was performed 24.2% slower requiring 10.8% more effort). However, it is more powerful when spoken words are not descriptive enough.

The seated position of the user in the proposed shoe-dressing scenario poses additional challenge for user tracking. Development of algorithms for user foot tracking and robot motion planning, as well as pointing recognition, and posture recognition, required exhaustive system testing to achieve a robust task performance. The system integration required a development of a finite state machine that will be able to handle all steps of the dressing task and unexpected events such as withdrawal of the foot or illogical requests from the user, such as asking for a non-existent shoe.

The adaptation will be of utmost importance for user acceptance of assistive robots in the future. For the robot assistant in dressing, the ability to adapt to user preferences was achieved through development of algorithms for pointing calibration and robot position adjustment. Experimental results show that after applying the pointing calibration the number of pointing corrections was reduced by 79.2%. Also, the users that performed the adaptation accomplished the dressing task 23.3% faster and experienced less physical demand (5.0%) and frustration (10%) than the users who did not perform the adaptation. Although the analysis of results included various metrics for quantitative and qualitative system evaluation, one of the future tasks will be the to evaluate their statistical significance, for example a two-way repeated measure ANOVA test.

All the developments and evaluation were performed in the laboratory setting. Future work may include the extension of the current task to dressing of both feet. As a part of future work, the proposed HRI framework could be adapted for other applications by redefining the task segments as well as modifying the application-specific vocabularies for speech, gestures and poses. The implementation of the developed algorithms on a commercial robot platform is another long-term objective. The robot assistants in the future will be required to help their users with a variety of tasks, as well as serve as social companions. The ability to assist with dressing will represent a useful addition to the set of robot skills.

References

[1] J.E. Cohen. “Human population: the next half century”. In: *science* 302.5648 (2003), pp. 1172–1175.

[2] G.A. Warshaw, J.T. Moore, S.W. Friedman, C.T. Currie, D.C. Kennie, W.J. Kane, and P.A. Mears. “Functional disability in the hospitalized elderly”. In: *Jama* 248.7 (1982), pp. 847–850.

[3] J.C. Furlan, B.M. Sakakibara, W.C. Miller, and A.V. Krassioukov. “Global incidence and prevalence of traumatic spinal cord injury”. In: *The Canadian Journal of Neurological Sciences* 40.04 (2013), pp. 456–464.

[4] A.S. Kim and S.C. Johnston. “Global variation in the relative burden of stroke and ischemic heart disease”. In: *Circulation* 124.3 (2011), pp. 314–323.

[5] M.E. Pollack. “Intelligent technology for an aging population: The use of AI to assist elders with cognitive impairment”. In: *AI magazine* 26.2 (2005), p. 9.

[6] J.J. Cabibihan, H. Javed, M. Ang Jr, and S. M. Aljunied. “Why robots? A survey on the roles and benefits of social robots in the therapy of children with autism”. In: *International journal of social robotics* 5.4 (2013), pp. 593–618.

[7] H. Robinson, B. MacDonald, and E. Broadbent. “The role of healthcare robots for older people at home: A review”. In: *International Journal of Social Robotics* 6.4 (2014), pp. 575–591.

[8] P. Flandorfer. “Population ageing and socially assistive robots for elderly persons: the importance of sociodemographic factors for user acceptance”. In: *International Journal of Population Research* 2012 (2012).

[9] J. Broekens, M. Heerink, and H. Rosendal. “Assistive social robots in elderly care: a review”. In: *Gerontechnology* 8.2 (2009), pp. 94–103.

[10] J. Fasola and M.J. Mataric. “Using socially assistive human–robot interaction to motivate physical exercise for older adults”. In: *Proceedings of the IEEE* 100.8 (2012), pp. 2512–2526.

[11] C.A. Smarr, T.L. Mitzner, J.M. Beer, A. Prakash, T.L. Chen, C.C. Kemp, and W.A. Rogers. “Domestic robots for older adults: attitudes, preferences, and potential”. In: *International journal of social robotics* 6.2 (2014), pp. 229–247.

[12] S. Satake, T. Kanda, D.F. Glas, M. Imai, H. Ishiguro, and N. Hagita. “How to approach humans?-strategies for social robots to initiate interaction”. In: *Human-Robot Interaction (HRI), 2009 4th ACM/IEEE International Conference on.* IEEE. 2009, pp. 109–116.

[13] A. Jevtić, A. Colomé, G. Alenyà, and C. Torras. “User Evaluation of an Interactive Learning Framework for Single-Arm and Dual-Arm Robots”. In: *International Conference on Social Robotics.* Springer. 2016, pp. 52–61.

[14] International Federation of Robotics IFR. *Definition of Service Robots.* URL: `http://www.ifr.org/service-robots/` (visited on 01/20/2017).

[15] Silicon Valley. *Service Robotics Case Studies.* URL: `https://svrobo.org/wp-content/uploads/2015/05/Service-Robotics-Case-Studies.pdf` (visited on 01/20/2017).

[16] C. Torras. “Service robots for citizens of the future”. In: *European Review* 24.01 (2016), pp. 17–30.

[17] D. Feil-Seifer and M.J. Mataric. “Defining socially assistive robotics”. In: *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.* IEEE. 2005, pp. 465–468.

[18] A. Tapus, M.J. Mataric, and B. Scassellati. “The grand challenges in socially assistive robotics”. In: *Robotics and Automation Magazine* 14.1 (2007), pp. 1–7.

[19] K. Yamazaki, R. Ueda, S. Nozawa, Y. Mori, T. Maki, N. Hatao, K. Okada, and M. Inaba. “System integration of a daily assistive robot and its application to tidying and cleaning rooms”. In: *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on.* IEEE. 2010, pp. 1365–1371.

[20] F. Gravot, A. Haneda, K. Okada, and M. Inaba. “Cooking for humanoid robot, a task that needs symbolic and geometric reasonings”. In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE. 2006, pp. 462–467.

[21] T. Tamei, T. Matsubara, A. Rai, and T. Shibata. “Reinforcement learning of clothing assistance with a dual-arm robot”. In: *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on.* IEEE. 2011, pp. 733–738.

[22] N. Koganti, T. Tamei, T. Matsubara, and T. Shibata. "Estimation of human cloth topological relationship using depth sensor for robotic clothing assistance". In: *Proceedings of Conference on Advances In Robotics*. ACM. 2013, pp. 1–6.

[23] T. Matsubara, D. Shinohara, and M. Kidode. "Reinforcement learning of a motor skill for wearing a T-shirt using topology coordinates". In: *Advanced Robotics* 27.7 (2013), pp. 513–524.

[24] N. Koganti, T. Tamei, T. Matsubara, and T. Shibata. "Real-time estimation of Human-Cloth topological relationship using depth sensor for robotic clothing assistance". In: *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*. IEEE. 2014, pp. 124–129.

[25] A. Colomé, A. Planells, and C. Torras. "A friction-model-based framework for reinforcement learning of robotic tasks in non-rigid environments". In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE. 2015, pp. 5649–5654.

[26] Y. Gao, H.J. Chang, and Y. Demiris. "User modelling for personalised dressing assistance by humanoid robots". In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE. 2015, pp. 1840–1845.

[27] J.P. Costeira, F.S. Melo, and M. Veloso. "Personalized Assistance for Dressing Users". In: *Social Robotics: 7th International Conference, ICSR 2015, Paris, France, October 26-30, 2015, Proceedings*. Vol. 9388. Springer. 2015, p. 359.

[28] K. Yamazaki, R. Oya, K. Nagahama, K. Okada, and M. Inaba. "Bottom Dressing by a Dual-arm Robot Using a Clothing State Estimation Based on Dynamic Shape Changes". In: (2016).

[29] K. Yamazaki, R. Oya, K. Nagahama, and M. Inaba. "A method of state recognition of dressing clothes based on dynamic state matching". In: *System Integration (SII), 2013 IEEE/SICE International Symposium on*. IEEE. 2013, pp. 406–411.

[30] G. Chance, A. Camilleri, B. Winstone, P. Caleb-Solly, and S. Dogramadzi. "An assistive robot to support dressing–Strategies for planning and error handling". In: *Proceedings of the 6th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics*. IEEE. 2016.

[31] J.A. Markowitz. *Using speech recognition*. Markowitz, J. Consultants, 2000.

[32] C. Theobalt, J. Bos, T. Chapman, A. Espinosa-Romero, M. Fraser, G. Hayes, E. Klein, T. Oka, and R. Reeve. "Talking to Godot: Dialogue with a mobile robot". In: *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on.* Vol. 2. IEEE. 2002, pp. 1338–1343.

[33] G. Bugmann. "Effective Spoken interfaces to service robots: Open problems." In: *Companions: Hard Problems and Open Challenges in Robot-Human Interaction* (2005), p. 18.

[34] P. Nugues, M. Haage, and S. Schötz. "A prototype robot speech interface with multimodal feedback". In: *Proceedings of the 2002 IEEE-Int. Workshop Robot and Human Interactive Communication.* 2005, pp. 247–252.

[35] D. Estival. "Adding language capabilities to a small robot". In: *Department of Linguistics and Applied Linguistics, University of Melbourne* (1998).

[36] H. Huttenrauch, A. Green, M. Norman, L. Oestreicher, and K.S. Eklundh. "Involving users in the design of a mobile office robot". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 34.2 (2004), pp. 113–124.

[37] J. Norberto Pires. "Robot-by-voice: Experiments on commanding an industrial robot using the human voice". In: *Industrial Robot: An International Journal* 32.6 (2005), pp. 505–511.

[38] S. Durling and J. Lumsden. "Speech recognition use in healthcare applications". In: *Proceedings of the 6th international conference on advances in mobile computing and multimedia.* ACM. 2008, pp. 473–478.

[39] R. Parente, N. Kock, and J. Sonsini. "Analysis of the Implementation and Impact of Speech-Recognition Technology in the Healthcare Sector". In: *Analysis of the Implementation and Impact of Speech-Recognition Technology in the Healthcare Sector/AHIMA, American Health Information Management Association* (2004).

[40] D.I. Rosenthal, F.S. Chew, D.E. Dupuy, S.V. Kattapuram, W.E. Palmer, R.M. Yap, and L.A. Levine. "Computer-based speech recognition as a replacement for medical transcription." In: *AJR. American journal of roentgenology* 170.1 (1998), pp. 23–25.

[41] K.H. Cohen. "Patient monitoring system including speech recognition capability". In: US Patent 6,014,626. Google Patents, 2000.

[42] L. Chen, H. Wei, and J. Ferryman. "A survey of human motion analysis using depth imagery". In: *Pattern Recognition Letters* 34.15 (2013), pp. 1995–2006.

[43] J. Gall, C. Stoll, E. De Aguiar, C. Theobalt, B. Rosenhahn, and H. Seidel. “Motion capture using joint skeleton tracking and surface estimation”. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on.* IEEE. 2009, pp. 1746–1753.

[44] L. Arthur Schwarz, A. Mkhitaryan, D. Mateus, and N. Navab. “Human skeleton tracking from depth data using geodesic distances and optical flow”. In: *Image and Vision Computing* 30.3 (2012), pp. 217–226.

[45] E. Murphy-Chutorian and M.M. Trivedi. “Head pose estimation in computer vision: A survey”. In: *IEEE transactions on pattern analysis and machine intelligence* 31.4 (2009), pp. 607–626.

[46] J. Shotton, R. Girshick, A. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, R. Moore, P. Kohli, A. Criminisi, and A. Kipman. “Efficient human pose estimation from single depth images”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.12 (2013), pp. 2821–2840.

[47] S. Mitra and T. Acharya. “Gesture recognition: A survey”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37.3 (2007), pp. 311–324.

[48] T.L. Le and M.Q. Nguyen. “Human posture recognition using human skeleton provided by Kinect”. In: *Computing, Management and Telecommunications (ComManTel), 2013 International Conference on.* IEEE. 2013, pp. 340–345.

[49] Z. Ren, J. Meng, J. Yuan, and Z. Zhang. “Robust hand gesture recognition with kinect sensor”. In: *Proceedings of the 19th ACM international conference on Multimedia.* ACM. 2011, pp. 759–760.

[50] K.K. Biswas and S.K. Basu. “Gesture recognition using microsoft kinect®”. In: *Automation, Robotics and Applications (ICARA), 2011 5th International Conference on.* IEEE. 2011, pp. 100–103.

[51] M. Svenstrup, S. Tranberg, H.J. Andersen, and T. Bak. “Pose estimation and adaptive robot behaviour for human-robot interaction”. In: *Robotics and Automation, 2009. ICRA’09. IEEE International Conference on.* IEEE. 2009, pp. 3571–3576.

[52] E. Seemann, K. Nickel, and R. Stiefelhagen. “Head pose estimation using stereo vision for human-robot interaction”. In: *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on.* IEEE. 2004, pp. 626–631.

[53] B. Jansen, F. Temmermans, and R. Deklerck. “3D human pose recognition for home monitoring of elderly”. In: *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society.* IEEE. 2007, pp. 4049–4051.

[54] S. Obdržálek, G. Kurillo, F. Ofli, R. Bajcsy, E. Seto, H. Jimison, and M. Pavel. “Accuracy and robustness of Kinect pose estimation in the context of coaching of elderly population”. In: *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE. 2012, pp. 1188–1193.

[55] S. Oviatt. “Advances in robust multimodal interface design”. In: *IEEE Computer Graphics and Applications* 23.5 (2003), pp. 62–68.

[56] S. Iba, C.J. Paredis, and P.K. Khosla. “Interactive multi-modal robot programming”. In: *Experimental Robotics IX*. Springer, 2006, pp. 503–513.

[57] J. Coutaz, L. Nigay, D. Salber, A. Blandford, J. May, and R.M. Young. “Four easy pieces for assessing the usability of multimodal interaction: the CARE properties”. In: *Human—Computer Interaction*. Springer, 1995, pp. 115–120.

[58] R. Stiefelhagen, C. Fugen, R. Gieselmann, H. Holzapfel, K. Nickel, and A. Waibel. “Natural human-robot interaction using speech, head pose and gestures”. In: *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*. Vol. 3. IEEE. 2004, pp. 2422–2427.

[59] R. Bischoff and V. Graefe. “Integrating vision, touch and natural language in the control of a situation-oriented behavior-based humanoid robot”. In: *Systems, Man, and Cybernetics, 1999. IEEE SMC’99 Conference Proceedings. 1999 IEEE International Conference on*. Vol. 2. IEEE. 1999, pp. 999–1004.

[60] R. Meena, K. Jokinen, and G. Wilcock. “Integration of gestures and speech in human-robot interaction”. In: *Cognitive Infocommunications (CogInfoCom), 2012 IEEE 3rd International Conference on*. IEEE. 2012, pp. 673–678.

[61] S. Iba, C.J. Paredis, and P.K. Khosla. “Intention aware interactive multi-modal robot programming”. In: *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*. Vol. 4. IEEE. 2003, pp. 3479–3484.

[62] S. Lavallee. “A new system for computer assisted neurosurgery”. In: *Engineering in Medicine and Biology Society, 1989. Images of the Twenty-First Century., Proceedings of the Annual International Conference of the IEEE Engineering in*. IEEE. 1989, pp. 926–927.

[63] P. Mayer, C. Beck, and P. Panek. “Examples of multimodal user interfaces for socially assistive robots in Ambient Assisted Living environments”. In: *Cognitive Infocommunications (CogInfoCom), 2012 IEEE 3rd International Conference on*. IEEE. 2012, pp. 401–406.

[64] R. Khosla, M.T. Chu, R. Kachouie, K. Yamada, F. Yoshihiro, and T. Yamaguchi. “Interactive multimodal social robot for improving quality of care of elderly in Australian nursing homes”. In: *Proceedings of the 20th ACM international conference on Multimedia.* ACM. 2012, pp. 1173–1176.

[65] Microsoft. *How Speech Recognition Works (Microsoft.Speech).* URL: `https://msdn.microsoft.com/en-us/library/hh378337(v=office.14).aspx/#Match%20Input%20to%20Speech%20Models` (visited on 01/20/2017).

[66] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. “Real-time human pose recognition in parts from single depth images”. In: *Communications of the ACM* 56.1 (2013), pp. 116–124.

[67] Microsoft. *JointType Enumeration (Kinect for Windows SDK 2.0).* URL: `https://msdn.microsoft.com/en-us/library/microsoft.kinect.jointtype.aspx` (visited on 01/20/2017).

[68] D. Droeschel, J. Stückler, and S. Behnke. “Learning to interpret pointing gestures with a time-of-flight camera”. In: *Proceedings of the 6th international conference on Human-robot interaction.* ACM. 2011, pp. 481–488.

[69] A. Jevtić, G. Doisy, Y. Parmet, and Y. Edan. “Comparison of interaction modalities for mobile indoor robot guidance: direct physical interaction, person following, and pointing control”. In: *IEEE Transactions on Human-Machine Systems* 45.6 (2015), pp. 653–663.

[70] A. Jevtić, G. Doisy, S. Bodiroža, Y. Edan, and V.V. Hafner. “Human-robot interaction through 3D vision and force control”. In: *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction.* ACM. 2014, pp. 102–102.

[71] G.A. ten Holt, M.J. Reinders, and E.A. Hendriks. “Multi-dimensional dynamic time warping for gesture recognition”. In: *Thirteenth annual conference of the Advanced School for Computing and Imaging.* Vol. 300. 2007.

[72] S.G. Hart and L.E. Staveland. “Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research”. In: *Advances in psychology* 52 (1988), pp. 139–183.

[73] C.W. Nielsen, M.A. Goodrich, and R.W. Ricks. “Ecological interfaces for improving mobile robot teleoperation”. In: *IEEE Transactions on Robotics* 23.5 (2007), pp. 927–941.

Appendices

Appendix A

ROS framework and code

In this appendix, a detailed explanation about the ROS nodes with the implemented algorithms is provided. Following sections assumed that the reader has knowledge about the main concepts used in ROS, such as topic, message type, publisher, subscriber, service, etc. Detailed information about these concepts can be found in ROS tutorials[1].

A.1 ROS nodes

The description about each ROS node used in this work is provided in each of the following sections. The source code is available in a git repository, also provided.

A.1.1 Kinect 2 data socket

The Kinect 2 data socket was used in order to send data from the Windows machine to the ROS framework. It was developed under Windows 8.1 professional with Visual Studio 2013 professional in C++. This socket requires to connect the Kinect 2 to the Windows machine using a usb 3.0 connection. The socket consists in two different programs running simultaneously: the speech-recognition and the user body tracking programs.

For these programs to run properly, the local IP address of the Ubuntu machine where *roscore* is placed has to be specified. More detailed information about the communication between Windows and Ubuntu using ROS can be found in the rosserial_windows package.

Speech-recognition program

The speech-recognition program uses the speech-recognition algorithm, described in Section 3.2.1, to send the voice commands through ROS. It in-

---

[1] ROS tutorials: http://wiki.ros.org/ROS/Tutorials

tegrates the Microsoft Speech Platform SDK 11, the Kinect for Windows SDK 2.0 and ROS libraries for Windows generated by the rosserial_windows package. The audio stream comes from the microphone array of the Kinect 2. The output of this program is:

- A publisher of the topic *"/commands"* with the message type *iri_dressing_shoe_demo/speech_commands*. This is a custom message developed in the *dressing_shoe_demo* node, which is defined below. The speech-recognition algorithm is listening all the time until a voice command is recognized; then its semantic tag (defined in Table 3.2) is sent with this message type through ROS using the *rosserial_windows* node.

The *iri_dressing_shoe_demo/speech_commands* is compiled together with the *dressing_shoe_demo* node, and it is generated by the *rosserial_windows* package for Windows. It has the following fields:

- **string command**: It contains the semantic tag of the recognized voice command.
- **float32 angle**: It indicates the source angle of the sound measured in the xy plane of the Kinect 2 reference system, but this information is not used in this work.
- **float32 confidence**: This is the confidence of the recognized command. It is ranged within the interval $[0, 1]$, although it must be higher than the threshold 0.3 defined for the recognition. The confidence value is not used in this work once the recognition is performed.

The source code of this program can be found in the following repository.

https://gitlab.iri.upc.edu/anflores/speechRecognition.git

User body tracking program

The user-body-tracking program contains the user-tracking-and-following algorithm described in Section 3.2.3. It integrates the Kinect for Windows SDK 2.0 and ROS libraries for Windows generated by the rosserial_windows package. The user tracking and following algorithm uses the depth information provided by the Kinect 2 to recognize the user's joints in real time. The output of this program is:

- The transforms of the joints, with their location and orientation, from the Kinect 2 reference system (shown in Fig. 3.7). Transformations are sent using the *tf* package, see the following documentation for further information:

  wiki.ros.org/tf

The source code of this program can be found in the following repository:

https://gitlab.iri.upc.edu/anflores/body_joints_ros.git

A.1.2 Rosserial_windows package

This package not only provides the *rosserial_windows* node for the communication with the Windows machine, but also the tools to generate the ROS libraries for Windows. These libraries were generated from Ubuntu and imported into the Kinect 2 data socket.

The *rosserial_windows* node is executed in the Ubuntu machine to receive all the data from the socket and publish it through ROS. Detailed information can be found wikipage of the package.

http://wiki.ros.org/rosserial_windows

A.1.3 Recognize foot posture node

This node was developed under Ubuntu in C++. It integrates the posture recognition algorithm described in Section 3.13, which uses the user's joints provided by the *rosserial_windows* node. The output of this node is the following:

- A publisher in the topic *"/dressing_poses"* with the message type *iri_recognize_foot_posture/posture*. This is a custom message in ROS that was developed in order to indicate if a posture is recognized in real time.

The message *iri_recognize_foot_posture/posture* has the following fields defined:

- **string posName**: This is the name of the recognized posture. In this project the extended foot is the only posture recognized, therefore the only name defined is *"extended_right_foot"*.
- **bool recognized**: The boolean indicates if the posture is recognized (true) or not (false).
- **int32 userID**: This number is the index of the user provided by the user motion and tracking algorithm (Section 3.2.3). Since the algorithm is able to recognize 6 different skeleton, the *userID* ranges from 0 to 5.

The source code of this node can be found in the following repository:

https://gitlab.iri.upc.edu/anflores/iri_recognize_foot_posture.git

A.1.4 Kinect 1 data openNI launch

This package provides all the data obtained from the Kinect 1 and publishes it in the ROS framework. More specifically, it launches files to open an OpenNI device (Kinect 1) and load all nodelets to convert raw depth/RGB/IR streams to depth images, disparity images, and (registered) point clouds.

The relevant outputs of this package for this work are:

- A publisher in the topic *"/iri_wam/camera/depth_registered/points"* with the message type *sensor_msgs/PointCloud2*. The point clouds are published through this topic. Detailed information about the message type can be found in the following link:

  http://docs.ros.org/api/sensor_msgs/html/msg/PointCloud2.html

- A publisher in the topic *"/iri_wam/camera/rgb/image_color"* with the message type *sensor_msgs/Image*, for the RGB images. Detailed documentation about this message type can be found in:

  http://docs.ros.org/api/sensor_msgs/html/msg/Image.html

Information about this package can be found in its wikipage:

http://wiki.ros.org/openni_launch

A.1.5 Recognize shoe color node

The *recognize shoe color* node integrates the object color segmentation algorithm in the ROS framework, described in Section 3.2.4.

The inputs of this node are:

- A subscriber to the topic *"/iri_wam/camera/depth_registered/points"* with the message *sensor_msgs/PointCloud2*.
- A subscriber to the topic *"/iri_wam/camera/rgb/image_color"* with the message type *sensor_msgs/Image*

This information is required by recognize the shoes and locate them in the space. On the other hand, the output of this node is the following:

- A service defined in the topic *"/interest_points"* with a message defined as a structure in C++. This structure contains the following fields:

  - **int id_color**: this indicates the index of the recognized color. For the defined colors in Table 3.4, which are red, green, blue and yellow, the indexes are 0, 1, 3 and 7 respectively.
  - **int U, V**: these two numbers represent the position in pixels (width, height) of the recognized color in the RGB image.
  - **float X, Y, Z**: this represents the position in the space of the recognized color from the Kinect 1 reference system, shown in Fig. 3.11.

When a client request this node to recognize the available shoes, the service send an array made of elements defined by the previous structure. Each element in the array represent a different recognized color. The source code of the *recognize shoe color* node is provided in the following repository:

https://gitlab.iri.upc.edu/anflores/iri_color_interesting_points_deformable.git

A.1.6 Dressing shoe demo node

The *dressing shoe demo* node integrates the decision making, pointing recognition and robot motion planning algorithms, and it is the main node. It performed all the dressing assistance.

The inputs of this node are:

- A subscriber to the *”/dressing_poses”* topic with the custom message *iri_recognize_dress_pose/pose* defined in Section A.1.3. The information about the foot posture is received through this topic.
- A subscriber to the *”/commands”* topic with the custom message *iri_dressing_shoe_node/speech_command* which was defined in Section A.1.1. The recognized voice commands are received through this topic.

On the other hand, the outputs of the node are the following:

- A publisher in the topic *”/target_marker”* with the message *visualization_msgs/Marker”*. Spherical markers are published for the visualization in rviz of the pointing recognition of the user. For further information about this message type, see:

  http://docs.ros.org/jade/api/visualization$_{m}sgs/html/msg/Marker.html$

- A publisher in the topic *”/iri_wam/pose_st”* with the message *geometry_msgs/PoseStamped.* Through this topic, the position of the robot from the WAM robot reference system is sent to the *inverse kinematics WAM* launch. For detailed information about the message type, see its documentation:

http://docs.ros.org/api/geometry_msgs/html/msg/PoseStamped.html

- A publisher in the topic *"/gripper_state"* with the message *std_msgs/Bool*. In this topic, the state of the gripper is set by setting true to open it and false to close it. For detailed information about the message type, see the following documentation:

  http://docs.ros.org/api/std_msgs/html/msg/Bool.html

- A publiser in the topic *"/text2speech"* with the message *std_msgs/String*. Feedback from the autonomous robotic system is published as a text string to be turned into speech by the *text-to-speech* node.

  The source code of *dressing shoe demo* node is provided in the following repository:

  https://gitlab.iri.upc.edu/anflores/iri_dressing_shoe_demo.git

A.1.7 Inverse kinematics WAM launch

This package provides the inverse kinematics algorithm of the WAM robot to compute the joints angles from a given point in the space. The input of this package is:

- A subscriber to the *"/iri_wam/pose_st"* topic with the message *geometry_msgs/PoseStamped*. The message contains the position and orientation for the robot, in the space from the WAM robot reference system.

The package is developed under the license of the Institut de Robòtica i Informàtica industrial (IRI). Its source code is uploaded in a private repository, therefore access is required to use this package

https://devel.iri.upc.edu/labrobotica/ros/iri-ros-pkg_hydro/metapackages/iri_wam/iri_wam_dmp_tracker/

A.1.8 Gripper state node

The *gripper state* node is used to open and close the gripper. The input of this node is the following:

- A subscriber to the topic *"/gripper_state"* with the message *std_msgs/Bool*. In this topic, the state of the gripper is set by setting true to open it and false to close it-

This node has no outputs. The source code of this node is provided in the following repository:

https://gitlab.iri.upc.edu/anflores/iri_move_gripper.git

A.1.9 Text-to-speech node

The *text-to-speech* node contains the speech-synthesis algorithm to play in speakers the feedback of the robot. It has the following input:

- A subscriber to the topic *"/text2speech"* with the message *std_msgs/String*. The text string received via this topic is the feedback of the robot.

This is the only node developed using Python. The source code is provided in the following repository:

https://gitlab.iri.upc.edu/anflores/iri_text2speech.git

A.2 Calibration of the Kinect sensors

The calibration of the Kinect sensors was very important for the success of the dressing task. This calibration was performed using an implemented node called *calibrate_camera_rviz* which uses the visualization tool rviz. Detailed information about rviz can be found in the following link:

http://wiki.ros.org/rviz

This node uses an interactive marker that one can move with the mouse to change the transform of both Kinect sensors with respect to the WAM robot reference system. In this way, the values are not set manually, but instead the position of both Kinect sensor was interactively changed. The source code is provided in the following repository:

https://gitlab.iri.upc.edu/anflores/iri_calibrate_camera_rviz.git

A.3 Execution in ROS

The execution of the different nodes and packages is described in the following instructions. These nodes must be launched from a Ubuntu PC:
First *roscore* must be launched in a terminal:

```
$ roscore
```

In other terminal, the WAM robot driver node is launched launched by doing:

```
$ roslaunch iri_wam_bringup iri_wam_bringup.launch
```

Opening a new terminal, the inverse kinematics node is executed as:

```
$ roslaunch iri_wam_dmp_tracker test.launch
```

Another terminal is required to launch the *rosserial_windows* node to receive the data from Windows:

```
$ rosrun rosserial_server socket_node
```

Open a new terminal and launch the node *recognize shoe color*:

```
$ roslaunch iri_color_interesting_points_deformable

    iri_color_interesting_points_deformable_and_CAM_NewCalibration
    .launch
```

On a new terminal execute the node to calibrate the Kinect 1:

```
$ rosrun iri_calibrate_camera_rviz calibrate_kinect1
```

And the same for the Kinect 2:

```
$ rosrun iri_calibrate_camera_rviz calibrate_kinect2
```

Finally run the *dressing shoe demo* node in a new terminal:

```
$ roslaunch iri_dressing_shoe_demo demo.launch
```

Finally, the Kinect 2 data socket must be executed from Windows in order to receive all the Kinect 2 data.