אוניברסיטת בן-גוריון בנגב
Ben-Gurion University of the Negev

# *Lower body robotic trainer simulator*

## Final report, 2nd semester 2024


**Institution:** Ben-Gurion University, Faculty of Engineering, Department of Industrial Engineering & Management

**Written by:** Shelly Safrai

**Directed by:** Prof. Yael Edan & Ms. Maya Krakowski


**Project number:** 2024-01-038

**Submission date:** 20/08/2024

# Abstract

This project addresses the growing challenge in elder care, where the number of available caregivers is declining due to demographic shifts. Given the benefits of physical exercise for older adults and the risks associated with inactivity, this project aims to develop a system using a humanoid social robot (HSR) to motivate the elderly (ages 65+) to exercise more frequently. The primary objective is to design and implement lower body exercises that can be performed safely in a seated position, with the ultimate goal of reducing the need for caregivers and promoting physical activity among older adults.

The project involves developing a system using a Humanoid Social Robot (HSR) to guide elderly individuals through seated lower body exercises. I began the process by characterizing the exercises programmed into the system. Following that, I developed the system using Python. Finally, I conducted an experiment to test the system and evaluate the accuracy of the robot's camera.

The existing system includes the design of 5 exercises that are safe and effective for the elderly, focusing on strengthening the lower body, particularly the hips, knees, and ankles. The system includes simulator software that acts as a robotic trainer, demonstrating exercises and guiding the trainee. It also features a ZED-2i camera, a user interface, and provides voice feedback on successful performances. An experiment was conducted to evaluate the accuracy of the ZED-2i camera in capturing the movements of trainees in various seated positions. Following the experiment, analysis was performed to determine the optimal configuration for the system.

In conclusion, I developed a system with significant potential for physical therapy training among the elderly. This system could enhance motivation for physical exercise in this population and address the challenge of the shortage of physical therapists.

<u>Keywords:</u> Human-Robot Interaction (HRI), Humanoid Social Robot (HSR), Robotic Trainer, Lower Body Strengthening, Chair Exercises, Camera-Based Movement Tracking.

## Table of Contents

## List of Figures

# List of Tables

# 1 Introduction

## 1.1 Problem Description

SAR's (socially assistive robots) are designated to assist people by providing social, cognitive, or physical support [1] [2] . As artificial intelligence and engineering technology continues to develop, we are encountering even more sophisticated social robots [4].

Over the last years the issue of elderly care has become a growing problem. The number of working-age adults available to support elderly individuals is decreasing due to declining birth rates and increased longevity [5]. Therefore, robots are projected to play a significant role in the problem of elder care.

Physical exercise benefits older individuals by enhancing their immune systems, physical fitness, sleep quality, and stress management [6]. And still, more than a quarter of the world's adult population are insufficiently active [7].

Physical inactivity is a modifiable risk factor not only for adults, for cardiovascular disease and a widening variety of other chronic diseases and causes over 3.2 million early deaths globally each year [7].

## 1.2 Objectives

In this project, I will develop a system that uses a humanoid social robot (HSR) as a robotic trainer to motivate the elderly population (ages 65+) to exercise more frequently. This project is a continuation of previous work that focused on exercises for the upper body. My focus will be on developing exercises for the lower body. All exercises will be performed in a sitting position to ensure the safety of older trainees who may lose their balance while standing. To create an effective training program, I will conduct an experiment to evaluate the accuracy and repeatability of the robot's camera capture of the trainee in various sitting positions. Following the experiment, I will analyze the results and provide conclusions.

Main objectives:

- ✓ Characterize physiotherapy exercises that will help to strengthen the trainee's lower body.
- ✓ Develop a friendly system that is easy to use and will motivate elderly individuals.
- ✓ Reduce the need for caregivers.

# 2 Literature Review

## 2.1 Exercise

### 2.1.1 Physical Training

Physical training is defined as any bodily movement produced by skeletal muscles that requires energy expenditure[7]. Physical activity refers to all movement including during leisure time, for transport to get to and from places, or as part of a person's work.

Physical inactivity causes over 3.2 million early deaths globally each year[8]. It is also a modifiable risk factor for cardiovascular disease and a widening variety of other chronic diseases, including diabetes mellitus, cancer (colon and breast), obesity, hypertension, bone, and joint diseases (osteoporosis and osteoarthritis), and depression [6]. The World Health Organization [7]presented that more than a quarter of the world's adult population are insufficiently active, around 1 in 3 women and 1 in 4 men are not doing enough physical activity to stay healthy, and there has been no improvement in global levels of physical activity since 2001.

Additionally, there is a correlation between physical fitness and better reaction times, less frontal brain atrophy and superior functioning in the executive control processes [9].

In order to live a healthy life, doctors recommend maintaining a nutritious diet, performing physical exercise, and stimulating the brain with intellectual or social activities. Therefore, engaging in any activity that enhances blood circulation and keeps the brain active is crucial. In particular, physical activity plays a significant role in maintaining a healthy lifestyle and elevating one's sense of wellbeing [5].

### 2.1.2 Physical Training for the Elderly

Elderly's general health can be effectively maintained and improved with physical exercise. Warburton demonstrated that regular exercise benefits older individuals by enhancing their immune systems, physical fitness, sleep quality, and stress management [6].

To deal with inactivity among older adults several studies suggest solutions that can help motivate the elderly population to engage in physical activity.

Walking is crucial for slowing the progression of the diseases mentioned above, especially for elderly individuals [5]. Additionally, keeping an eye on the proper gait parameters when walking may help measure balance and avert falls, which are another source of impairment. Walking is also an extremely accessible type of physical exercise because it is free, accessible to all abilities, equipment-free, and can be performed anywhere. In addition, walking in an open space fosters social interaction, which makes it a social activity that can lift an individual’s spirits.

There is another form of exercise for the elderly, the kind of sitting activity known as "chair exercise" or "chair aerobics" which is guided by a robot trainer. This kind of activity includes training exercises while sitting in a chair. Chair exercises are well known for their safety, accessibility to people with limited mobility, and health advantages, which include enhanced muscle strength, flexibility, and memory recall in addition to increased performance on daily chores [10].

For a healthy physical activity routine, The World Health Organization gives recommendations for individuals ages 65+, as followed:

- At least 150 min of moderate-intensity aerobic activity, or at least 75 min of vigorous-intensity aerobic activity, or an equivalent combination.
- Aerobic activity should be performed in bouts of at least 10 min duration.
- For additional health benefits, undertake up to 300 min of moderate-intensity or 150 min of vigorous-intensity aerobic activity, or an equivalent combination.
- People with poor mobility should do balance exercise to prevent falls on 3 or more days.
- Muscle-strengthening activities should be done on two or more days.
- If older adults are unable to do the recommended amounts of physical activity due to health conditions, they should be as physically active as they are able.

National Health Services also gives a sample of exercises recommended for older adults [11] :

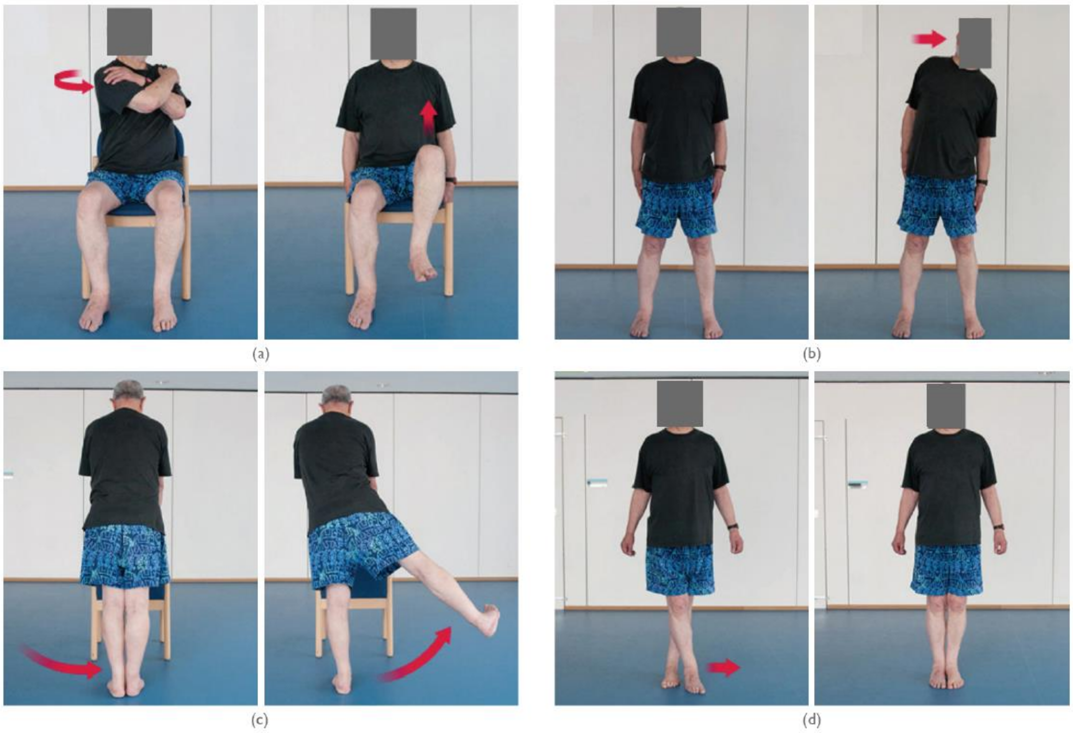



**Figure 3.** A sample of exercises recommended by National Health Services in the UK for older adults: (**a**) sitting exercises; (**b**) flexibility exercises; (**c**) strength exercises; and (**d**) balance exercises.

### 2.1.3 Physiotherapy

The main goal of physiotherapy is to help people affected by injury, illness or disability reduce pain and restore or maintain optimal physical functioning. A wide range of non-pharmacological treatment modalities can be accessed by physiotherapists, including manual therapies, electrophysical agents, thermotherapy, hydrotherapy, graded exercise, education and advice. Physiotherapy helps to encourage development and facilitate recovery, enabling people to stay in work while helping them remain independent for as long as possible [12].

Physios use their knowledge and skills to improve a range of conditions associated with different systems of the body, such as [13]:

- Neurological (stroke, multiple sclerosis, Parkinson's).
- Neuromusculoskeletal (back pain, whiplash associated disorder, sports injuries, arthritis).
- Cardiovascular (chronic heart disease, rehabilitation after heart attack).

- Respiratory (asthma, chronic obstructive pulmonary disease, cystic fibrosis).

#### 2.1.4 Physiotherapy for the Elderly

The body's systems become less efficient as a person ages biologically, but each person can retain optimal function throughout their life by making the most use of these systems. Physiotherapists play a crucial part in helping elderly individuals maximize the usage of several bodily systems to improve their mobility and independence. Physiotherapists can also help elderly persons stay comfortable and pain-free while maintaining or even improving functional mobility is not a realistic aim [14] .

## 2.2 Social Robots

Humanoid social robots (HSRs) are artificial creations, appearing either in physical or digital forms, that emulate human characteristics in their appearance or behavior, specifically designed for interacting with humans [3].

Examples of HSRs include embodied conversational agents serving as healthcare providers, consumer robots specializing in education and home care, and robotic trainers designed primarily for human interaction [3]

As artificial intelligence and engineering technology continues to develop, we are encountering even more sophisticated social robots [4].
The significance of the expected rise of social robots in human society is highlighted by the efforts of major international research agencies, such as the National Science Foundation and the European Commission, to enhance their support for robotics development and to develop strategic visions for the seamless integration of robots into all facets of society. Furthermore, corporations such as Facebook, Amazon, and Google, along with a plethora of smaller start-ups and medium-sized digital organizations, are still making substantial investments in the advancement of social robots and artificial intelligence. These and numerous other instances foreshadow the increasing prevalence of social robots [4].

## 2.3 Human robot interaction

#### 2.3.1 General

HRI can be divided into four areas of application [15]:

- Human supervisory control of robots in performance of routine tasks.

- Remote control of space, airborne, terrestrial, and undersea vehicles for nonroutine tasks in hazardous or inaccessible environments.
- Automated vehicles in which a human is a passenger, including automated highway and rail vehicles and commercial aircraft.
- Human–robot social interaction, including robot devices to provide entertainment, teaching, comfort, and assistance for children and elderly, autistic, and handicapped persons.

### 2.3.2 Issues In Human-Robot Interaction

HSRs are created to satisfy the human user's needs. However, HSRs can violate user's expectations and diminish feelings of closeness and trust. Therefore, social interaction has been named one of the ten grand challenges the field of robotics is now facing. One key factor contributing to this issue is the inherent limitation in the way modern HSRs comprehend interactional history compared to humans. These robots lack the capacity to make sense of past interactions and struggle to apply that knowledge to navigate new social situations effectively. Even in cases where certain HSRs can remember specific parameters for a particular user, this knowledge fails to translate into real-time personalization or message tailoring based on the user's verbal and nonverbal cues. Robot memory lacks persistence and sufficiently refined searchability for sensible ongoing social interactions. Even for the small part of HSRs that can maintain a memory of past interactions, the retrieval process is limited to a few task-specific queries [1] [2] .

Another reason is that HSRs are limited in the tasks they are programmed to perform. The robot is constrained to a small set of possible responses and has minimal conversation control. Moreover, HSRs lack the autonomy to initiate changes in topics or tasks, and they struggle to interrupt or terminate interactions with human users. This lack of conversational control impedes their ability to dynamically adapt to the evolving needs and preferences of users, hindering the fluidity and naturalness of human-robot interactions [16].

Researchers in the field of human-robot interaction and neuroscientists collaborating on robot-related studies share a common goal of enhancing seamless and effective social interactions between robots and humans. This joint effort should ultimately enable society at large to take advantage of the potential of robots to provide economical care, company, and coaching [3].

### 2.3.3 Elderly – Robot Interaction

The number of working-age adults available to support elderly individuals is decreasing due to declining birth rates and increased longevity [5]. Therefore, robots are projected to play a significant role in the problem of elder care. In addition to rehabilitation and physical help, researchers are focusing on the use of social robots for mental support. That is because older people in senior care have trouble getting along with their caregivers due to mental illnesses, cognitive and socioemotional decline, and other factors. As these impacts differ greatly from person to person, care should be tailored to meet the needs of everyone. But such care is not possible for social robots as they exist today. When actual robots successfully respond to older people, they converse with them more than they do with virtual robots. Therefore, depending on how effectively the system functions, having a physical body is useful in generating high involvement [17]

SAR's (socially assistive robots) are designed to assist and meet the demands of a particular user category (such as elderly people or children with autism) through interactions with these users. Those robots are created to help people by providing social, cognitive, or physical support [5].

The primary functions of social assistive robots (SAR) for the elderly include keeping monitoring them, assisting them with everyday tasks, and providing support through social rather than physical interaction. On the other hand, rehabilitation robots designed for physical training typically lack the social aspect and a person-like embodiment, which is considered to be helpful in scenarios involving senior care. The majority of systems now in use for senior physical exercise opt for screen-based interfaces over robots altogether. The goal exercise can be realistically visualized by an embodied conversational agent (ECA) or a comparable 3D avatar on a screen. But when it comes to engaging the subject, screen-based solutions compared to social robots are somewhat at a disadvantage due to their lack of real and tangible embodiment. According to Marcel Heerink [18] the social skills help older people embrace the robot more, which may boost their motivation to utilize it. Therefore, older adults may be more motivated to exercise if the system is more appealing to them. In an exercise scenario, [19] compared user responses to real and virtual robots, and they found that users perceive the physical robot as more engaging, entertaining, and a better exercise partner. Similarly, real robots elicited better mimicking responses than simulated robots in a physiotherapy scenario, as demonstrated by [20].

Piazzo claims that the degree of tolerance that users find acceptable is what determines a robot's success [5]. If a person meets the following criteria, they are more likely to accept and utilize a particular tool: they are motivated to use it, they believe it to be simple to use, and they feel both physically and psychologically at ease with it. Consequently, such factors must be considered when designing a social robot.

## 2.4 Lower Body Trainers

There is a large population with lower limb mobility problems, particularly among the elderly. Shi present that in China for example, there are over 40 million senior impaired individuals who are unable to walk because of age, and about 15 million disabled persons in China who have lower limb motor dysfunctions such as cerebral palsy, hemiplegia, and paraplegia. Active rehabilitation training ought to begin as soon as possible for those individuals. In the meanwhile, less than 20,000 technical personnel are available to assist the approximately 350,000 people who are in urgent need of rehabilitation there. Therefore, Robots for the rehabilitation of lower limbs are very important [22].

There are two kinds of lower limb rehabilitation robots. Those kinds differ from each other according to exercise posture. The first kind is a sitting/lying training robot, designed for use in sitting or lying exercises. Those robots help patients who are weak in their muscles and are unable to stand or walk comfortably. The patient may be more self-sufficient and able to concentrate on the training if balance concerns are eliminated. With the help of this type of robot, the patient can improve joint mobility, coordination, and muscle strength. Another kind of robot is designed for use in standing and walking exercises. The gait training robot was designed to train on a treadmill with a body weight support system or on the ground. The gait training robot, however, is only appropriate for patients who have adequate endurance and ability to stand[23].

Colombo have focused on robots for rehabilitation because they provide some advantages over physiotherapists [24]. For example, robots designed for treadmill locomotion training are largely intended to help meet the increase demand for caregivers and improve it since activity is physically demanding and laborious from an ergonomic standpoint and there is a shortage of caregivers. Increased repetition counts and training session time are possible without putting too much strain on the body [25]. While a therapist's performance could change day to day and different therapists may use different intervention strategies, a robot adheres to a predetermined control algorithm and offers the patient systematic intervention. Furthermore, robots can use a variety of sensors to gather and store data, including position, velocity, contact force, and biosignals. This quantitative data can be utilized to adjust the behavior of the robot to the patient's current state or for additional offline analysis that results in an objective assessment of the patient's recovery. Rehabilitation robots can also perform a variety of exercises and movement patterns. Furthermore, the robot can be integrated with games or a virtual reality system to encourage the patient's active participation [23].

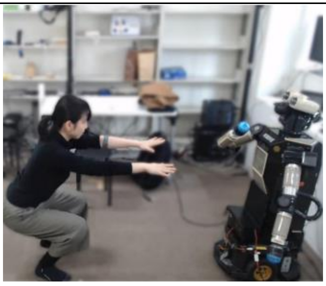
## 2.5 Robotic Trainers

### 2.5.1 Guiding robots

Human like social robotic trainers. Those robots guide and show the trainers the exercises they need to perform. Some of these robots have been programmed to guide only movement in the upper body and some can guide movement in the lower body as well.

Table 1 - Types of guiding robots

| Name | Picture | Description | References |
|---|---|---|---|
| Aldebaran Nao |  *Figure 1 - Aldebaran Nao* | A self-guided fitness robot designed to support seniors in their daily exercise routines.<br>The system operates in two modes:<br>a) A human instructor verbally explains and instructs the robot on the specific exercise.<br>b) The robot demonstrates the exercise to an elderly individual, observes their movements, and provides feedback. | [21] |

| | | | |
|---|---|---|---|
| No name | <br>Figure 2 - Squats robotic trainer | A robotic trainer that was designed to guide its trainers to do the “squat” exercise.<br>The robot has a human-like movement in its neck and arms and can move with its wheeled base.<br>The robot can’t make the “squat” exercise by itself, but it can introduce itself, instruct the user and count time and sets. | [26] |
| RoboPhilo | <br>Figure 3 - RoboPhilo | A programmable humanoid robot that was designed to show and monitor the user’s physician-prescribed exercise program. It has 20 servomotors that enable the turning movements of the head, waist, thighs, and joint movements of the limbs. | [27] |
| Pepper | <br>*Figure 4 - Pepper* | A robotic walking instructor designed for seniors.<br>The system operates in three modes:<br>a) The robot tracks and follows a human user.<br>b) The robot is manipulated and guided by the human operator through direct physical contact.<br>c) The robot guides and influences a person's path toward a specific goal. | [5] |
| Socially Assistive Robot (SAR) | <br>*Figure 5 - SAR* | A socially assistive robot designed to oversee, guide, evaluate, and encourage users to participate in fundamental physical activities. The robot prompts users to engage in seated arm gesture exercises during exercise sessions. In a one-on-one interaction setting, the user sits facing the robot. The robot takes charge of leading exercise sessions, assessing user performance, and delivering real-time feedback.<br>The system operates in four modes:<br>a) The Workout game.<br>b) The Sequence game.<br>c) The Imitation game.<br>d) The Memory game. | [19] |

| | | | |
|---|---|---|---|
| Taizo |  Figure 6 - Taizo | A socially assistive robot designed to engage older adults in seated physical activities.<br>Taizo, characterized by its diminutive size comparable to a table lamp. Unlike traditional robots, it lacks a visual feedback display screen. | [11] |
| Robo Coach Xuan |  Figure 7 - Robo Coach Xuan | A socially assistive robot tailored to engage older adults in seated physical activities.<br>Xuan, with its substantial, nearly human-sized structure, features an arm and a display screen that offers visual feedback during the workout sessions. | [11] |

### 2.5.2 Supporting robots

Robotic devices that are physically worn on the body. In this section I will first present all types, and then examples of the different types.

Table 2 - Types of supporting robots for the lower body

| Name | Description |
|---|---|
| **Treadmill gait trainer** | A partial body-weight support treadmill training, its goal is to improve functional mobility. Based on exoskeleton type robots in combination with a treadmill. While the patient walks on a treadmill, three therapists support the patient's legs and hips. An overhead harness supports a portion of the patient's body weight (Díaz et al., 2011). |
| **Foot-plate-based gait trainer** | Based on programmable foot plates. The patient's feet are placed on distinct foot plates, and the robotic system manipulates these foot plates' motions to replicate various gait patterns (Díaz et al., 2011). |
| **Overground gait trainer** | Based on servo-following robots that walk over ground in sync with the patient. Instead of putting patients through preset movement patterns, it aims to give them the freedom to move however they please (Díaz et al., 2011). |
| **Stationary gait and ankle trainer** | A robotic device made to simulate walking and train human ankle and knee movements. The trainer's objectives are to achieve effective muscle strengthening, endurance development, joint mobility, and movement coordination. Based on robotic systems that prioritize controlled limb |

| | |
|---|---|
| | movements for the best possible therapeutic and functional outcomes (Díaz et al., 2011). |
| **Active foot orthoses (AFO)** | Active foot orthoses are actuated exoskeletons that the user wears while walking overground or on a treadmill. The goal is to control position and motion of the ankle, compensate for weakness, or correct deformities. [25] |
| **Hip exoskeleton robot (HER)** | Based on the hip exoskeleton's structure. Mostly worn at the thigh's root and hip. With hip joint flexion/extension/abduction and adduction rehabilitation exercises, these HERs can help patients (Zhou et al., 2021). |
| **Knee exoskeleton robot (KER)** | based on a robot knee exoskeleton that can support the patient's knee joint with strength while they are sitting or standing. Improved leg movement ability, decreased movement consumption, and power and assistance for lower limb movement are all possible with KERs (Zhou et al., 2021). |

<u>Treadmill gait trainer</u>

*Figure 9 - **The Lokomat** (Hacoma AG)*

*Figure 8 - **The LokoHelp** (LokoHelp Group)*

*Figure 10 - **The ReoAmbulator** (Motorika Ltd)*

<u>Foot-plate-based gait trainer</u>

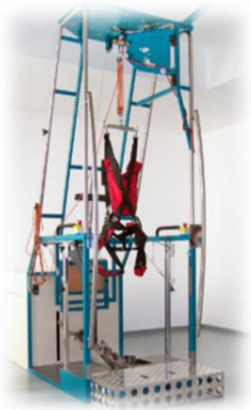


*Figure 12 - **The Gangtrainer GT I** (Reha-Stim)*

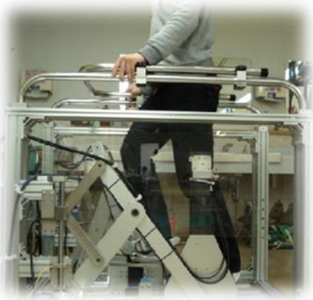


*Figure 11 - **The GaitMaster5** (Dr. Hiroaki Yano, University of Tsukuba)*

## Overground gait trainer

*Figure 15 - **The KineAssist** (Kinea Design, LLC)*

*Figure 14 - **The ReWalk** (ARGO Medical Technologies Ltd)*

*Figure 13 - **The HAL - hybrid assistive limb** (Professon Sankai, CYBERDYNE Univeristy of Tsukuba*

## Stationary gait and ankle trainer

*Figure 16 - **The MotionMaker** (Swortec SA)*

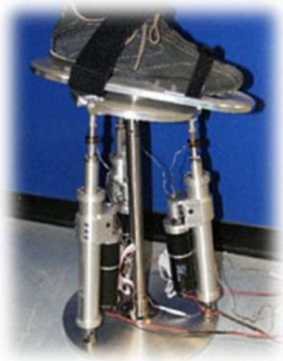


*Figure 17 - **High Performance Ankle Rehabilitation Robot** (The Istituto Italiano di Tecnologia)*

## Active foot orthoses

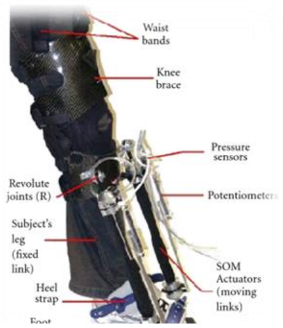



*Figure 18 - **The Robotic Gait Trainer** (Arizona State University)*

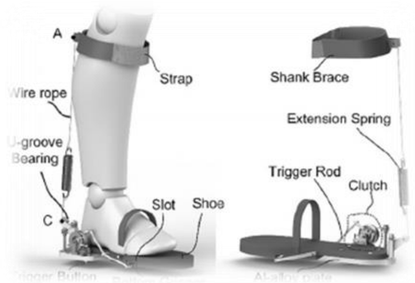



*Figure 19 - **Lightweight passive ankle exercise device** (Wang e al)*

## Hip exoskeleton robots

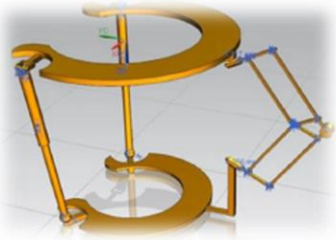


*Figure 21 - **Hip joint assistant asymmetric parallel mechanism** (Ding et al)*

*Figure 20 - **GEMSv2** (Xue et al)*

*Figure 23 - **NREL-Exo** (Zhang et al)*

*Figure 24 - **HipBot** (Zhang et al)*

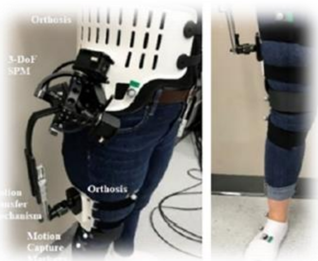


*Figure 22 - **GRF/M** (Mahdavian et al)*

## Knee exoskeleton robots

*Figure 25 - **FUM-Knee Exo** (Kamali et al)*

*Figure 26 - **Robotic knee exoskeleton** (Games et al)*

In conclusion, I saw that there are many more **supporting robots** as a **lower body** rehabilitation solution **than guiding robots**. I can infer that this is mainly since most of the robotic trainers exist today, are intended for the populations that have mobility problems, and not for those who are in good health conditions. Individuals with lower limb limitations may not be able to use such trainers. However, there is an apparent need for the elderly with high lower body function, of social robotic trainers. The development of such a robot may improve a human health (especially elderly's) and prevent health problems that may arise in the future.

# 3 Methods

## 3.1 Overview

The system I developed is a robotic trainer designed to allow older people to perform physical exercises to strengthen their lower body, independently without the help of a caregiver. Based on the literature survey, the system was characterized first by the lower body exercises it is going to offer. All the selected exercises are performed in real time on a chair, to allow the elderly to exercise safely without fear of losing their balance. The system development includes programming the exercises that the robot performs by simulator software, configuring the camera to monitor the trainee's movements, creating the feedback system for the trainee, and designing the interface screens. Later in this section, I will describe the experiment I conducted to verify the accuracy of the robot's camera when capturing the trainee seated at different chair angles, as well as the analysis of the results.

## 3.2 Hardware

The system consists of 4 components:

- ✓ <u>Humanoid social robot (HSR)</u>- version "Poppy Humanoid v1.0.2". This robot version is 83cm and 3.5kg. Its embedded system is Android XU4 with Ubuntu 14.04. It has 25 actuators and a large FOV and HD camera.
  The robot serves as a robotic trainer that guide the trainee how to perform the exercises [29].
- ✓ <u>V-REP simulator</u>- the HSR is currently missing in the lab, so the system was developed using simulator software that simulates the real HSR.
- ✓ <u>Zed camera</u>- version "ZED 2i". A deep sensing camera that monitors the user's performances during the session. The camera works by detecting the angles between different joints in the trainee's body [30].
- ✓ <u>Trainee</u>- An elderly person aged 65+ who performs the training.

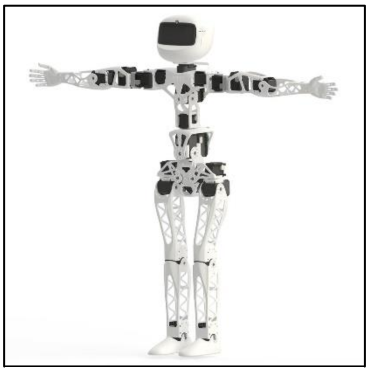


Figure 28 - **Poppy Humanoid v1.0.2**


Figure 27 - **Zed Camera**

| Parameter | ZED-2i |
|---|---|
| **Technology** | Stereo vision and neural networks |
| **Depth range (m)** | 0.2m–20m |
| **RGB resolution (pixel)** | 3840 X 1080 |
| **Depth resolution (pixel)** | 3840 X 1080 |
| **Field-of-view depth (FOV)** | 110(H) X 70(V) X 120(D) |
| **Frames per second (FPS)** | 30 FPS |
| **Weight (g)** | 166 g |
| **Cost ($)** | 500 $ |

*Figure 29 - Technical specifications of the ZED-2i camera*

## 3.3 Software

The programming language I used in my development is python 3, because all Poppy's libraries are written in this language.

The software section includes programming the exercises for the robot using simulator software, configuring the camera to monitor the trainee's movements, creating the feedback system for the trainee and designing the interface screens.

## 3.4 Training Process Overview

At the beginning of the training the robot introduces itself and displays the home screen. To start the workout, the trainee presses the "Start" button. The robot then instructs the user to wave "hello" for identification. Once the camera detects the user, the training begins.

The training starts with the robot instructing the user to perform four exercises in a random order with the right leg, demonstrating each exercise for the trainee. After completing these, the trainee performs the same exercises with the left leg, again in a random order. The session concludes with a fifth exercise, alternating between the right and left legs.

Throughout the training, the robot announces and displays on the screen the names of the exercises and counts the number of repetitions each time it recognizes the trainee performing the exercise correctly. After each exercise, the robot provides feedback on the trainee's performance. At the end of the training, an Excel file documenting the user's performance is generated automatically. This documentation will be used for further analyze the trainee's performance and the camera's ability to effectively capture their movements.

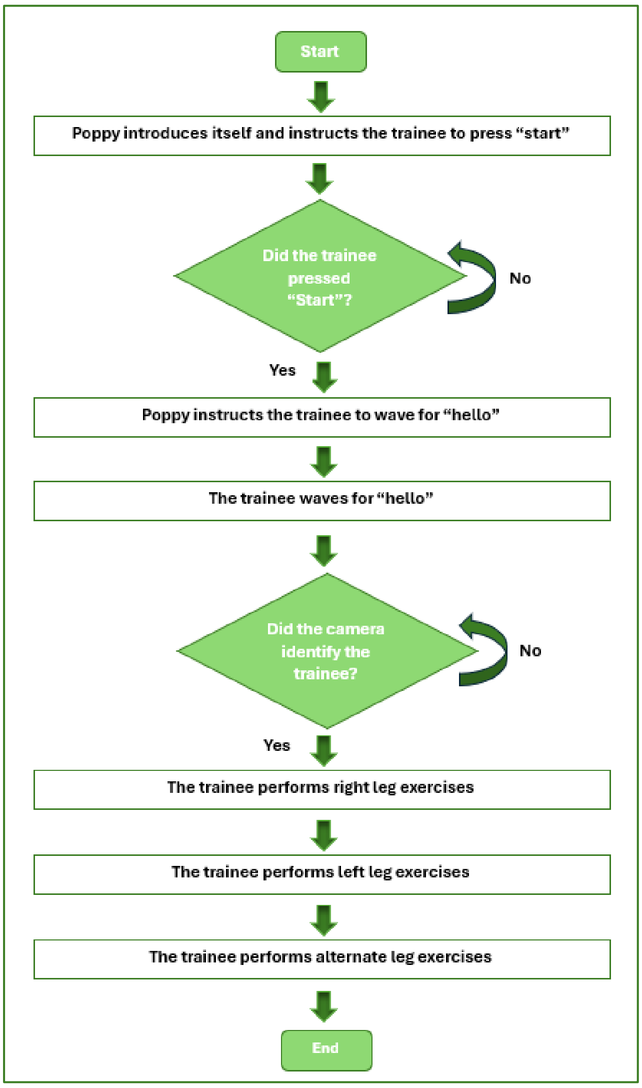



*Figure 30* - ***Training Process Diagram***

## 3.5 Code Overview

In this section I will explain how the project code is structured. A class diagram is attached below.

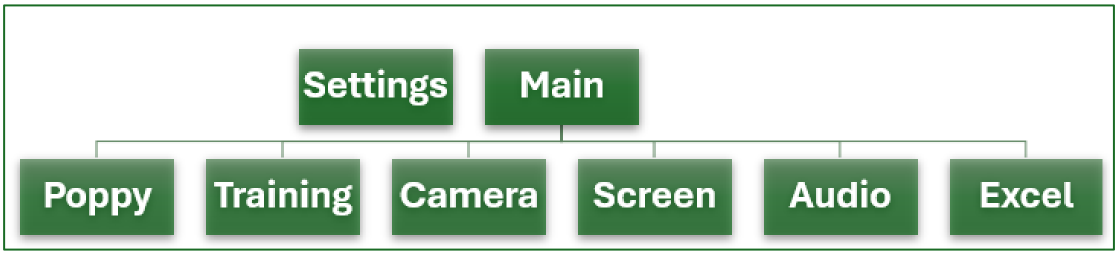



*Figure 31 - **Classes***

- **Settings**- This class defines all global variables of the system, that need to be accessed by all classes. These variables facilitate synchronization between classes and ensure the smooth execution of the training. Examples of them are the number of exercises and repetitions, status of other departments completing their tasks, and more (Settings).
- **Main**- This class defines the values of global variables used throughout the training process. Additionally, this class executes the program by creating objects from various classes and activating them (using the "start" command) (Main).
- **Poppy**- This class defines the exercises performed by the robot. Activating this class prompts the robot to execute the specified exercises. Each exercise involves moving the joints of "Poppy Humanoid" to specific angles in a predetermined sequence (Poppy).
  Important libraries- poppy humanoid.
- **Training**- This class defines the training program flow. It specifies the sequence of exercises and monitors their performance. The class manages transitions between exercises, the end of the training, triggers screen displays, and audio feedback based on success or failure (Training).
  Important libraries- random.
- **Camera**- This class defines the camera's control of the exercises. Each exercise in training has a built-in function in this department that verifies if the trainee performed the exercise correctly and achieved the desired number of repetitions, typically 10 (Camera).
  Important libraries- pyzed.sl, numpy.
- **Screen**- This class defines the different screens displayed on the robot's screen (its head) during training. This forms the UI (user interface). All screen images are stored in the

"Pictures" folder (Screen, Pictures).

Important libraries- Tkinter, PIL.

- **Audio**- This class defines audio files played during training. These files include the names of the exercises, feedback on the user's performance (very good, excellent, good job), instructions for starting a workout and a farewell message at the end. All audio files are stored in the "Audio" folder (Audio, Sounds).

  Important libraries- pygame, winsound.

- **Excel**- This class defines actions that automatically generate an Excel file at the end of each training session, containing two types of sheets:
  - "Success" sheet- Lists all the exercises and records the number of successful repetitions performed by the trainee. For example, if the trainee attempted 10 repetitions but only did 7 correctly, the sheet will record 7.
  - Named after each exercise, these sheets log the trainee's joint angles at every moment during the training. Each joint's angles are recorded along three different axes: X, Y, and Z. The sheets also capture the angles defined by the specific exercise's function in the 'camera' class.

  These files will be used later to analyze the accuracy of the camera when the trainee is seated at different chair angles (Excel).

  Important libraries- xlsxwriter.

- **Joints**- The Joint class represents a skeletal joint in a 3D space. It is designed to handle the coordinates of the joint, store joint's data, and provide utility functions for joint's data management (Joints).
- **MP**- The MP class is designed to work with the ZED-2i camera for capturing images and performing body tracking using the ZED SDK. It leverages threading to run the camera operations in parallel. The class also integrates with OpenCV for image display and ZED SDK for further processing (MP).

## 3.6 Body Tracking Overview

I used the PyZED algorithm for my project. This body tracking module utilizes a neural network for keypoint detection, followed by depth and positional tracking through the ZED SDK module to obtain the final 3D position of each keypoint. The ZED SDK supports multiple body formats. In my project, I used the BODY38 format, which includes 38 keypoints (defined in def Get_skeleton_data())[31].

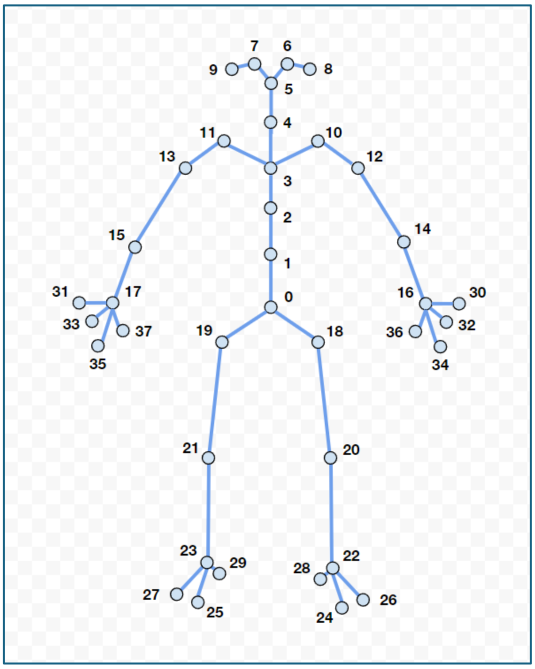



*Figure 32 - **BODY38 Tracking Overview***

Within the 'Camera' class, I wrote a function called "def Calc_angle_3D()" that calculates the angles between three different joints based on the received position data. This enables the evaluation of the correctness of the trainee's movements during exercises. The 'Camera' class includes various functions specifically designed to verify the accuracy of each exercise by evaluating the angles between three specific joints relevant to that exercise. (Camera).

## 3.7 Exercises

The exercises were carefully designed with the limitations of the target audience in mind- elderly individuals (ages 65+). These considerations ensure that all activities can be safely performed while seated in a chair, maintaining a moderate level of difficulty. All exercises were taken from "physitrack" website [32].

There are 3 types of exercises:

- Hip strengthening.
- knee strengthening.
- Ankle strengthening.

<u>Table 3</u> - The description of the different exercises

| Type | Exercise | Description | Example | Implementation |
|---|---|---|---|---|
| Knee | Knee extension | Sit on a chair. Stretch one leg out and keep it in that position for the required duration. Return your leg to the starting position and then repeat with your other leg. |   | Each side seperately. |
| | Knee flexion | Sit in a neutral posture, your shoulders back and down. Pull your heel back towards the chair as far as you can go. Relax to the neutral position and switch sides. Pull your heel back and relax. |   | Each side seperately. |
| | Sit to stand | Sit on a chair.<br>Stand up and then sit again. |   | Wasn't implemented because the robot was unstable. |
| Ankle | Ankle stretch | Sit up straight in a chair. Straighten one leg out in front of you. Keeping your | | Each side seperately. This was implemented by |

| | | | | |
|---|---|---|---|---|
| | | heel on the floor, bring your toes up towards you, then point your toes away from you. Continue this movement. |  | raising the leg from the floor, as the robot is unable to keep its foot on the ground. |
| | Soleus stretch, sliding foot under chair | Sit up straight in a chair with both legs bent and feet flat on the floor. Slide the foot of the affected leg backwards towards the chair. Endeavour to keep your heel on the ground. When you feel a stretch in your calf, hold this position. |  | Wasn't implemented because the robot cannot keep its foot on the floor. |
| | Heel raises soleus strengthening. | Sit with a weighted object on your knees.<br>Raise up and down on to your toes, controlling the movement back to the start position. |  | Wasn't implemented because the robot cannot keep its foot on the floor. |

| | | | | |
|---|---|---|---|---|
| Hips | Hip Marching | Sit up straight in a chair. Keeping your posture upright, alternately lift your knees up and down as though marching your feet on the ground. Make sure you do not lean your body backward or sideways as you do this. |  | Each side seperately. |
| | Hip abduction | Sit up straight in a chair. Keeping your good leg still, turn the thigh of your affected leg out as far as you can without twisting the body and then steadily return to the start position. You will feel this down the side of your hip. Relax and repeat. |  | Alternate. One time by the right side and one time by the left. This was implemented without the bend to facilitate the exercise and better suit the target audience. |

The selection of exercises was constrained by several factors related to both the robot and the intended users. The robot's limitations include instability, lack of flexibility to perform human-like movements, and an inability to keep its leg connected to the floor in the simulator software. After understanding the robot's constraints, the following exercises were programmed into the system:

- Knee extension
- Knee flexion
- Ankle stretch
- Hip Marching
- Hip abduction

## 3.8 Experiment objective

The experiment's objective is to evaluate the effectiveness of various chair positions (that the trainee sat on during training) relative to a ZED-2i camera in accurately capturing the movements of trainees. Specifically, it investigates how different angles (0°, 45°, 90°, 135°, 180°) of the chair in relation to the camera affect the identification of joint angles during exercises. The experiment seeks to determine, for each exercise, and for the training in its entirety, the chair position that will result in optimal identification of the trainee's movements, contributing to the enhancement of motion-tracking accuracy on the trainee.

Dependent Variable: The camera's ability to accurately capture the trainee's movements.

Independent Variable: The chair angle (0°, 45°, 90°, 135°, 180°) at which the trainee is positioned relative to the camera.

## 3.9 Participants

The experiment involved five trainees, consisting of two males and three females aged between 24 and 28 years old, students at Ben-Gurion university. These participants were selected to perform physical training exercises using the V-REP simulator, which served as a simulated environment for training with the real "Poppy Humanoid." Each participant completed five separate training sessions, during which they performed nine distinct exercises, each repeated ten times.

## 3.10 Course of the Experiment

1) The experiment was conducted on June 22, 2024, at the Human Robot Interaction Laboratory at Ben Gurion University.
2) Each participant entered the room alone to ensure individual focus and minimize distractions.
3) A ZED-2i camera was placed in a fixed position throughout the entire experiment.
4) Participants were seated in a chair, which was placed at five different angles (0°, 45°, 90°, 135°, 180°) relative to the camera.
5) For each chair position, participants performed a complete training session containing nine different exercises, with each exercise consisting of ten repetitions. The training included a demonstration of the exercises by the robot (simulator software), different screens, and voice feedback.

6) Participants were given a 10-minute break between each training session to ensure consistency and reduce fatigue.
7) The ZED-2i camera recorded the joint angles of the participants' movements along the X, Y, and Z axes, as well as the specified angles that were important for each specific exercise.
8) The data from each training session were documented in Excel files, resulting in a total of 25 files covering all exercises performed in each training by the five participants.
9) The experiment's design is depicted in the figure below:

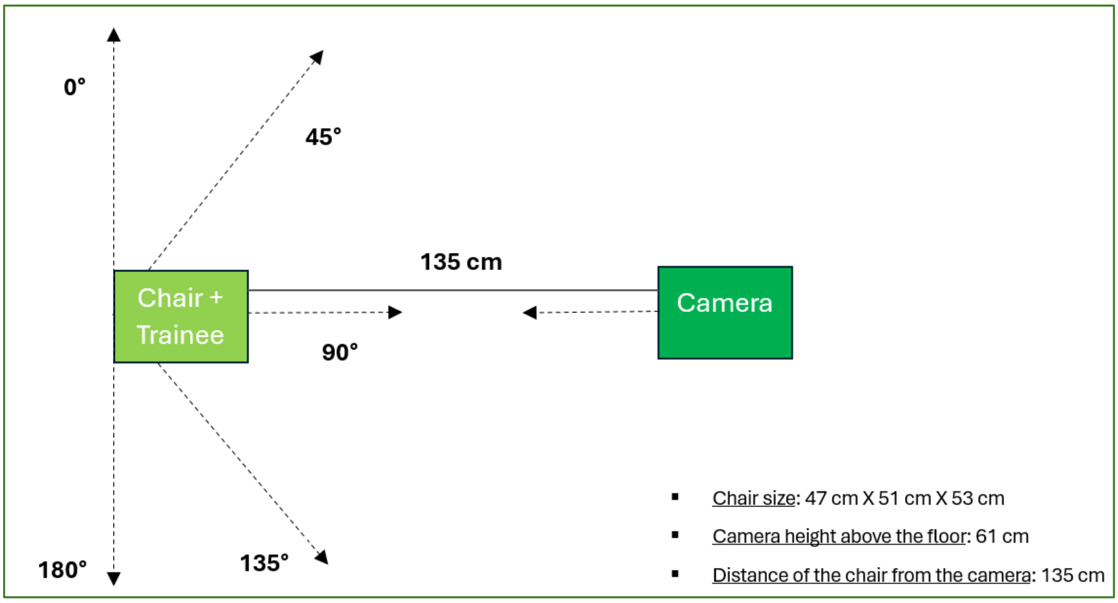



*Figure 33 - The different chair angles from camera*

## 3.11 Performance measures and Analysis techniques

**Measure of success** – refers to the percentage of successful repetitions performed by participants while doing the exercises at a specific chair angle. This was measured by two different techniques:

- Tabular Analysis **-** This method involved creating a table to display the success percentages of participants at different chair angles for each exercise. This provided a clear, quantitative comparison of the camera's ability to capture performance across various angles.
- Statistical Analysis - This technique involved performing an ANOVA test to determine if there is a significant difference between the succussed repetitions at different chair angles. It also included performing a post-hoc test to identify the specific differences

between the chair angles. This technique helped to further analyze if there was a significant difference in the camera's ability to capture successful repetitions between the different chair angles and identify the specific differences.

**Measure of repeatability -** Refers to the degree of consistent frequency in the specific joint angles measured for trainees while performing a particular exercise. This was measured by graphical analysis.

- Graphical Analysis - Using the data from the Excel files, I generated various graphs to illustrate the changes in angles throughout each exercise. There are two types of graphs:
  <u>Angle Change Over Time at a Specific Chair Angle</u> - Each line on these graphs represents one of the five participants, showing how their angles changed over the course of the exercise at a particular chair angle.
  <u>Angle Change Over Time for a Specific Participant</u> - Each line represents a different chair angle, illustrating how a single participant's angles varied across different chair positions while performing the same exercise.

For these analyses, I used 25 Excel files that represent the participant’s training results. All the results, including the graphical and tabular analyses, as well as the codes used to create the graphs and perform the statistical tests, are available on **GitHub** at the following link: https://github.com/shellysaf/PoppyProject

# 4 Results

In this section, I will analyze each exercise individually to determine the optimal chair angle for ensuring the camera captures the performance with the highest accuracy. Following this, I will conduct a statistical analysis to identify the best chair angle for performing the entire training session.

The tabular analysis is attached in the appendixes (Tabular Analysis). I used this table to analyze the initial results for each exercise, which I will detail below.

The graphical analysis of each exercise is based on the angles defined for each exercise according to its specific function within the camera class (Exercise check in Camera class). For a repetition to be considered successful, it must move from a defined low range of angles to a defined high range of angles, or vice versa.

## 4.1 Knee extension

<u>Tabular analysis:</u>

- The camera captures the user's movements most effectively when the trainee is positioned at a small chair angle in front of the camera (45° or 135°) or at a vertical angle to the camera (0° or 180°).
- The camera captures the user's movements in 50% success rate and less when the trainee is positioned at a chair angle in front of the camera (90°).
- The camera captures the movement of the working leg at all these angles, regardless of whether the working leg is close or far from the camera.

<u>Graphical Analysis:</u>

To investigate the reasons for the poor reception of the user's movements by the camera at a 90° chair angle, I will compare it with a chair angle that has been found to be effective at 135°.

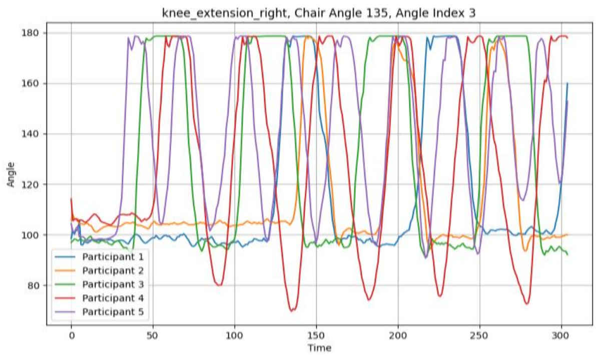



*Figure 35 – HIP, KNEE, ANKLE angle over time, right leg, chair angle of 135°*

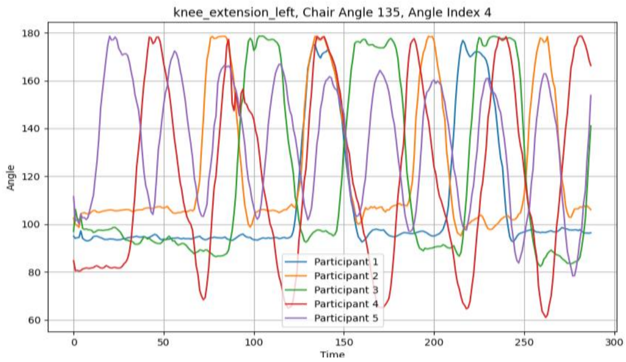



*Figure 34 - HIP, KNEE, ANKLE angle over time, left leg, chair angle of 135°*

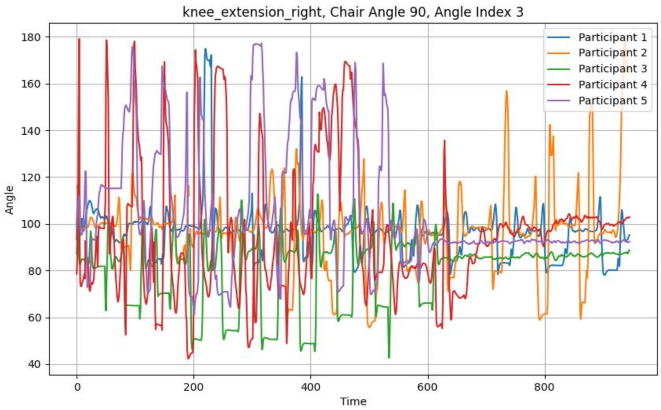



*Figure 36 - HIP, KNEE, ANKLE angle over time, right leg, chair angle of 90°*

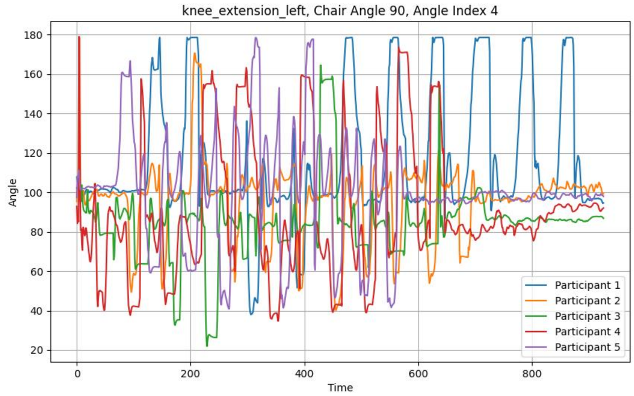



*Figure 37 - HIP, KNEE, ANKLE angle over time, left leg, chair angle of 90°*

The difference between the two types of chair angles (90° and 135°) are clear. At a chair angle of 90°, there is no consistent frequency in the performances of the various participants. Additionally, some participants were unable to reach the joint angles defined in the initial function (from a range of 150°-180° to 80°-125°).

I will examine the exercise performances of the two participants who managed to perform the fewest successful repetitions at a chair angle of 90° (participant 1 and 3).

| Angle | 90 |
|---|---|
| Exercise | knee_extension_right |

| Row Labels | Sum of Num of success |
|---|---|
| 1 | 2 |
| 2 | 3 |
| 3 | 0 |
| 4 | 7 |
| 5 | 9 |
| **Grand Total** | **21** |

*Figure 38 - participant's number of successful repetitions (out of 10), right leg, chair angle of 180°*

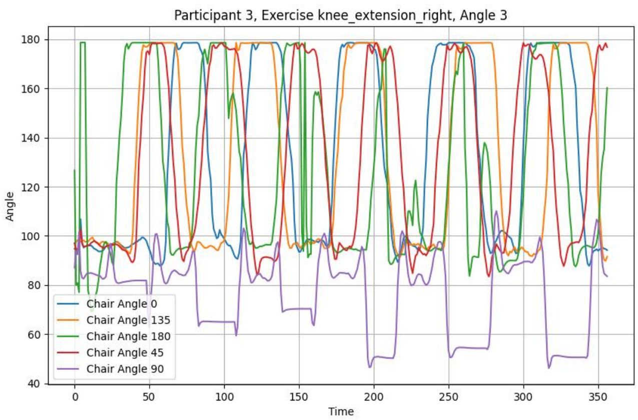



*Figure 40 -participant 3, HIP, KNEE, ANKLE angle over time, right leg, all chair angles*

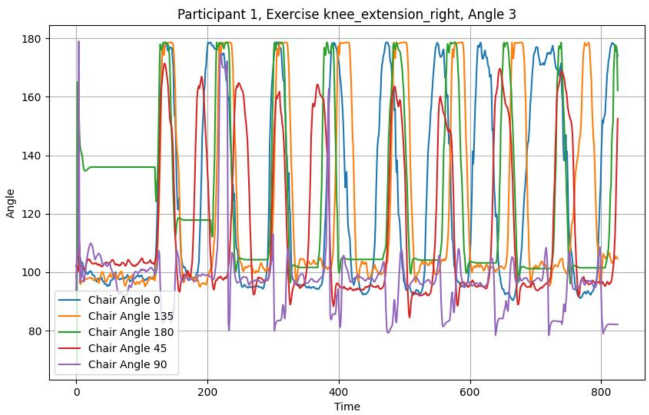



*Figure 39 - participant 1, HIP, KNEE, ANKLE angle over time, right leg, all chair angles*

Constant frequency can be seen in the angle changes at each of the different chair angles. The purple graph, representing the 90° chair angle, is unusual and indicates that even when the exercise is performed the same way by the same participant, the camera fails to capture it correctly at this angle. The range of the trainee's angles is limited, and the perceived angles are lower than the true angle.

The same analysis was done for the other participants, and the results were the same (Knee extension).

## 4.2 Hip marching

Tabular analysis:

- The camera captures the user's movements most effectively when the trainee is positioned at a small chair angle in front of the camera (45° or 135°) or at a vertical angle to the camera (0° for the right side, 180° for the left side).
- The camera captures the user's movements in almost 100% success rate even when the working leg is the leg farthest from the camera (0° for the left side, 180° for the right side) or when the trainee is sitting directly in front of the camera at a chair angle of 90°.

## 4.3 Knee flexion

Tabular analysis:

- In the right side, all participants were able to perform all ten repetitions at all chair angles with 100% success rate.
- On the left side, the camera captures the user's movements most effectively when the trainee is positioned at a small chair angle in front of the camera with the left leg closest to the camera (135°) and when the working leg is completely vertical to the camera but when the left leg is the farthest from the camera (0°). The exercise is also well captured with 100% success rate when the trainee is sitting directly in front of the camera (90°).
- On the left side, at a small chair angle in front of the camera with the left leg farthest from the camera (45°), high success rates of 92% were found.
- On the left side, when the working leg is completely vertical to the camera and close to the camera (180°), the success rate is just 80%.

Graphical Analysis:

I will try to investigate the reasons for the poor reception of the user's movements by the camera at a 180° chair angle with the left leg.

According to the table on the right, it seems that all the participants were able to perform the exercise in its entirety with the left leg, except from participant number 1.

| | |
|---|---|
| Angle | 180 |
| Exercise | knee_flexion_left |

| Row Labels | Sum of Num of success |
|---|---|
| 1 | 0 |
| 2 | 10 |
| 3 | 10 |
| 4 | 10 |
| 5 | 10 |
| **Grand Total** | **40** |

*Figure41 - participant's number of successful repetitions (out of 10), left leg, chair angle of 180°*

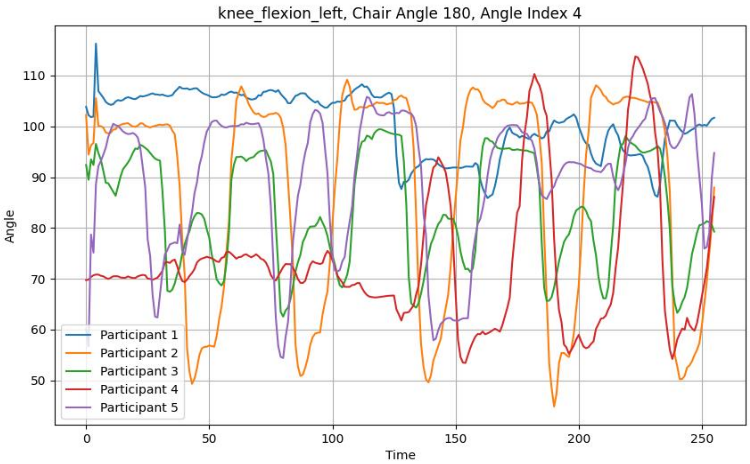



*Figure 42 - HIP, KNEE, ANKLE angle over time, left leg, chair angle of 180°*

All participants seem stable and have a constant frequency, except from participant number 1, whose graph (the blue graph) remains constant and almost does not change. From analyzing the successes of the participants in this exercise at a chair angle of 180°, we can see that participant number 1 had zero successes, compared to 100% success for the other participants in 180° chair angle. I will take a closer look on participant number 1.

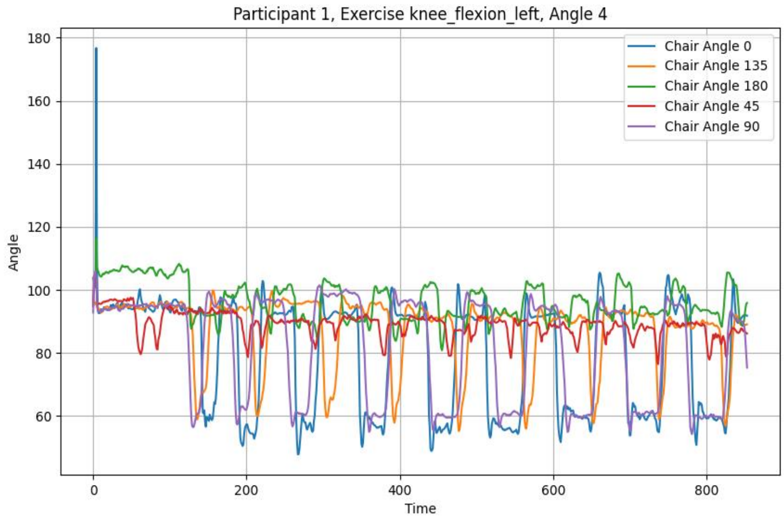



*Figure 43 - participant 1, HIP, KNEE, ANKLE angle over time, left leg, all chair angles*

If we try to analyze the green graph, which represents the 180°chair angle, we can see that the angles captured by the camera do not reach the lower range defined in the function (60°-80°). The red graph, representing the 45° chair angle, also shows very small changes that barely reach the lower range. I will analyze the same graph for participant number 2 for example, to check if the 45° and 180° angles barely manage to reach the lower range of the function for other participants as well.

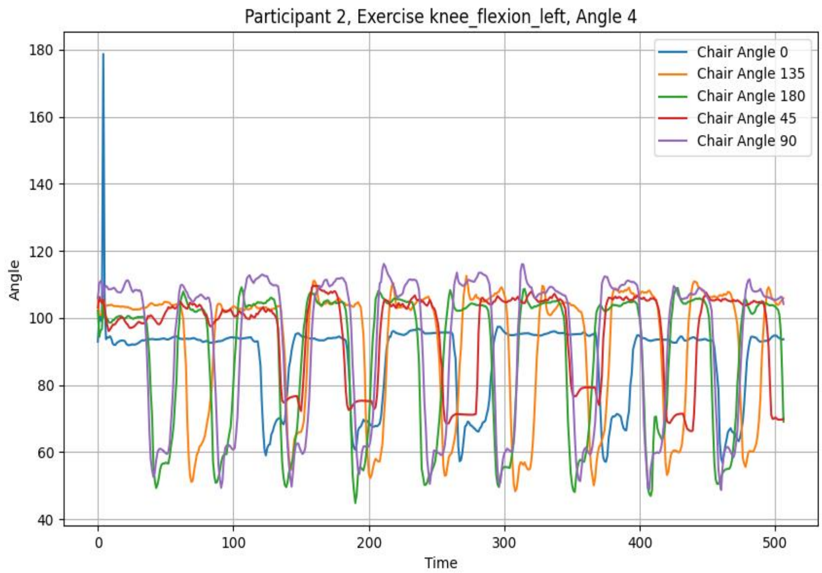



*Figure44 - participant 2, HIP, KNEE, ANKLE angle over time, left leg, all chair angles*

We can see that with participant number 2, the overall performance appears uniform. The angles are within the desired ranges and the frequencies are constant.

A similar analysis was performed on the other participants, and angles within the desired ranges and constant frequencies were found for them as well (Knee flexion).

I can infer from this that participant number 1 did not perform the exercise well at the 45° and 180° chair angles.

## 4.4 Ankle stretch

<u>Tabular analysis:</u>

In this exercise, there was a difficulty in identifying the movements of the trainees by the camera.

<u>Graphical Analysis:</u>

To investigate the reasons for the poor reception of the user's movements by the camera, I will take participant number 4 for example, who succeeded in some of the exercises and try to understand the reasons.

| Exercise | right_ankle_strech |
|---|---|
| Participant | 4 |

| Row Labels | Sum of Num of success |
|---|---|
| 0 | 0 |
| 45 | 10 |
| 90 | 0 |
| 135 | 10 |
| 180 | 7 |
| **Grand Total** | **27** |

*Figure45 - participant 4, number of successful repetitions (out of 10), right leg, all chair angles*

| Exercise | left_ankle_strech |
|---|---|
| Participant | 4 |

| Row Labels | Sum of Num of success |
|---|---|
| 0 | 1 |
| 45 | 10 |
| 90 | 3 |
| 135 | 10 |
| 180 | 1 |
| **Grand Total** | **25** |

*Figure46 - participant 4, number of successful repetitions (out of 10), left leg, all chair angles*

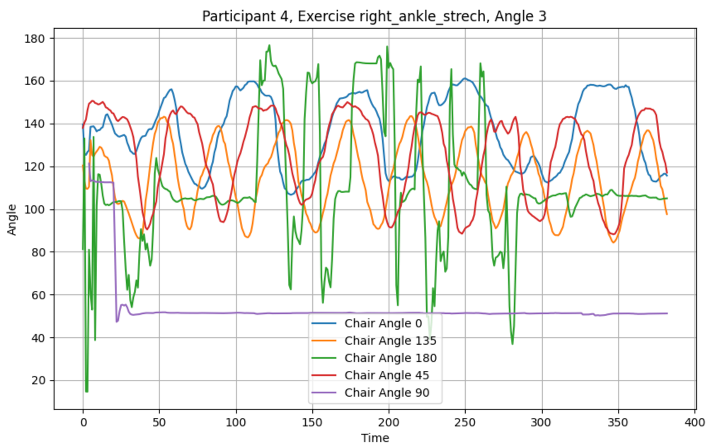



*Figure47 - participant 4, KNEE, ANKLE, BIG TOE angle over time, right leg, all chair angles*

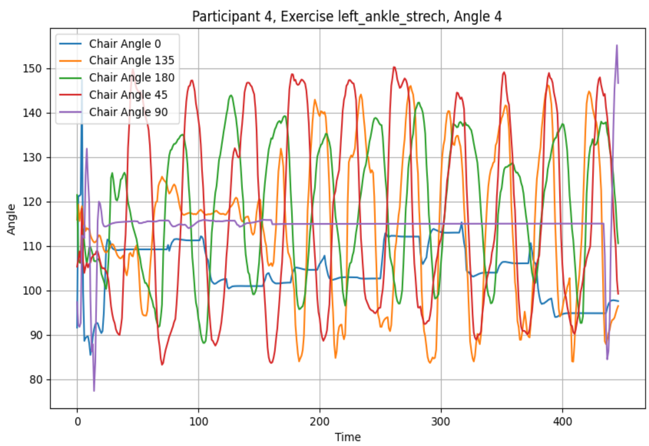



*Figure48 - participant 4, KNEE, ANKLE, BIG TOE angle over time, left leg, all chair angles*

Participant number 4 was able to correctly perform the exercise with both the right leg and the left leg at small chair angles in front of the camera (45° and 135°). The participant was unable to perform even one successful repetition with the right leg at chair angles of 0° (blue graph) and 90° (purple graph). The camera failed to pick up the angles between the user's joints at an angle of 90°, as the purple graph shows no frequency and is a constant line parallel to the x-axis.

Related results were observed with the other participants as well (Ankle stretch). For the other participants, the chair angle of 90° (purple graph) seems to have constant line parallel to the X-axis, or a little frequency that is moving in lower ranges and doesn't get to the upper range (130°-150°).

On the other hand, at a chair angle of 0°, the camera does seem to pick up a frequency. I will examine this angle later.

Participant number 4 also had a very low success rate in performing the exercise with the left leg at chair angles of 0° (blue graph), 180° (green graph), and 90° (purple graph). Here, too, we can see that the camera clearly failed to pick up the angles between the user's joints at an angle of 90°, making it impossible to perform the exercise at this angle.

I will perform a more in-depth analysis of performing the exercise at angles of 0° and 180° chair angles. According to the calling function, for an exercise to be considered successful, the angle should move from a lower range of 80°-100° to an upper range of 130°-150°.

<u>chair angle 0</u>

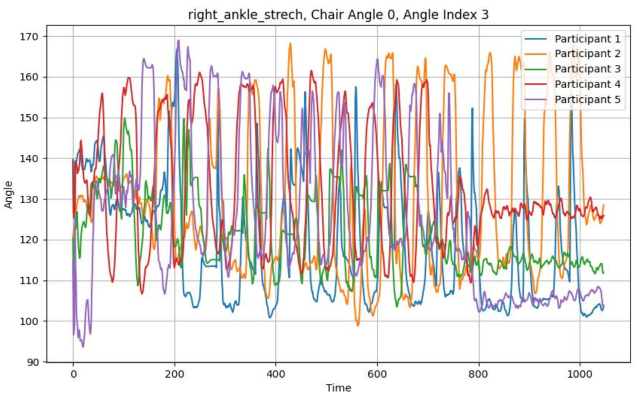



*Figure 50 - KNEE, ANKLE, BIG TOE angle over time, right leg, chair angle of 0°*

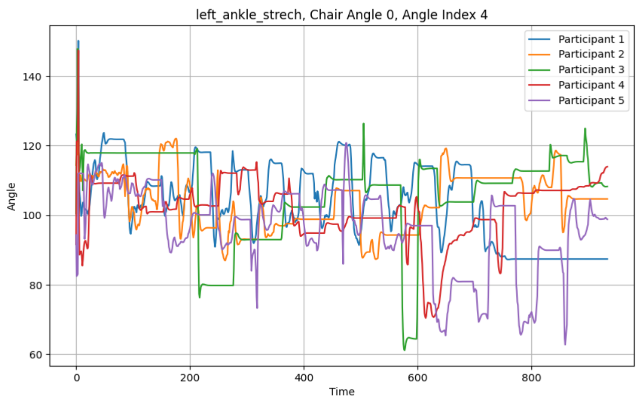



*Figure 49 - KNEE, ANKLE, BIG TOE angle over time, left leg, chair angle of 0°*

It appears that on the right side, the camera almost never managed to capture an angle between 80°-100°.

The opposite happens with the left side. The camera fails to capture the upper range between 130°-150°.

<u>chair angle 180</u>

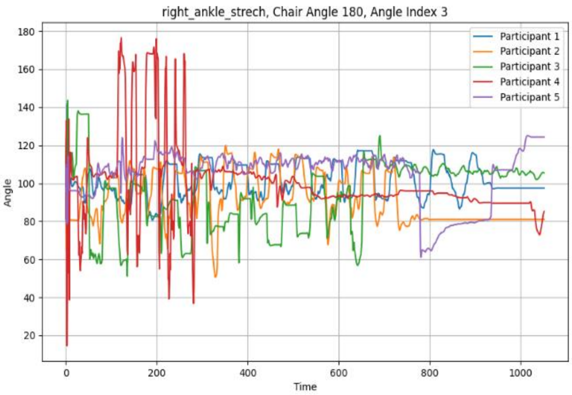



*Figure52 - KNEE, ANKLE, BIG TOE angle over time, right leg, chair angle of 180°*

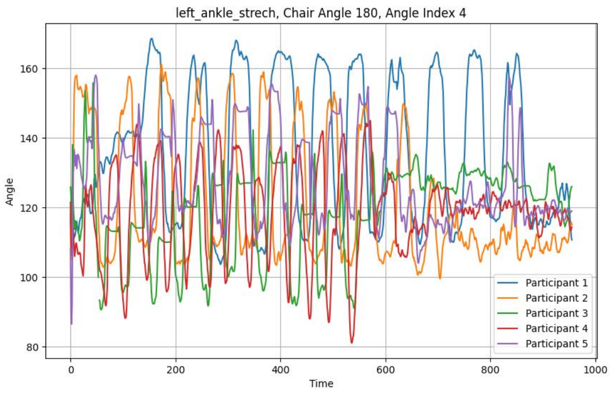



*Figure51 - KNEE, ANKLE, BIG TOE angle over time, left leg, chair angle of 180°*

We can notice that the graph of performing the exercise on the left leg when the camera is close to the working leg is very similar to the graph of performing the exercise on the right leg when the camera is close to the working leg. In both cases, the only range captured is the high range.

Additionally, we can also notice that when the exercise being performed on the right leg when the working leg is far from the camera, all graphs are relatively static. There is no frequency, and the camera does not properly capture the movement.

## 4.5 Hip abduction

<u>Tabular analysis:</u>

- The camera captures the user's movements most effectively when the trainee is positioned in front of the camera at a chair angle of 90°.
- The camera captures the user's movements in 78% success rate when the trainee is positioned at a small chair angle in front of the camera (135°), and in 74% success rate when the trainee is positioned at the other small chair angle of 45°.
- The camera captures the user's movements with very low success rate when the trainee is positioned in a vertical chair angle (0° and 180°).

<u>Graphical Analysis:</u>

The graphs of the exercises were difficult to analyze after performing them alternately. Each repetition included two movements, one for each leg, and no unusual findings were apparent.

## 4.6 Statistical Analysis

In this section, I will determine the best chair angle for performing the entire training session, rather than evaluating each exercise separately. The goal is to identify the angle at which the camera can most effectively capture the participant's movements throughout the training.

I have a database containing records, where each record represents the number of successes achieved by a specific user at a certain chair angle for a particular exercise (Success DB).

I will first use the Anona test, to get a clear picture of how the "Num of Success" differ across all chair angles.

Step 1: Checking assumptions

- **Independence** - The observations in each group are independent of each other, and the observations within groups were obtained through a random sampling process. This assumption is valid because each participant entered the experiment room alone, performed the training independently and performed the exercises in a random order. Participants were also given a 10-minute break between each training session to ensure consistency and reduce fatigue.
- **Normality** - Each sample was drawn from a normally distributed population.
  I will use the Shapiro-Wilk test to check the normality assumption in a significance level of $\alpha = 5\%$.
  Assumptions
  **$H_0$:** The data is normally distributed.
  **$H_1$:** The data is not normally distributed.
  Results
  - **$p > 0.05$**: Suggests that the data can be considered normally distributed.
  - **$p \leq 0.05$**: Suggests that the data can't be considered normally distributed.

Based on the test results, the data for chair angle of 135° fails the normality test ($p = 0.0125$), which suggests that the data for this angle is not normally distributed. The other angles (0°, 45°, 90°, 180°) passed the normality test, so their data can be considered normally distributed.

Since 4 out of 5 angles are normally distributed and there is a small sample size, I can assume normality for the 135° chair angle and proceed with the ANOVA test.

```
Angle: 0°
  W-Statistic: 0.8327
  p-Value: 0.1458

Angle: 45°
  W-Statistic: 0.9400
  p-Value: 0.6658

Angle: 90°
  W-Statistic: 0.8896
  p-Value: 0.3552

Angle: 135°
  W-Statistic: 0.7110
  p-Value: 0.0125

Angle: 180°
  W-Statistic: 0.9241
  p-Value: 0.5566
```

*Figure 53 – Shapiro-Wilk test output*

- **Equal Variances** – The variances of the populations that the samples come from are equal.

  I will use the Levene's Test to check this assumption in a significance level of $\alpha = 5\%$.

  (I won't use F-test since I need to check more than 2 groups).

  <u>Assumptions</u>

  **$H_0$:** Variances are equal across groups.

  **$H_1$:** Variances are not equal across groups.

  <u>Results</u>

  - **$p > 0.05$**: Suggests that the variances are equal across the angles (homogeneity of variances).
  - **$p \le 0.05$**: Suggests that the variances are significantly different (heterogeneity of variances).

  Since the p-value is greater than 0.05 ($p = 0.4313$), we fail to reject the null hypothesis. This suggests that the variances across the different chair angles are not significantly different, meaning that the assumption of equal variances holds.

```
Levene's Test for Equal Variances:
Levene Statistic: 0.9987
p-Value: 0.4313
Equal variances can be assumed.
```

*Figure 54 - Levene's test output*

<u>Step 2: Performing ANOVA</u>

I will perform the ANOVA test in a significance level of $\alpha = 5\%$.

<u>Assumptions</u>

**$H_0$:** $\mu_0=\mu_{45}=\mu_{90}=\mu_{135}=\mu_{180}$

**$H_1$:** At least one $\mu_i$ is different from the others.

According to **$H_0$**, the mean number of successful repetitions is the same across all chair angles.

According to **$H_1$**, at least one chair angle has a mean number of successful repetitions that is different from the others.

**DF Between** = degrees of freedom between groups = number of groups – 1 = 5-1 = 4

**DF Within** = degrees of freedom within groups = (number of groups − 1) × Num **DF Between** = (5-1) X 4 = 16

Results

- **p > 0.05**: We fail to reject the null hypothesis and suggest that any observed differences are likely due to random chance.
- **p ≤ 0.05**: We reject the null hypothesis and suggest that there is a statistically significant difference between the group means.

The significant p-value ($p < 0.0001$) suggests that the chair angle has a significant impact on the success rate of the exercises. This means that the angle at which the chair is positioned affects how successfully the camera captures the participants performances of the exercises.

```
Repeated Measures ANOVA Results:
               Anova
===================================
      F Value Num DF  Den DF Pr > F
-----------------------------------
Angle 25.0838 4.0000 16.0000 0.0000
===================================
```

*Figure55 - Anova test output*

High F-statistic (25.0838) indicates that the variability between the group means (the success rates at different chair angles) is significantly greater than the variability within the groups (the success rates for different participants within the same angle). That suggests that there are meaningful differences in success rates across the different chair angles, meaning that the chair angle has a significant impact on the camera's ability to accurately capture movements.

Step 3: Performing post-hoc analysis

I will use the post-hoc analysis to determine which specific groups (chair angles) are significantly different from each other. While ANOVA showed us that there is a significant difference among the groups, it does not specify where those differences lie. Post-hoc tests help identify these specific differences.

The corrected p-value is an adjusted p-value that accounts for multiple comparisons. In the post-hoc analysis I will perform multiple pairwise tests. In this kind of test the risk of Type I error (false positives) increases, so the p-values are adjusted to control this risk.

One common method to correct p-values is the Bonferroni correction, where the original p-value is multiplied by the number of comparisons being made.

I will do the post-hoc analysis with performing multiple pairwise tests (for all pairs of chair angles) with Bonferroni correction in a significance level of $\alpha = 5\%$.

Assumptions for each Pairwise Comparison

**$H_0$:** $\mu_{Angle1} = \mu_{Angle2}$

**$H_1$:** $\mu_{Angle1} \neq \mu_{Angle2}$

Results

- **corrected p-value < 0.05** - Significant differences.
- **corrected p-value ≥ 0.05** - non-significant differences.
- A **larger absolute value** of the T-statistic indicates a larger difference between the groups relative to the variability in the data.
- A **positive T-statistic** means the first group has a higher mean than the second group, while a **negative T-statistic** means the opposite.

The Significant differences we can see in the results are:

1. **0° vs 45°**: The significant negative T-statistic (-11.4264) and corrected p-value (0.0033) indicate that the success rates at 0° are significantly lower than at 45°.
2. **0° vs 135°**: Similarly, the success rates at 0° are significantly lower than at 135° (T-statistic = -10.7813, corrected p-value = 0.0042).
3. **45° vs 90°**: The positive T-statistic (7.8812) and corrected p-value (0.0140) suggest that success rates at 45° are significantly higher than at 90°.
4. **90° vs 135°**: The significant negative T-statistic (-10.9962) and corrected p-value (0.0039) indicate that the success rates at 90° are significantly lower than at 135°.

The non-significant differences we can see in the results are:

The comparisons between some angle pairs (e.g., 0° vs 90°, 0° vs 180°, 45° vs 135°) did not show statistically significant differences after applying the Bonferroni correction. This suggests that the success rates at these angles are not significantly different from each other.

| | Angle 1 | Angle 2 | T-Statistic | P-Value | P-Value Corrected |
|---|---|---|---|---|---|
| 0 | 0 | 45 | -11.426421 | 0.000335 | 0.003347 |
| 1 | 0 | 90 | 1.620281 | 0.180487 | 1.804873 |
| 2 | 0 | 135 | -10.781273 | 0.000420 | 0.004197 |
| 3 | 0 | 180 | -0.306386 | 0.774597 | 7.745966 |
| 4 | 45 | 90 | 7.881160 | 0.001401 | 0.014014 |
| 5 | 45 | 135 | -2.303474 | 0.082623 | 0.826227 |
| 6 | 45 | 180 | 3.520632 | 0.024436 | 0.244361 |
| 7 | 90 | 135 | -10.996173 | 0.000389 | 0.003887 |
| 8 | 90 | 180 | -1.521058 | 0.202890 | 2.028897 |
| 9 | 135 | 180 | 4.225429 | 0.013419 | 0.134191 |

*Figure56 - post-hoc test output*

# 5 Discussion

- ✓ For the knee extension exercise, the camera captures the user's movements most effectively when the trainee is positioned at chair angles of 45°, 135°, 0°, and 180°.
- ✓ For the knee extension exercise, the camera captures the movement of the working leg at all these angles, regardless of whether the leg is close to or far from the camera.
- ✓ The knee extension cannot be performed correctly at a chair angle of 90°. This suggests that the camera cannot accurately capture the trainee's movements at this angle. When the exercise is performed correctly with the trainee sitting in front of the camera and lifting their leg, the foot in the high range of the movement (150°-180°) obscures the calculated angle. This could explain why only the angles in the lower range (80°-125°) were visible.
- ✓ For the hip marching exercise, there were no significant differences between the various chair angles. This indicates that the exercise can be effectively performed and captured at any chair angle.
- ✓ For the knee flexion exercise, there were no significant differences between the various chair angles. However, since the range of motion is small, the trainee must perform the exercise correctly for the camera to capture the movements accurately. This suggests that the exercise can be performed and perceived effectively at any chair angle, depending on the trainee's range of motion.
- ✓ For the ankle stretch exercise, the camera captures the user's movements most effectively when the trainee is positioned at chair angles of 45° and 135°.
- ✓ The ankle stretch cannot be performed correctly at a chair angle of 90°. This suggests that the camera cannot accurately capture the trainee's movements at this angle. When the exercise is performed correctly with the trainee sitting in front of the camera and lifting their leg, the foot obscures the calculated angle.
- ✓ The ankle stretch cannot be performed effectively at chair angles of 0° and 180°, which are vertical to the camera. When the working leg is vertical and close to the camera, the camera perceives the angles as high. Conversely, when the working leg is vertical and far from the camera, the camera captures the movement at low angles and within a very small range.
- ✓ For the hip abduction exercise, the camera captures the user's movements most effectively when the trainee is positioned Infront of the camera at a chair angle of 90°.

- ✓ The hip abduction exercise cannot be performed correctly when the trainee is position vertically to the camera, at a chair angle of 0° or 180°. I hypothesize that because the exercise is performed with both legs alternately, and each repetition involves two movements (one for each leg), when the trainee is positioned vertical to the camera, it may not accurately capture the angle of the leg farthest from it. As a result, the camera likely recorded only the movements of the leg closest to it as successful.
- ✓ Although I believe the most effective chair angle for the hip abduction exercise is when the trainee is positioned directly in front of the camera (90°), since this position ensures the angle that the camera needs to capture is not obstructed, an additional experiment should be conducted. The exercise should be performed on each leg separately to obtain a definitive result for the optimal chair angle.
- ✓ According to ANOVA test there are meaningful differences in success rates across the different chair angles. This means that the angle at which the chair is positioned affects how successfully the camera captures the participants performances of the exercises.
- ✓ According to post-hoc test, there are several significant differences between chair angles.
- ✓ According to post-hoc test, the success rates at chair angles of 0° and 90° are significantly lower than at 45° and 135°. No significant difference was found for a chair angle of 180°. I can infer from this that if we need to choose one chair angle to do the whole training, the preferred one will be 45° or 135°. From the previous analysis of each exercise separately, I conclude that these two angles can be used, regardless of whether the working leg is close to or far from the camera. This conclusion is based on the observation that when the trainee sits at a small angle in front of the camera, the camera effectively identifies all the joints, and no angles are obscured.

# 6 Conclusions

In this project, I developed a humanoid social robot (HSR) designed to motivate elderly individuals (65+) to perform lower body exercises. The robot demonstrates the exercises, provides feedback, and monitors the trainee's movements using a ZED-2i camera. The exercises focus on strengthening the knees, hips, and ankles, all performed in a seated position to ensure safety. Due to the unavailability of the physical robot, I developed and tested the system using the V-REP simulator.

To assess the ability of the camera to accurately capture the trainee’s performances, I conducted an experiment to evaluate how the camera captures the user's movements when the trainee is seated at five different angles. When I analyzed each exercise separately, I found that the effective chair angles for each exercise varied. However, in all of them, except for hip abduction, the 45° and 135° angles were effective. The hip abduction exercise was challenging to analyze both graphically and in tables, so I recommend making an additional prediction for the effective chair angle for performing this exercise on each leg separately. When determining the optimal angle for performing the entire training, regardless of the specific exercises, the 45° and 135° angles were found to be the most suitable.

# 7 Recommendations for Future Research

- ✓ **Extended Participant Demographics** - I recommend including a wider age range and participants with varying levels of physical abilities to assess the system's effectiveness across different user groups.
- ✓ **Real-World Testing** - I suggest conducting trials in real-world environments, such as senior care facilities, to evaluate the system's performance outside of controlled lab settings.
- ✓ **Improvement of the Robotic System** - Enhancing the robot’s stability to enable exercises that require more dynamic movements would improve the system.
- ✓ **Long-Term Usage Studies** - Investigating the long-term effects of using the robotic trainer on elderly individuals’ physical health and exercise adherence would provide valuable insights.

- ✓ **User Feedback Integration** - Implementing a system for real-time user feedback to further tailor and improve the exercise programs and robotic interactions based on user preferences and difficulties would be beneficial.

By addressing these recommendations, future research can build on the foundation laid by this project, enhancing the effectiveness and applicability of robotic trainers for elderly exercise programs.

# References


[1] S. Strohkorb, C.-M. Huang, A. Ramachandran, and B. Scassellati, “Establishing Sustained, Supportive Human-Robot Relationships: Building Blocks and Open Challenges.”

[2] R. Aylett *et al.*, “Do i remember you? Memory and identity in multiple embodiments,” in *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*, 2013, pp. 143–148. doi: 10.1109/ROMAN.2013.6628435.

[3] J. Fox and A. Gambino, “Relationship Development with Humanoid Social Robots: Applying Interpersonal Theories to Human-Robot Interaction,” *Cyberpsychol Behav Soc Netw*, vol. 24, no. 5, pp. 294–299, May 2021, doi: 10.1089/cyber.2020.0181.

[4] E. S. Cross, R. Hortensius, and A. Wykowska, “From social brains to social robots,” *Philosophical Transactions: Biological Sciences*, vol. 374, 2019, doi: 10.2307/26643667.

[5] C. Piezzo and K. Suzuki, “Feasibility study of a socially assistive humanoid robot for Guiding elderly individuals during walking,” *Future Internet*, vol. 9, no. 3, Jul. 2017, doi: 10.3390/fi9030030.

[6] D. E. R. Warburton, C. W. Nicol, and S. S. D. Bredin, “Health benefits of physical activity: The evidence,” Mar. 14, 2006. doi: 10.1503/cmaj.051351.

[7] “Physical activity.” Accessed: Aug. 16, 2024. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/physical-activity

[8] “Physical activity.” Accessed: Aug. 16, 2024. [Online]. Available: https://www.who.int/westernpacific/health-topics/physical-activity

[9] S. Colcombe and A. F. Kramer, “FITNESS EFFECTS ON THE COGNITIVE FUNCTION OF OLDER ADULTS: A Meta-Analytic Study,” 2003.

[10] V. S. Thomas and P. A. Hageman, “Can Neuromuscular Strength and Function in People With Dementia Be Rehabilitated Using Resistance-Exercise Training? Results From a Preliminary Intervention Study,” 2003. [Online]. Available: https://academic.oup.com/biomedgerontology/article/58/8/M746/614908

[11] A. Lotfi, C. Langensiepen, and S. Yahaya, “Socially Assistive Robotics: Robot Exercise Trainer for Older Adults,” *Technologies (Basel)*, vol. 6, no. 1, p. 32, Mar. 2018, doi: 10.3390/technologies6010032.

[12] M. Fransen, “When is physiotherapy appropriate?,” *Best Pract Res Clin Rheumatol*, 2004.

[13] “What is physiotherapy? | The Chartered Society of Physiotherapy.” Accessed: Aug. 16, 2024. [Online]. Available: https://www.csp.org.uk/careers-jobs/what-physiotherapy

[14] “Physiotherapy and Older People - Physiopedia.” Accessed: Aug. 16, 2024. [Online]. Available: https://www.physio-pedia.com/Physiotherapy_and_Older_People

[15] T. B. Sheridan, “Human-Robot Interaction,” Jun. 01, 2016, *SAGE Publications Inc.* doi: 10.1177/0018720816644364.

[16] C. L. van Straten, J. Peter, and R. Kühne, "Child–Robot Relationship Formation: A Narrative Review of Empirical Research," *Int J Soc Robot*, vol. 12, no. 2, pp. 325–344, May 2020, doi: 10.1007/s12369-019-00569-0.

[17] M. Ghafurian, J. Hoey, and K. Dautenhahn, "Social Robots for the Care of Persons with Dementia: A Systematic Review," *ACM Trans Hum Robot Interact*, vol. 10, no. 4, Dec. 2021, doi: 10.1145/3469653.

[18] Marcel Heerink, Ben Kröse, Vanessa Evers, and Bob Wielinga, "JoPha_2_2_05," vol. 2, 2008.

[19] J. Fasola and M. Mataric, "A Socially Assistive Robot Exercise Coach for the Elderly," *J Hum Robot Interact*, vol. 2, no. 2, Jun. 2013, doi: 10.5898/jhri.2.2.fasola.

[20] D. Lopez Recio, "Designing a new model for the elderly in an assisted ITERATIVE PROGRAMMING OF THE NAO ROBOT WITHIN A MULTIDISCIPLINARY DESIGN ENVIRONMENT," 2013.

[21] B. Görer, A. A. Salah, and H. L. Akın, "An autonomous robotic exercise tutor for elderly people," *Auton Robots*, vol. 41, no. 3, pp. 657–678, Mar. 2017, doi: 10.1007/s10514-016-9598-5.

[22] D. Shi, W. Zhang, W. Zhang, and X. Ding, "A Review on Lower Limb Rehabilitation Exoskeleton Robots," Dec. 01, 2019, *Chinese Mechanical Engineering Society*. doi: 10.1186/s10033-019-0389-8.

[23] T. Eiammanussakul and V. Sangveraphunsiri, "A lower limb rehabilitation robot in sitting position with a review of training activities," *J Healthc Eng*, vol. 2018, 2018, doi: 10.1155/2018/1927807.

[24] G. Colombo, M. Joerg, ; Reinhard Schreier, and V. Dietz, "Treadmill training of paraplegic patients using a robotic orthosis."

[25] I. Díaz, J. J. Gil, and E. Sánchez, "Lower-Limb Robotic Rehabilitation: Literature Review and Challenges," *Journal of Robotics*, vol. 2011, pp. 1–11, 2011, doi: 10.1155/2011/759764.

[26] D. J. Rea, S. Schneider, and T. Kanda, "'Is this all you can do? Harder!': The effects of (Im)polite robot encouragement on exercise effort," in *ACM/IEEE International Conference on Human-Robot Interaction*, IEEE Computer Society, Mar. 2021, pp. 225–233. doi: 10.1145/3434073.3444660.

[27] P. Gadde, H. Kharrazi, H. Patel, and K. F. MacDorman, "Toward Monitoring and Increasing Exercise Adherence in Older Adults by Robotic Intervention: A Proof of Concept Study," *Journal of Robotics*, vol. 2011, pp. 1–11, 2011, doi: 10.1155/2011/438514.

[28] J. Zhou, S. Yang, and Q. Xue, "Lower limb rehabilitation exoskeleton robot: A review," 2021, *SAGE Publications Inc.* doi: 10.1177/16878140211011862.

[29] "Poppy Project - Poppy Humanoid." Accessed: Aug. 16, 2024. [Online]. Available: https://www.poppy-project.org/en/robots/poppy-humanoid/

[30] "ZED 2 - AI Stereo Camera | Stereolabs." Accessed: Aug. 16, 2024. [Online]. Available: https://www.stereolabs.com/en-il/products/zed-2

[31] “Body Tracking Overview - Stereolabs.” Accessed: Jul. 12, 2024. [Online]. Available: https://www.stereolabs.com/docs/body-tracking

[32] “Physitrack® - The world leader in remote patient engagement and Telehealth.” Accessed: Aug. 16, 2024. [Online]. Available: https://www.physitrack.com/

# Appendixes

## Classes

A. Settings

```
def __init__():

    # classes pointers
    global training
    global camera
    global robot
    global screen

    global participant_code
    global excel_workbook
    global ex_list
    global worksheet
    global worksheet1
    global worksheet2

    # training variables
    global exercise_amount
    global rep
    global req_exercise
    global finish_workout
    global waved
    global success_exercise
    global calibration
    global poppy_done
    global camera_done
    global robot_count
    global try_again # Adaptive scenario - successful performance

    # audio variables
    global audio_path

    # screen variables
    global picture_path
    global ExercisePage

    global camera_num

    # adaptation
    global adaptation_model
    global adaptive


    global performance_class
    global startPress
```

B. Main

```
import time
import Settings as s
import Excel
from Camera import Camera
from Poppy import Poppy
from Audio import Audio
from Training import Training
from Screen import Screen, FullScreenApp
```

```python
from PIL import Image, ImageTk
import pickle
import datetime


def print_hi(name):
    print(f'Hi, {name}')  # Press Ctrl+F8 to toggle the breakpoint.

if __name__ == '__main__':
    s.camera_num = 0  # 0 - webcam, 2 - second USB in maya's computer

    # Audio variables initialization
    language = 'Hebrew'
    gender = 'Female'
    s.audio_path = 'Sounds/'
    s.picture_path = 'audio files/' + language + '/' + gender + '/'
    s.str_to_say = ""
    current_time = datetime.datetime.now()
    s.participant_code = "Participant 5"

    # Training variables initialization
    s.exercise_amount = 9  # num of exercises
    s.rep = 10  # num of repetitions
    s.req_exercise = "" # current exercise
    s.finish_workout = False
    s.waved = False # hello at the beginning
    s.success_exercise = False
    s.training_done = False
    s.poppy_done = False
    s.camera_done = False
    s.robot_count = False # counting repetitions according to the trainee /
robot
    s.try_again = False

    # Excel variable
    Excel.create_workbook()  # Excel file for each training
    s.ex_list = []
    s.performance_class = []

    s.startPress = False  # training doesn't start until the user press
"start"

    # Create all components
    s.camera = Camera()
    s.training = Training()
    s.robot = Poppy()


    # Start all threads
    s.camera.start()
    s.training.start()
    s.robot.start()
    s.screen = Screen()
    s.screen.mainloop()
```

C. Poppy

```python
import threading
from pypot.creatures import PoppyHumanoid
import poppy_humanoid
```

```
import time
import Settings as s
from Audio import say


class Poppy(threading.Thread):

    def __init__(self):
        threading.Thread.__init__(self)
        self.poppy = PoppyHumanoid(simulator='vrep')
        print("ROBOT INITIALIZATION")
        for m in self.poppy.motors:  # motors need to be initialized,
False=stiff, True=loose
            m.compliant = False
        self.init_robot()

    def init_robot(self):
        for m in self.poppy.motors:
            if not m.name == 'r_elbow_y' and not m.name == 'l_elbow_y' and
not m.name == 'head_y':
                m.goto_position(0, 1, wait=True)
        time.sleep(2)


    def run(self):
        print("ROBOT START")
        while s.startPress == False: # press "start" to start the training
            time.sleep(0.00001)
        while not s.finish_workout: # as long as the training isn't over
            time.sleep(0.00000001)  # Prevents the MP to stuck
            if s.req_exercise != "" and not (s.req_exercise=="hello_waving"
and s.try_again): # if there is exercise, or hello waving
                time.sleep(1)
                print("ROBOT: Exercise ", s.req_exercise, " start")
                self.exercise_demo(s.req_exercise)  # robot shows the
exercise
                print("ROBOT: Exercise ", s.req_exercise, " done")
                while not s.waved:
                    time.sleep(0.01)  # for hello_waiving exercise, wait
until user wave
                s.req_exercise = ""
                s.poppy_done = True
        print("Robot Done")

    def exercise_demo(self, ex):  # robot performs the exercise
        if ex == "hello_waving":
            self.hello_waving()
        else:
            repetitions = s.rep if ex != "hip_abduction" else s.rep // 2
            for i in range(repetitions):  #  10 repetitions
                getattr(self, ex)(i)
                if s.robot_count:
                    print(i)
                    say(str(i+1))
                if s.success_exercise:
                    break

    def hello_waving(self):
```

```
        self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
        self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
        self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
        self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)
        time.sleep(2)
        self.poppy.r_shoulder_x.goto_position(-90, 1.5, wait=False)
        self.poppy.r_elbow_y.goto_position(-20, 1.5, wait=False)
        self.poppy.r_arm_z.goto_position(-80, 1.5, wait=False)
        for i in range(3):
            self.poppy.r_arm[3].goto_position(-35, 0.6, wait=True)
            self.poppy.r_arm[3].goto_position(35, 0.6, wait=True)
        self.finish_waving()

    def finish_waving(self):
        self.poppy.r_shoulder_x.goto_position(0, 1.5, wait=False)
        self.poppy.r_elbow_y.goto_position(90, 1.5, wait=False)
        self.poppy.r_arm_z.goto_position(0, 1.5, wait=False)


    def hip_abduction(self, i):
        time.sleep(2)
        if i==0: # sit
            self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
            self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)
            time.sleep(2)
        self.poppy.l_hip_z.goto_position(90, 1.5, wait=False)
        time.sleep(1)
        self.poppy.l_hip_z.goto_position(-20, 1.5, wait=False)
        time.sleep(1)
        self.poppy.r_hip_z.goto_position(-90, 1.5, wait=False)
        time.sleep(1)
        self.poppy.r_hip_z.goto_position(20, 1.5, wait=False)
        time.sleep(1)

    def hip_marching_right(self, i):
        time.sleep(2)
        if i==0: # sit
            self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
            self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)
            time.sleep(2)
        self.poppy.r_hip_y.goto_position(-130, 1.5, wait=False)
        time.sleep(1)
        self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)
        time.sleep(1)

    def hip_marching_left(self, i):
        time.sleep(2)
        if i==0: # sit
            self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
            self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)
            time.sleep(2)
        self.poppy.l_hip_y.goto_position(-130, 1.5, wait=False)
```

```
        time.sleep(1)
        self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
        time.sleep(1)

    def knee_extension_right (self, i):
        time.sleep(2)
        if i==0: # sit
            self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
            self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)
            time.sleep(2)
        self.poppy.r_knee_y.goto_position(0, 1.5, wait=False)
        time.sleep(1)
        self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
        time.sleep(1)

    def knee_extension_left (self, i):
        time.sleep(2)
        if i==0: # sit
            self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
            self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)
            time.sleep(2)
        self.poppy.l_knee_y.goto_position(0, 1.5, wait=False)
        time.sleep(1)
        self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
        time.sleep(1)


    def knee_flexion_right (self, i):
        time.sleep(2)
        if i==0: # sit
            self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
            self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)
            time.sleep(2)
        self.poppy.r_knee_y.goto_position(135, 1.5, wait=False)
        time.sleep(1)
        self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
        time.sleep(1)

    def knee_flexion_left(self, i):
        time.sleep(2)
        if i==0: # sit
            self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
            self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)
            time.sleep(2)
        self.poppy.l_knee_y.goto_position(135, 1.5, wait=False)
        time.sleep(1)
        self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
        time.sleep(1)
```

```
    def right_ankle_strech (self, i):
        time.sleep(2)
        if i==0: # sit
            self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
            self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
            self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)
            time.sleep(2)
            self.poppy.r_knee_y.goto_position(20, 1.5, wait=False)
            time.sleep(2)
        self.poppy.r_ankle_y.goto_position(0, 1.5, wait=False)
        time.sleep(1)
        self.poppy.r_ankle_y.goto_position(45, 1.5, wait=False)
        time.sleep(1)
        if i==s.rep-1:
            time.sleep(2)
            self.poppy.r_ankle_y.goto_position(-10, 1.5, wait=False)
            time.sleep(1)
            self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
            time.sleep(1)
            self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)


    def left_ankle_strech (self, i):
    time.sleep(2)
    if i==0: # sit
        self.poppy.r_knee_y.goto_position(90, 1.5, wait=False)
        self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
        self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
        self.poppy.r_hip_y.goto_position(-90, 1.5, wait=False)
        time.sleep(2)
        self.poppy.l_knee_y.goto_position(20, 1.5, wait=False)
        time.sleep(2)
    self.poppy.l_ankle_y.goto_position(0, 1.5, wait=False)
    time.sleep(1)
    self.poppy.l_ankle_y.goto_position(45, 1.5, wait=False)
    time.sleep(1)
    if i==s.rep-1:
        self.poppy.l_ankle_y.goto_position(-10, 1.5, wait=False)
        time.sleep(1)
        self.poppy.l_knee_y.goto_position(90, 1.5, wait=False)
        time.sleep(1)
        self.poppy.l_hip_y.goto_position(-90, 1.5, wait=False)
```

D. Training

```
import threading
import time
import Settings as s
import Excel
import random
from Screen import ExercisePage, GoodbyePage, VeryGoodPage, ExcellentPage,
WellDonePage,NextTimePage
from Screen import Screen
from Audio import say
from tkinter import Tk, Label
from PIL import Image, ImageTk
```

```
class Training(threading.Thread):
    def __init__(self):
        threading.Thread.__init__(self)

    def run(self):
        print("TRAINING START")
        while s.startPress == False:
            time.sleep(0.00001)
        say("hello_waving")
        self.run_exercise("hello_waving")
        print("Training: start waving")
        while not s.waved:  # waits for the trainee to say "hello"
            time.sleep(0.00000001)  # Prevents the MP to stuck
            continue
        time.sleep(3)
        time.sleep(2.5)
        print("Training: finish waving")
        s.poppy_done = False  # AFTER HELLO
        s.camera_done = False  # AFTER HELLO
        self.training_session()
        self.finish_workout()


    def training_session(self):
        print("Training: start exercises")
        # TODO - adding random choice of exercises.
        exercise_names_right = ["hip_marching_right",
"knee_extension_right", "knee_flexion_right", "right_ankle_strech"]
        exercise_names_left = ["hip_marching_left", "knee_extension_left",
"knee_flexion_left", "left_ankle_strech"]
        random.shuffle(exercise_names_right)  # Shuffle exercises randomly
        random.shuffle(exercise_names_left)
        time.sleep(2)
        for e in exercise_names_right:
            time.sleep(2)
            say(e)
            time.sleep(4) # wait between exercises
            self.run_exercise(e)
            while (not s.poppy_done) or (not s.camera_done):  # waits for
the robot and the camera to continue to the next exercise
                print("not done")
                time.sleep(1)
            s.poppy_done = False
            s.camera_done = False
        for e in exercise_names_left:
            time.sleep(2)
            say(e)
            time.sleep(4)  # wait between exercises
            self.run_exercise(e)
            while (not s.poppy_done) or (not s.camera_done):
                print("not done")
                time.sleep(1)
            s.poppy_done = False
            s.camera_done = False
        time.sleep(2)
        say('hip_abduction')
        s.rep = 20
        self.run_exercise("hip_abduction")
        while (not s.poppy_done) or (not s.camera_done):
            print("not done")
            time.sleep(1)
```

```
        s.poppy_done = False
        s.camera_done = False

    def finish_workout(self):
        say('goodby')
        s.screen.switch_frame(GoodbyePage)
        Excel.success_worksheet()
        Excel.close_workbook()
        s.finish_workout = True
        time.sleep(10)
        s.screen.quit()
        print("TRAINING DONE")

    def run_exercise(self, name):
        s.success_exercise = False
        s.robot_count = False
        print("TRAINING: Exercise ", name, " start")
        time.sleep(3)  # Delay the robot movement after the audio is played
        s.req_exercise = name
        s.screen.switch_frame(ExercisePage, name)
        while s.req_exercise == name :
            time.sleep(0.001)  # Prevents the MP to stuck
        if s.success_exercise:
            encourage = self.random_encouragement()
            say(encourage)
            if encourage == "well_done":
                s.screen.switch_frame(WellDonePage)
            elif encourage == "very_good":
                s.screen.switch_frame(VeryGoodPage)
            elif encourage == "excellent":
                s.screen.switch_frame(ExcellentPage)

        print("TRAINING: Exercise ", name, " done")
        time.sleep(1)

    def random_encouragement(self):
        enco = ["well_done", "very_good", "excellent"]
        return random.choice(enco)
```

E. Camera

```
import random
import pyzed.sl as sl
import threading
import socket
from Screen import TryAgainPage, WellDonePage
import Screen
from Audio import say
from Joint import Joint
from MP import MP
import Settings as s
import time
import Excel
import numpy as np
from openpyxl import Workbook


3 usages
```

```python
class Camera(threading.Thread):

    14 usages
    def calc_angle_3d(self, joint1, joint2, joint3):
        a = np.array([joint1.x, joint1.y, joint1.z], dtype=np.float32)  # First
        b = np.array([joint2.x, joint2.y, joint2.z], dtype=np.float32)  # Mid
        c = np.array([joint3.x, joint3.y, joint3.z], dtype=np.float32)  # End

        ba = a - b
        bc = c - b

        norm_ba = np.linalg.norm(ba)
        norm_bc = np.linalg.norm(bc)

        if norm_ba == 0 or norm_bc == 0:
            print("One of the vectors is of zero length, could not calculate the angle")
            return None  # or return a default value, e.g., 0 or np.nan

        cosine_angle = np.dot(ba, bc) / (norm_ba * norm_bc)

        # Clip the cosine value to the range [-1, 1] to avoid domain errors in arccos
        cosine_angle = np.clip(cosine_angle, -1.0, 1.0)

        angle = np.arccos(cosine_angle)
        return round(np.degrees(angle), 2)
```

```python
    #camera initialization
    def __init__(self):
        threading.Thread.__init__(self)
        # Initialize the ZED camera
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        self.server_address = ('localhost', 7000)
        self.sock.bind(self.server_address)
        print("CAMERA INITIALIZATION")


    def run(self):
        while True:
            print("CAMERA START")
            medaip = MP()
            medaip.start()
            self.zed = MP.get_zed(medaip)
            while not s.finish_workout:
                time.sleep(0.00000001)  # Prevents the MP to stuck
                if s.req_exercise != "":
                    ex = s.req_exercise
                    print("CAMERA: Exercise ", ex, " start")
                    getattr(self, ex)()
                    print("CAMERA: Exercise ", ex, " done")
                    s.req_exercise = ""
                    s.camera_done = True
            print("Camera Done")
```

```python
def get_skeleton_data(self):
        time.sleep(0.01)
        bodies = sl.Bodies()  # Structure containing all the detected bodies
        body_runtime_param = sl.BodyTrackingRuntimeParameters()
        body_runtime_param.detection_confidence_threshold = 40
        if self.zed.grab() == sl.ERROR_CODE.SUCCESS:
            self.zed.retrieve_bodies(bodies, body_runtime_param)
            body_array = bodies.body_list
            num_of_bodies = len(body_array)
            if num_of_bodies!=0:
                body = bodies.body_list[0]
                arr_organs = ['PELVIS', 'SPINE_1', 'SPINE_2', 'SPINE_3', 'NECK', 'NOSE', 'LEFT_EYE', 'RIGHT_EYE', 'LEFT_EAR',
                              'RIGHT_EAR', 'LEFT_CLAVICLE', 'RIGHT_CLAVICLE', 'LEFT_SHOULDER', 'RIGHT_SHOULDER', 'LEFT_ELBOW',
                              'RIGHT_ELBOW', 'LEFT_WRIST', 'RIGHT_WRIST', 'LEFT_HIP', 'RIGHT_HIP', 'LEFT_KNEE', 'RIGHT_KNEE',
                              'LEFT_ANKLE', 'RIGHT_ANKLE', 'LEFT_BIG_TOE', 'RIGHT_BIG_TOE', 'LEFT_SMALL_TOE', 'RIGHT_SMALL_TOE',
                              'LEFT_HEEL', 'RIGHT_HEEL', 'LEFT_HAND_THUMB_4', 'RIGHT_HAND_THUMB_4', 'LEFT_HAND_INDEX_1',
                              'RIGHT_HAND_INDEX_1', 'LEFT_HAND_MIDDLE_4', 'RIGHT_HAND_MIDDLE_4', 'LEFT_HAND_PINKY_1', 'RIGHT_HAND_PINKY_1']
                joints={}
                i=0
                for kp_3d in body.keypoint:
                    organ = arr_organs[i]
                    joint = Joint(organ,kp_3d)
                    joints[organ]=joint
                    i+=1
                return joints
            else:
                return None
        else:
            return None
```

```python
def exercise_two_angles_3d(self, exercise_name, joint1, joint2, joint3, up_lb, up_ub, down_lb, down_ub,
                           joint4, joint5, joint6, up_lb2, up_ub2, down_lb2, down_ub2, use_alternate_angles=False,
                           left_right_differ=False):

    flag = True
    counter = 0
    list_joints = []
    while s.req_exercise == exercise_name:
        joints = self.get_skeleton_data()
        if joints is not None:
            right_angle = self.calc_angle_3d(joints[str("RIGHT_" + joint1)], joints[str("RIGHT_" + joint2)],
                                             joints[str("RIGHT_" + joint3)])
            left_angle = self.calc_angle_3d(joints[str("LEFT_" + joint1)], joints[str("LEFT_" + joint2)],
                                            joints[str("LEFT_" + joint3)])
            if use_alternate_angles:
                right_angle2 = self.calc_angle_3d(joints[str("RIGHT_" + joint4)], joints[str("RIGHT_" + joint5)],
                                                  joints[str("LEFT_" + joint6)])
                left_angle2 = self.calc_angle_3d(joints[str("LEFT_" + joint4)], joints[str("LEFT_" + joint5)],
                                                 joints[str("RIGHT_" + joint6)])

                new_entry = [joints[str("RIGHT_" + joint1)], joints[str("RIGHT_" + joint2)],
                             joints[str("RIGHT_" + joint3)],
                             joints[str("LEFT_" + joint1)], joints[str("LEFT_" + joint2)],
                             joints[str("LEFT_" + joint3)],
                             joints[str("RIGHT_" + joint4)], joints[str("RIGHT_" + joint5)],
                             joints[str("LEFT_" + joint6)],
                             joints[str("LEFT_" + joint4)], joints[str("LEFT_" + joint5)],
                             joints[str("RIGHT_" + joint6)],
                             right_angle, left_angle, right_angle2, left_angle2]
                list_joints.append(new_entry)
```

```
        else:
            right_angle2 = self.calc_angle_3d(joints[str("RIGHT_" + joint4)], joints[str("RIGHT_" + joint5)],
                                              joints[str("RIGHT_" + joint6)])
            left_angle2 = self.calc_angle_3d(joints[str("LEFT_" + joint4)], joints[str("LEFT_" + joint5)],
                                             joints[str("LEFT_" + joint6)])

            new_entry = [joints[str("RIGHT_" + joint1)], joints[str("RIGHT_" + joint2)],
                         joints[str("RIGHT_" + joint3)],
                         joints[str("LEFT_" + joint1)], joints[str("LEFT_" + joint2)],
                         joints[str("LEFT_" + joint3)],
                         joints[str("RIGHT_" + joint4)], joints[str("RIGHT_" + joint5)],
                         joints[str("RIGHT_" + joint6)],
                         joints[str("LEFT_" + joint4)], joints[str("LEFT_" + joint5)],
                         joints[str("LEFT_" + joint6)],
                         right_angle, left_angle, right_angle2, left_angle2]
            list_joints.append(new_entry)

        ##############################################################################
        print(left_angle, " ", right_angle)
        print(left_angle2, " ", right_angle2)
        ##############################################################################

        if right_angle is not None and left_angle is not None and \
                right_angle2 is not None and left_angle2 is not None:

            if left_right_differ:
                if (up_lb < right_angle < up_ub) & (down_lb < left_angle < down_ub) & \
                        (up_lb2 < right_angle2 < up_ub2) & (down_lb2 < left_angle2 < down_ub2) & (not flag):
                    flag = True
```

```
                    counter += 1
                    print("counter:" + str(counter))
                    #  if not s.robot_count:
                    if counter > 0 and counter %2 == 0:
                        new_counter = int(counter / 2)
                        say(str(new_counter))
                elif (down_lb < right_angle < down_ub) & (up_lb < left_angle < up_ub) & \
                        (down_lb2 < right_angle2 < down_ub2) & (up_lb2 < left_angle2 < up_ub2) & (flag):
                    flag = False


            else:
                if (up_lb < right_angle < up_ub) & (up_lb < left_angle < up_ub) & \
                        (up_lb2 < right_angle2 < up_ub2) & (up_lb2 < left_angle2 < up_ub2) & (not flag):
                    flag = True
                    counter += 1
                    print("counter:" + str(counter))
                    if counter > 0 and counter % 2 == 0:
                        new_counter = int(counter/2)
                        say(str(new_counter))
                elif (down_lb < right_angle < down_ub) & (down_lb < left_angle < down_ub) & \
                        (down_lb2 < right_angle2 < down_ub2) & (down_lb2 < left_angle2 < down_ub2) & (flag):
                    flag = False

    if counter == s.rep:
        s.req_exercise = ""
        s.success_exercise = True
        break
```

```
s.ex_list.append([exercise_name, new_counter])
print("append")
Excel.wf_joints(exercise_name, list_joints)
print("joints")
```

```python
4 usages
def exercise_right_angles_one_leg(self, exercise_name, joint1, joint2, joint3, L_lb, L_ub,
                                  joint4, joint5, joint6, R_up_lb2, R_up_ub2, R_down_lb2, R_down_ub2):

        flag = True
        counter = 0
        list_joints = []
        while s.req_exercise == exercise_name:
            joints = self.get_skeleton_data()
            if joints is not None:
                right_angle = self.calc_angle_3d(joints[str("RIGHT_" + joint1)], joints[str("RIGHT_" + joint2)],
                                                 joints[str("RIGHT_" + joint3)])
                left_angle = self.calc_angle_3d(joints[str("LEFT_" + joint1)], joints[str("LEFT_" + joint2)],
                                                joints[str("LEFT_" + joint3)])
                right_angle2 = self.calc_angle_3d(joints[str("RIGHT_" + joint4)], joints[str("RIGHT_" + joint5)],
                                                  joints[str("RIGHT_" + joint6)])
                left_angle2 = self.calc_angle_3d(joints[str("LEFT_" + joint4)], joints[str("LEFT_" + joint5)],
                                                 joints[str("LEFT_" + joint6)])

                new_entry = [joints[str("RIGHT_" + joint1)], joints[str("RIGHT_" + joint2)],
                             joints[str("RIGHT_" + joint3)],
                             joints[str("LEFT_" + joint1)], joints[str("LEFT_" + joint2)],
                             joints[str("LEFT_" + joint3)],
                             joints[str("RIGHT_" + joint4)], joints[str("RIGHT_" + joint5)],
                             joints[str("RIGHT_" + joint6)],
                             joints[str("LEFT_" + joint4)], joints[str("LEFT_" + joint5)],
                             joints[str("LEFT_" + joint6)],
                             right_angle, left_angle, right_angle2, left_angle2]
                list_joints.append(new_entry)
```

```python
                ##############################################################################
                print(left_angle, " ", right_angle2)
                ##############################################################################

                if right_angle is not None and left_angle is not None and \
                        right_angle2 is not None and left_angle2 is not None:

                    if (L_lb < left_angle < L_ub) & (R_up_lb2 < right_angle2 < R_up_ub2) & (not flag):
                        flag = True
                        counter += 1
                        print("counter:" + str(counter))
                        #  if not s.robot_count:
                        say(str(counter))
                    elif (L_lb < left_angle < L_ub) & (R_down_lb2 < right_angle2 < R_down_ub2) & (flag):
                        flag = False

            if counter == s.rep:
                s.req_exercise = ""
                s.success_exercise = True
                break

        s.ex_list.append([exercise_name, counter])
        print("append")
        Excel.wf_joints(exercise_name, list_joints)
        print("joints")
```

```python
def exercise_left_angles_one_leg(self, exercise_name, joint1, joint2, joint3, R_lb, R_ub,
                                 joint4, joint5, joint6, L_up_lb2, L_up_ub2, L_down_lb2, L_down_ub2):

        flag = True
        counter = 0
        list_joints = []
        while s.req_exercise == exercise_name:
            joints = self.get_skeleton_data()
            if joints is not None:
                right_angle = self.calc_angle_3d(joints[str("RIGHT_" + joint1)], joints[str("RIGHT_" + joint2)],
                                                 joints[str("RIGHT_" + joint3)])
                left_angle = self.calc_angle_3d(joints[str("LEFT_" + joint1)], joints[str("LEFT_" + joint2)],
                                                joints[str("LEFT_" + joint3)])
                right_angle2 = self.calc_angle_3d(joints[str("RIGHT_" + joint4)], joints[str("RIGHT_" + joint5)],
                                                  joints[str("RIGHT_" + joint6)])
                left_angle2 = self.calc_angle_3d(joints[str("LEFT_" + joint4)], joints[str("LEFT_" + joint5)],
                                                 joints[str("LEFT_" + joint6)])

                new_entry = [joints[str("RIGHT_" + joint1)], joints[str("RIGHT_" + joint2)],
                             joints[str("RIGHT_" + joint3)],
                             joints[str("LEFT_" + joint1)], joints[str("LEFT_" + joint2)],
                             joints[str("LEFT_" + joint3)],
                             joints[str("RIGHT_" + joint4)], joints[str("RIGHT_" + joint5)],
                             joints[str("RIGHT_" + joint6)],
                             joints[str("LEFT_" + joint4)], joints[str("LEFT_" + joint5)],
                             joints[str("LEFT_" + joint6)],
                             right_angle, left_angle, right_angle2, left_angle2]

                list_joints.append(new_entry)
```

```python
                ##################################################################################
                print(left_angle2, " ", right_angle)
                ##################################################################################

                if right_angle is not None and left_angle is not None and \
                        right_angle2 is not None and left_angle2 is not None:

                    if (R_lb < right_angle < R_ub) & (L_up_lb2 < left_angle2 < L_up_ub2) & (not flag):
                        flag = True
                        counter += 1
                        print("counter:" + str(counter))
                        #  if not s.robot_count:
                        say(str(counter))
                    elif (R_lb < right_angle < R_ub) & (L_down_lb2 < left_angle2 < L_down_ub2) & (flag):
                        flag = False

            if counter == s.rep:
                s.req_exercise = ""
                s.success_exercise = True
                break

        s.ex_list.append([exercise_name, counter])
        print("append")
        Excel.wf_joints(exercise_name, list_joints)
        print("joints")
```

```python
def hello_waving(self):  # check if the participant waved
    if s.try_again:
        print("Camera: Wave for trying again")
        say('next_time')
    else:
        time.sleep(4)
        print("Camera: Wave for start")
        # say('ready_wave')
    while s.req_exercise == "hello_waving":
        joints = self.get_skeleton_data()
        if joints is not None:
            right_shoulder = joints[str("RIGHT_SHOULDER")]
            right_wrist = joints[str("RIGHT_WRIST")]
            if right_shoulder.y > right_wrist.y != 0:
                print(right_shoulder.y)
                print(right_wrist.y)
                s.waved = True
                say('excellent')
                print("True")
                s.req_exercise = ""
```

F. Exercise check in Camera class

```
# check exercises

#ex1
    def hip_abduction(self):
        self.exercise_two_angles_3d("hip_abduction", "SHOULDER", "HIP", "KNEE", 80, 130, 80, 130,
                                    "KNEE", "HIP", "KNEE", 40, 60, 0, 25, True, False)

#ex2

    def hip_marching_right(self):
        self.exercise_right_angles_one_leg("hip_marching_right", "HIP", "KNEE", "ANKLE", 0, 180,
                                           "SHOULDER", "HIP", "KNEE", 75, 100, 110, 135)

    def hip_marching_left(self):
        self.exercise_left_angles_one_leg("hip_marching_left", "HIP", "KNEE", "ANKLE", 0, 180,
                                          "SHOULDER", "HIP", "KNEE", 75, 100, 110, 135)

#ex3
    def knee_extension_right(self):
        self.exercise_right_angles_one_leg("knee_extension_right", "HIP", "KNEE", "ANKLE", 80, 125,
                                           "HIP", "KNEE", "ANKLE", 150, 180, 80, 125)

    def knee_extension_left(self):
        self.exercise_left_angles_one_leg("knee_extension_left", "HIP", "KNEE", "ANKLE", 80, 125,
                                          "HIP", "KNEE", "ANKLE", 150, 180, 80, 125)
```

```
#ex4
    def knee_flexion_right(self):
        self.exercise_right_angles_one_leg("knee_flexion_right", "HIP", "KNEE", "ANKLE", 80, 125,
                                           "HIP", "KNEE", "ANKLE", 60, 80, 80, 125)

    def knee_flexion_left(self):
        self.exercise_left_angles_one_leg("knee_flexion_left", "HIP", "KNEE", "ANKLE", 80, 125,
                                          "HIP", "KNEE", "ANKLE", 60, 80, 80, 125)

#ex5

    def left_ankle_strech(self):
        self.exercise_left_angles_one_leg("left_ankle_strech", "HIP", "KNEE", "ANKLE", 80, 125,
                                          "KNEE", "ANKLE", "BIG_TOE", 80, 100, 130, 150)

    def right_ankle_strech(self):
        self.exercise_right_angles_one_leg("right_ankle_strech", "HIP", "KNEE", "ANKLE", 80, 125,
                                           "KNEE", "ANKLE", "BIG_TOE", 80, 100, 130, 150)
```

G. Screen

```
# -*- coding: utf-8 -*-
import time

from Audio import say
import tkinter as tk
from PIL import Image, ImageTk
import Settings as s
import random


class Screen(tk.Tk):
    def __init__(self):
        print("screen start")
        tk.Tk.__init__(self, className='Poppy')
        self._frame = None
        self.switch_frame(EyesPage)
        say("lets_start")
        self["bg"] = "#F3FCFB"

    def switch_frame(self, frame_class, *args):
        """Destroys current frame and replaces it with a new one."""
        new_frame = frame_class(self, *args)
        if self._frame is not None:
```

```
            if hasattr(self._frame, 'background_label'):
                self._frame.background_label.destroy()
            self._frame.destroy()
        self._frame = new_frame
        self._frame.pack()

    def start_application(self):
        # Initialize and start main components here
        # For example:
        # s.training = Training()
        # s.robot = Poppy()
        # s.screen = Screen()
        # s.screen.mainloop()

        # For demonstration, switch to the StartPage after clicking start
button
        self.master.switch_frame(StartPage)


class EyesPage(tk.Frame):
    def __init__(self, master):
        super().__init__(master)
        try:
            image = Image.open('Pictures/start_screen.png')
            self.photo_image = ImageTk.PhotoImage(image)
            background_label = tk.Label(self, image=self.photo_image)
            background_label.pack()

            start_button = tk.Button(
                self,
                text="Start",
                command=lambda: master.switch_frame(StartPage),
                bg="black",   # Background color
                fg="white",   # Text color
                font=("Helvetica", 18, "bold"),  # Font style
                padx=10,  # Padding on the x-axis
                pady=5  # Padding on the y-axis
            )
            button_width = 150
            button_height = 100
            button_x = 865  # Absolute X coordinate
            button_y = 800  # Absolute Y coordinate

            start_button.place(x=button_x, y=button_y, width=button_width,
height=button_height)

        except IOError:
            tk.Label(self, text="Image not found").pack()

class ExercisePage(tk.Frame):
    def __init__(self, master, exercise_name):
        super().__init__(master)
        try:
            image = Image.open(f'Pictures/{exercise_name}.png')
            self.photo_image = ImageTk.PhotoImage(image)
            tk.Label(self, image=self.photo_image).pack()
        except IOError:
            tk.Label(self, text="Exercise image not found").pack()

class StartPage(tk.Frame):
    def __init__(self, master):
        super().__init__(master)
```

```
        try:
            s.startPress = True
            image = Image.open('Pictures/lets_start.png')
            self.photo_image = ImageTk.PhotoImage(image)
            tk.Label(self, image=self.photo_image).pack()
            #  self.after(5000, master.start_training)  # Wait for 5
seconds then start training
        except IOError:
            tk.Label(self, text="Image not found").pack()

class TryAgainPage(tk.Frame):
    def __init__(self, master):
        tk.Frame.__init__(self, master)
        image1 = Image.open('Pictures//try_again.png')
        self.photo_image1 = ImageTk.PhotoImage(image1)
        self.background_label = tk.Label(image=self.photo_image1)
        self.background_label.pack()

class VeryGoodPage(tk.Frame):
    def __init__(self, master):
        tk.Frame.__init__(self, master)
        image = Image.open('Pictures//very_good.png')
        self.photo_image = ImageTk.PhotoImage(image)
        tk.Label(self, image=self.photo_image).pack()

class ExcellentPage(tk.Frame):
    def __init__(self, master):
        tk.Frame.__init__(self, master)
        image = Image.open('Pictures//excellent.png')
        self.photo_image = ImageTk.PhotoImage(image)
        tk.Label(self, image=self.photo_image).pack()

class WellDonePage(tk.Frame):
    def __init__(self, master):
        tk.Frame.__init__(self, master)
        image = Image.open('Pictures//well_done.png')
        self.photo_image = ImageTk.PhotoImage(image)
        tk.Label(self, image=self.photo_image).pack()

class NextTimePage(tk.Frame):
    def __init__(self, master):
        tk.Frame.__init__(self, master)
        image = Image.open('Pictures//next_time.png')
        self.photo_image = ImageTk.PhotoImage(image)
        tk.Label(self, image=self.photo_image).pack()

class GoodbyePage(tk.Frame):
    def __init__(self, master):
        tk.Frame.__init__(self, master)
        image = Image.open('Pictures//goodbye.png')
        self.photo_image = ImageTk.PhotoImage(image)
        tk.Label(self, image=self.photo_image).pack()

class FullScreenApp(object):
    def __init__(self, master, **kwargs):
        self.master = master
        pad = 3
        self._geom = '200x200+0+0'
        master.geometry("{0}x{1}+0+0".format(
            master.winfo_screenwidth()-pad, master.winfo_screenheight()-
pad))
        master.bind('<Escape>', self.toggle_geom)
```

```
    def toggle_geom(self, event):
        geom=self.master.winfo_geometry()
        print(geom, self._geom)
        self.master.geometry(self._geom)
        self._geom = geom
```

H. Pictures

- Pictures
  - excellent.png
  - goodbye.png
  - hello_waving.png
  - hip_abduction.png
  - hip_marching.png
  - hip_marching_left.png
  - hip_marching_right.png
  - knee_extension.png
  - knee_extension_left.png
  - knee_extension_right.png
  - knee_flexion.png
  - knee_flexion_left.png
  - knee_flexion_right.png
  - left_ankle_strech.png
  - lets_start.png
  - next_time.png
  - right_ankle_strech.png
  - start_screen.png
  - try_again.png
  - very_good.png
  - well_done.png

I. Audio

```
import threading
import pygame
import Settings as s
import winsound
from pygame import mixer
import time


class Audio(threading.Thread):

    def __init__(self):
        threading.Thread.__init__(self)
        print ("AUDIO INITIALIZATION")

    def run(self):
        while not s.finish_workout:
            if s.str_to_say!="":
                self.say_no_wait(s.str_to_say)
                print("tts says: ", s.str_to_say)
                s.str_to_say = ""
        print ("AUDIO DONE")
```

```
    def say1(self, str_to_say):
        if (str_to_say != ""):
            winsound.PlaySound(s.audio_path+str_to_say+'.wav',
winsound.SND_FILENAME)


def say(str_to_say):
    '''
    str_to_say = the name of the file
    This function makes the robot say whatever there is in the file - play
the audio (parallelly)
    :return: audio
    '''
    mixer.init()
    try:
        mixer.music.load('Sounds/' + str_to_say + '.wav')
        mixer.music.play()
    except pygame.error as e:
        print("Error loading audio file:", e)
```

J. Sounds

```
Sounds
    1.wav
    2.wav
    3.wav
    4.wav
    5.wav
    6.wav
    7.wav
    8.wav
    9.wav
    10.wav
    excellent.wav
    goodby.wav
    hello_waving.wav
    hip_abduction.wav
    hip_marching_left.wav
    hip_marching_right.wav
    knee_extension_left.wav
    knee_extension_right.wav
    knee_flexion_left.wav
    knee_flexion_right.wav
    left_ankle_strech.wav
    lets_start.wav
    next_time.wav
    right_ankle_strech.wav
    very_good.wav
    well_done.wav
```

K. Excel

```
import xlsxwriter
import datetime
import Settings as s
from Joint import Joint


def create_workbook():
    # current_time = datetime.datetime.now()
    workbook_name = s.participant_code + ".xlsx"
    s.excel_workbook = xlsxwriter.Workbook(workbook_name)


def wf_joints(ex_name, list_joints):
    """
    Writing joints data for an exercise in Excel file in two versions
    :param ex_name:
    :param list_joints:
    :return:
    """
    current_time = datetime.datetime.now()
    name = ex_name + str(current_time.minute) + str(current_time.second)
    s.worksheet = s.excel_workbook.add_worksheet(name)
    frame_number = 1

    for l in range(1, len(list_joints)):
        row = 1
        s.worksheet.write(0, frame_number, frame_number)
        for j in list_joints[l]:
            if type(j) == Joint:
                j_ar = j.joint_to_array()
                for i in range(len(j_ar)):
                    s.worksheet.write(row, frame_number, str(j_ar[i]))
                    row += 1
            else:
                s.worksheet.write(row, frame_number, j)
                row += 1
        frame_number += 1


def success_worksheet():
    row = 1
    col = 0
    s.worksheet = s.excel_workbook.add_worksheet("success")
    for ex in s.ex_list:
        s.worksheet.write(row, col, ex[0])
        s.worksheet.write(row, col+1, ex[1])
        row += 1
        col = 0

def close_workbook():
    s.excel_workbook.close()
```

L. Joints

```
# class represent skeleton joint
import math

class Joint(object):

    def __init__(self, type, kp_3d):
        self.type = type
```

```
        if self.is_Nan(kp_3d):
            self.x = 0
            self.y = 0
            self.z = 0
            self.visible=0

        else:
            self.x = kp_3d[0]
            self.y = kp_3d[1]
            self.z = kp_3d[2]
            self.visible=1

    def __str__(self):
        return self.type+" "+str(self.x)+" "+str(self.y)+" "+str(self.z)

    def joint_to_array(self):
        arr = [self.type, self.x, self.y, self.z]
        return arr

    def is_joint_all_zeros(self):
        if self.x == 0 and self.y == 0 and self.z == 0:
            return True
        else:
            return False


    def is_Nan(self, point):
        value1 = float(point[0])
        value2 = float(point[1])
        value3 = float(point[2])

        if math.isnan(value1) and math.isnan(value2) and
math.isnan(value3):
            return True

        return False
```

M. MP

```
import pyzed.sl as sl
import threading
import cv2
import Settings as s
import sys


class MP(threading.Thread):

    def __init__(self):
        threading.Thread.__init__(self)
        print("MP INITIALIZATION")
        self.zed = sl.Camera()

    def run(self):
        print("MP START")

        # Set configuration parameters
        init = sl.InitParameters()
        init.camera_resolution = sl.RESOLUTION.HD720
        init.coordinate_system = sl.COORDINATE_SYSTEM.IMAGE
        init.depth_mode = sl.DEPTH_MODE.ULTRA
        init.coordinate_units = sl.UNIT.MILLIMETER
```

```
        init.camera_fps = 60

        # Open the camera
        self.zed.close()
        err = self.zed.open(init)
        if err != sl.ERROR_CODE.SUCCESS:
            print(repr(err))
            self.zed.close()
            exit(1)

        # Define the Objects detection module parameters
        body_params = sl.BodyTrackingParameters()
        # Set runtime parameters for body tracking
        body_params.detection_model =
sl.BODY_TRACKING_MODEL.HUMAN_BODY_FAST
        body_params.enable_tracking = True
        body_params.enable_body_fitting = True
        body_params.body_format = sl.BODY_FORMAT.BODY_38

        # Set runtime parameters after opening the camera
        runtime = sl.RuntimeParameters()
        sl.RuntimeParameters.enable_fill_mode
        runtime.measure3D_reference_frame = sl.REFERENCE_FRAME.WORLD

        # if detection_parameters.enable_tracking:
        # Set positional tracking parameters
        positional_tracking_parameters = sl.PositionalTrackingParameters()
        # Enable positional tracking
        positional_tracking_parameters.set_as_static = True
        self.zed.enable_positional_tracking(positional_tracking_parameters)

        # Enable body tracking
        zed_error = self.zed.enable_body_tracking(body_params)
        if zed_error != sl.ERROR_CODE.SUCCESS:
            print("enable_body_tracking", zed_error, "\nExit program.")
            self.zed.close()
            exit(-1)

        image = sl.Mat()

        while self.zed.is_opened() and not s.finish_workout:
            if self.zed.grab(runtime) == sl.ERROR_CODE.SUCCESS:
                self.zed.retrieve_image(image, sl.VIEW.LEFT)
                # self.zed.retrieve_image(image, sl.VIEW.RIGHT)
                frame = image.get_data()
                cv2.imshow("ZED Camera with Skeleton", frame)
                # Stop MediaPipe:
                key = cv2.waitKey(10)
                if s.finish_workout or key == ord('q'):
                    s.finish_workout = True
                    break
        self.zed.close()

    def get_zed(self):
        return self.zed
```

## Tabular Analysis

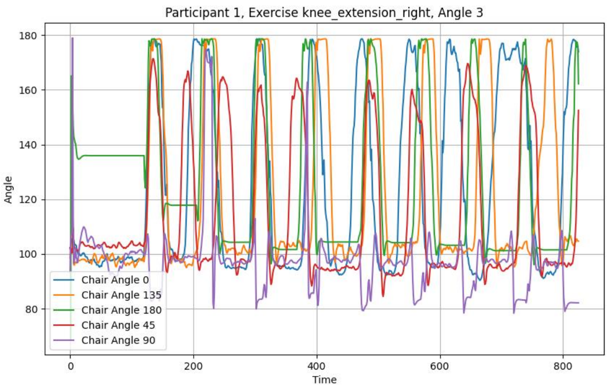
*Table 4 - The success percentages of participants at different chair angles for each exercise*

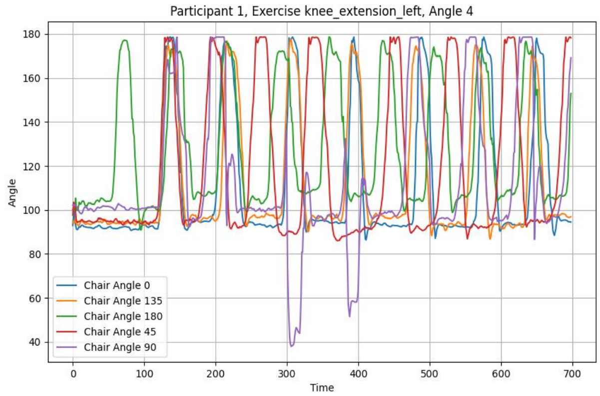
| Exercise | Sum of Num of success | Precent of success |
|---|---|---|
| **hip_abduction** | **161** | |
| 90 | 46 | 92% |
| 135 | 39 | 78% |
| 45 | 37 | 74% |
| 180 | 21 | 42% |
| 0 | 18 | 36% |
| **hip_marching_left** | **245** | |
| 180 | 50 | 100% |
| 135 | 50 | 100% |
| 45 | 50 | 100% |
| 0 | 49 | 98% |
| 90 | 46 | 92% |
| **hip_marching_right** | **248** | |
| 0 | 50 | 100% |
| 135 | 50 | 100% |
| 45 | 50 | 100% |
| 180 | 49 | 98% |
| 90 | 49 | 98% |
| **knee_extension_left** | **225** | |
| 135 | 50 | 100% |
| 0 | 50 | 100% |
| 180 | 50 | 100% |
| 45 | 50 | 100% |
| 90 | 25 | 50% |
| **knee_extension_right** | **221** | |
| 135 | 50 | 100% |
| 0 | 50 | 100% |
| 180 | 50 | 100% |
| 45 | 50 | 100% |
| 90 | 21 | 42% |
| **knee_flexion_left** | **236** | |
| 0 | 50 | 100% |
| 135 | 50 | 100% |
| 90 | 50 | 100% |
| 45 | 46 | 92% |
| 180 | 40 | 80% |
| **knee_flexion_right** | **250** | |
| 135 | 50 | 100% |
| 0 | 50 | 100% |
| 180 | 50 | 100% |
| 45 | 50 | 100% |
| 90 | 50 | 100% |

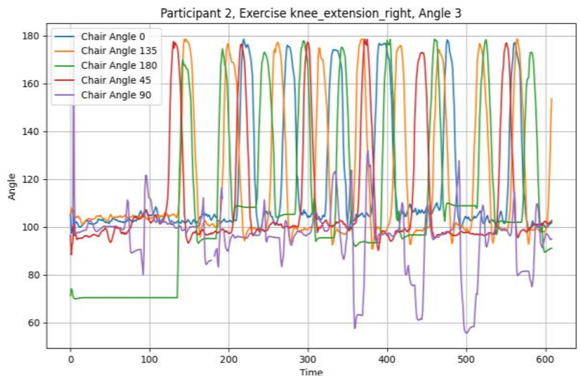
| | | |
|---|---|---|
| **left_ankle_strech** | **87** | |
| 135 | 44 | 88% |
| 45 | 28 | 56% |
| 180 | 7 | 14% |
| 90 | 5 | 10% |
| 0 | 3 | 6% |
| **right_ankle_strech** | **103** | |
| 135 | 48 | 96% |
| 45 | 41 | 82% |
| 180 | 9 | 18% |
| 90 | 5 | 10% |
| 0 | 0 | 0% |

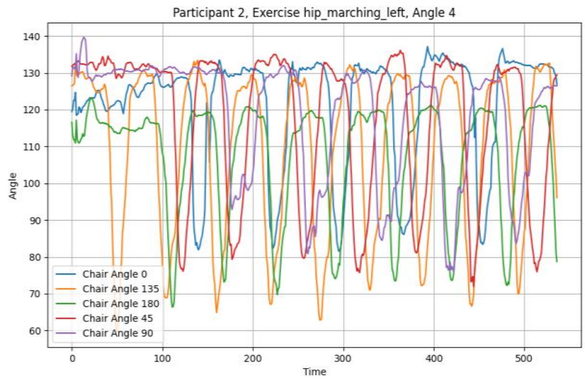
# Graphical Analysis

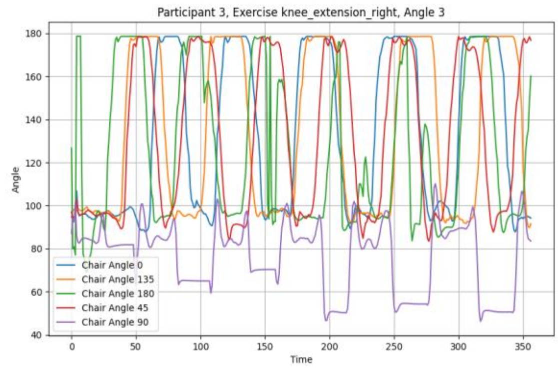
## N. Knee extension

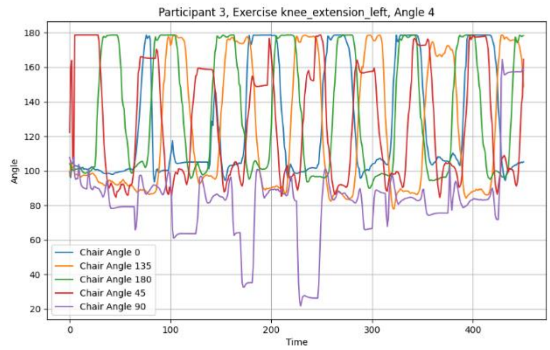


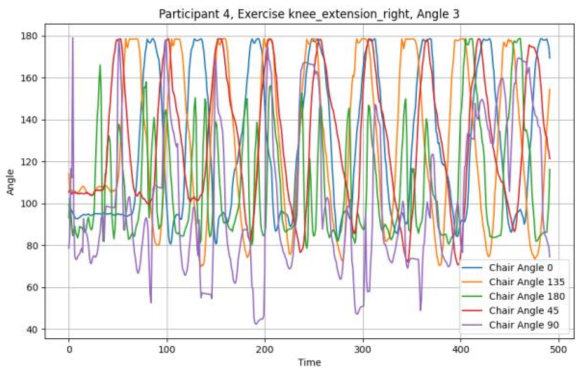



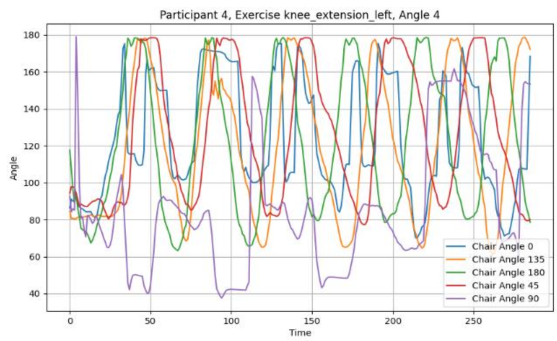



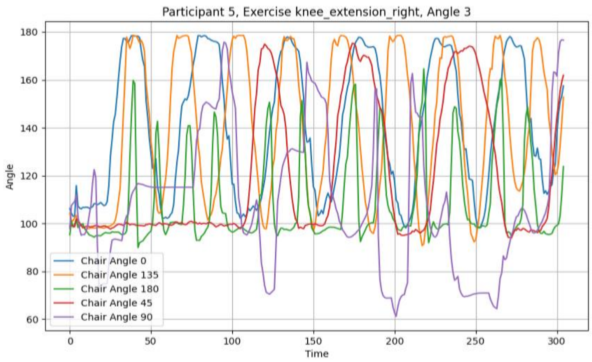



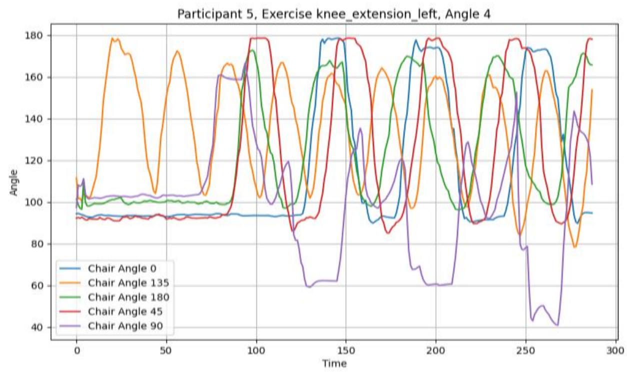



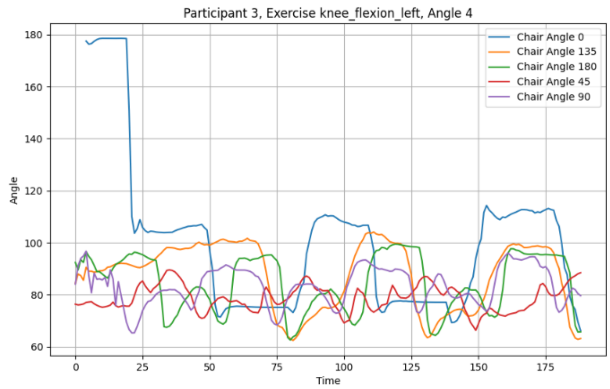



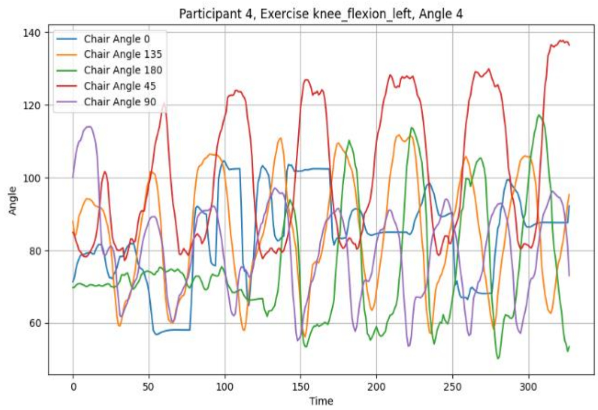


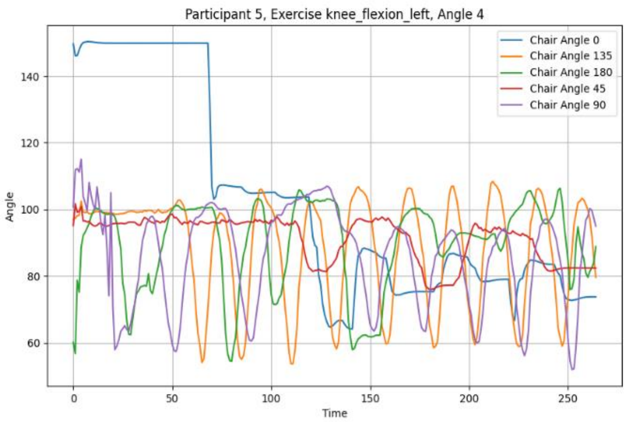




## O. Knee flexion

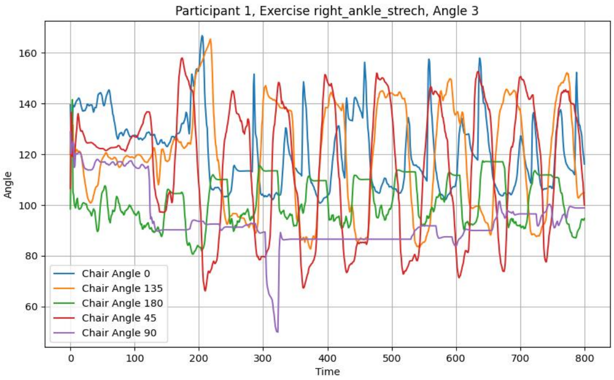


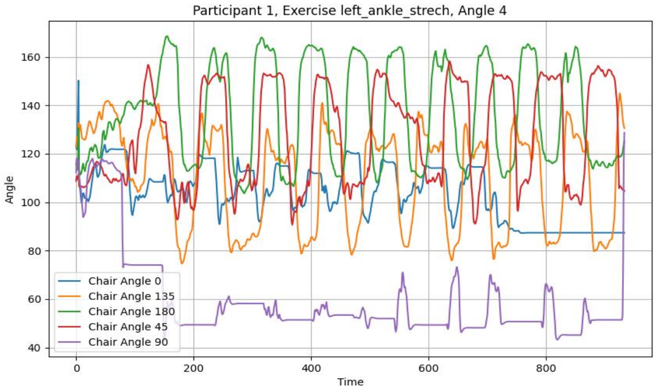


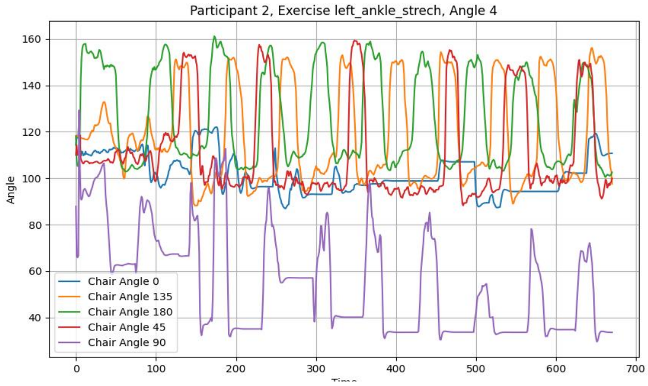



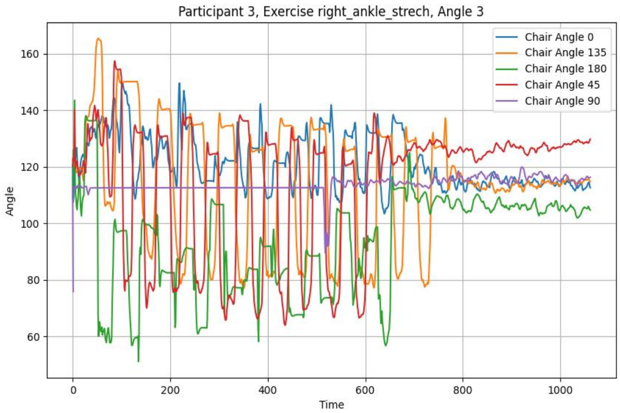
## P. Ankle stretch

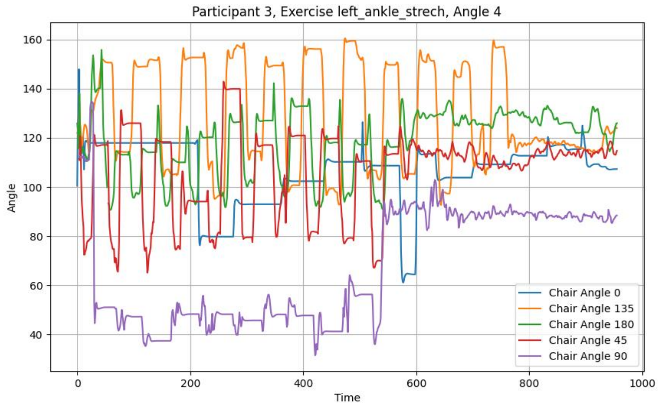


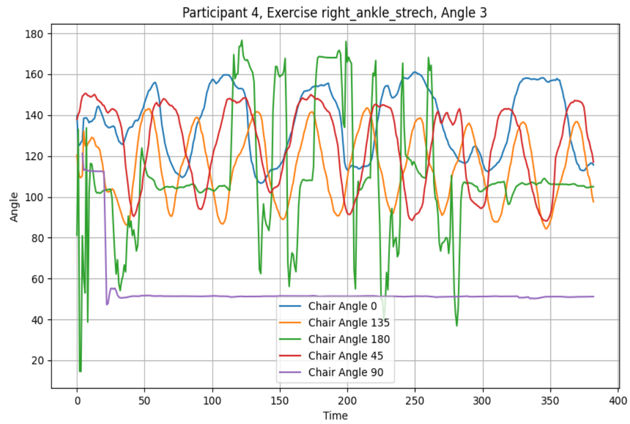



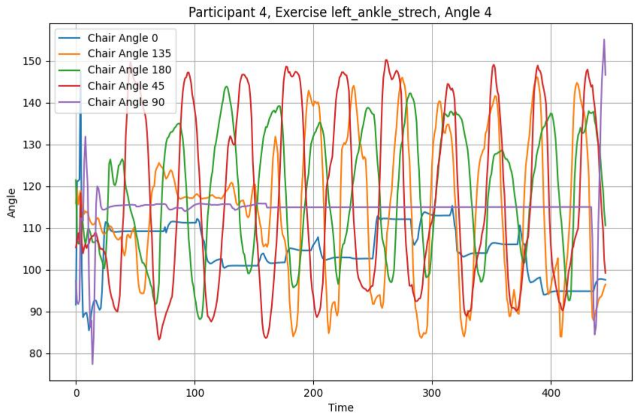


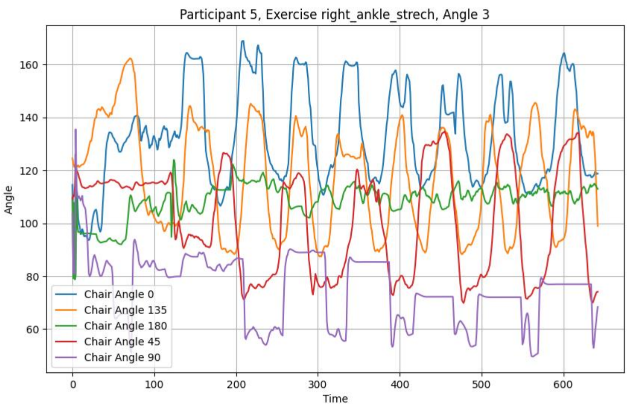


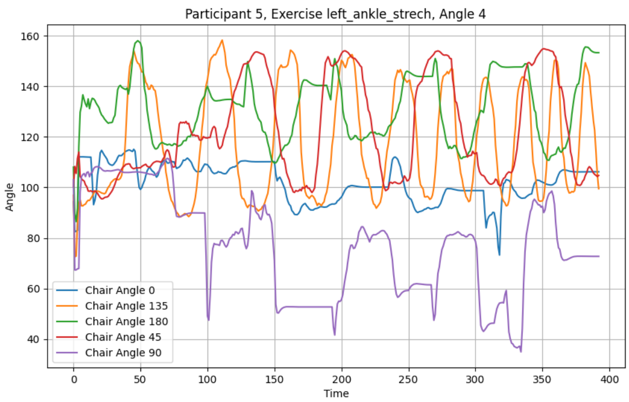






# Statistical Analysis

## Q. Success DB

*__Table 5__ - The successful repetitions of a specific participant, exercise and chair angle*

| Participant | Chair Angle | Exercise | Num of success |
|---|---|---|---|
| 1 | 0 | right_ankle_strech | 0 |
| 1 | 0 | hip_marching_right | 10 |
| 1 | 0 | knee_extension_right | 10 |
| 1 | 0 | knee_flexion_right | 10 |
| 1 | 0 | knee_extension_left | 10 |
| 1 | 0 | knee_flexion_left | 10 |
| 1 | 0 | hip_marching_left | 10 |
| 1 | 0 | left_ankle_strech | 1 |
| 1 | 0 | hip_abduction | 4 |
| 1 | 45 | knee_flexion_right | 10 |
| 1 | 45 | hip_marching_right | 10 |
| 1 | 45 | right_ankle_strech | 10 |
| 1 | 45 | knee_extension_right | 10 |
| 1 | 45 | left_ankle_strech | 6 |
| 1 | 45 | knee_flexion_left | 9 |
| 1 | 45 | hip_marching_left | 10 |
| 1 | 45 | knee_extension_left | 10 |
| 1 | 45 | hip_abduction | 10 |
| 1 | 90 | right_ankle_strech | 0 |
| 1 | 90 | knee_extension_right | 2 |
| 1 | 90 | knee_flexion_right | 10 |
| 1 | 90 | hip_marching_right | 9 |
| 1 | 90 | knee_extension_left | 8 |
| 1 | 90 | hip_marching_left | 10 |
| 1 | 90 | knee_flexion_left | 10 |
| 1 | 90 | left_ankle_strech | 0 |
| 1 | 90 | hip_abduction | 10 |

| | | | |
|---|---|---|---|
| 1 | 135 | knee_flexion_right | 10 |
| 1 | 135 | right_ankle_strech | 9 |
| 1 | 135 | knee_extension_right | 10 |
| 1 | 135 | hip_marching_right | 10 |
| 1 | 135 | left_ankle_strech | 10 |
| 1 | 135 | knee_flexion_left | 10 |
| 1 | 135 | knee_extension_left | 10 |
| 1 | 135 | hip_marching_left | 10 |
| 1 | 135 | hip_abduction | 10 |
| 1 | 180 | knee_extension_right | 10 |
| 1 | 180 | right_ankle_strech | 1 |
| 1 | 180 | hip_marching_right | 9 |
| 1 | 180 | knee_flexion_right | 10 |
| 1 | 180 | knee_flexion_left | 0 |
| 1 | 180 | knee_extension_left | 10 |
| 1 | 180 | left_ankle_strech | 0 |
| 1 | 180 | hip_marching_left | 10 |
| 1 | 180 | hip_abduction | 3 |
| 2 | 0 | knee_flexion_right | 10 |
| 2 | 0 | hip_marching_right | 10 |
| 2 | 0 | right_ankle_strech | 0 |
| 2 | 0 | knee_extension_right | 10 |
| 2 | 0 | knee_extension_left | 10 |
| 2 | 0 | hip_marching_left | 10 |
| 2 | 0 | left_ankle_strech | 0 |
| 2 | 0 | knee_flexion_left | 10 |
| 2 | 0 | hip_abduction | 5 |
| 2 | 45 | hip_marching_right | 10 |
| 2 | 45 | knee_extension_right | 10 |
| 2 | 45 | right_ankle_strech | 10 |
| 2 | 45 | knee_flexion_right | 10 |
| 2 | 45 | knee_flexion_left | 10 |
| 2 | 45 | knee_extension_left | 10 |
| 2 | 45 | left_ankle_strech | 4 |
| 2 | 45 | hip_marching_left | 10 |
| 2 | 45 | hip_abduction | 6 |
| 2 | 90 | hip_marching_right | 10 |
| 2 | 90 | knee_extension_right | 3 |
| 2 | 90 | right_ankle_strech | 4 |
| 2 | 90 | knee_flexion_right | 10 |
| 2 | 90 | hip_marching_left | 10 |
| 2 | 90 | left_ankle_strech | 0 |
| 2 | 90 | knee_extension_left | 2 |
| 2 | 90 | knee_flexion_left | 10 |
| 2 | 90 | hip_abduction | 9 |
| 2 | 135 | hip_marching_right | 10 |
| 2 | 135 | knee_flexion_right | 10 |

| | | | |
|---|---|---|---|
| 2 | 135 | knee_extension_right | 10 |
| 2 | 135 | right_ankle_strech | 10 |
| 2 | 135 | knee_extension_left | 10 |
| 2 | 135 | left_ankle_strech | 10 |
| 2 | 135 | knee_flexion_left | 10 |
| 2 | 135 | hip_marching_left | 10 |
| 2 | 135 | hip_abduction | 9 |
| 2 | 180 | knee_flexion_right | 10 |
| 2 | 180 | knee_extension_right | 10 |
| 2 | 180 | right_ankle_strech | 0 |
| 2 | 180 | hip_marching_right | 10 |
| 2 | 180 | hip_marching_left | 10 |
| 2 | 180 | knee_flexion_left | 10 |
| 2 | 180 | left_ankle_strech | 1 |
| 2 | 180 | knee_extension_left | 10 |
| 2 | 180 | hip_abduction | 7 |
| 3 | 0 | right_ankle_strech | 0 |
| 3 | 0 | knee_extension_right | 10 |
| 3 | 0 | knee_flexion_right | 10 |
| 3 | 0 | hip_marching_right | 10 |
| 3 | 0 | knee_extension_left | 10 |
| 3 | 0 | hip_marching_left | 10 |
| 3 | 0 | knee_flexion_left | 10 |
| 3 | 0 | left_ankle_strech | 1 |
| 3 | 0 | hip_abduction | 3 |
| 3 | 45 | knee_extension_right | 10 |
| 3 | 45 | knee_flexion_right | 10 |
| 3 | 45 | hip_marching_right | 10 |
| 3 | 45 | right_ankle_strech | 7 |
| 3 | 45 | knee_extension_left | 10 |
| 3 | 45 | hip_marching_left | 10 |
| 3 | 45 | left_ankle_strech | 3 |
| 3 | 45 | knee_flexion_left | 10 |
| 3 | 45 | hip_abduction | 10 |
| 3 | 90 | right_ankle_strech | 0 |
| 3 | 90 | knee_extension_right | 0 |
| 3 | 90 | hip_marching_right | 10 |
| 3 | 90 | knee_flexion_right | 10 |
| 3 | 90 | hip_marching_left | 10 |
| 3 | 90 | knee_flexion_left | 10 |
| 3 | 90 | knee_extension_left | 2 |
| 3 | 90 | left_ankle_strech | 1 |
| 3 | 90 | hip_abduction | 10 |
| 3 | 135 | right_ankle_strech | 9 |
| 3 | 135 | knee_flexion_right | 10 |
| 3 | 135 | knee_extension_right | 10 |
| 3 | 135 | hip_marching_right | 10 |

| | | | |
|---|---|---|---|
| 3 | 135 | knee_extension_left | 10 |
| 3 | 135 | hip_marching_left | 10 |
| 3 | 135 | left_ankle_strech | 4 |
| 3 | 135 | knee_flexion_left | 10 |
| 3 | 135 | hip_abduction | 5 |
| 3 | 180 | right_ankle_strech | 1 |
| 3 | 180 | hip_marching_right | 10 |
| 3 | 180 | knee_extension_right | 10 |
| 3 | 180 | knee_flexion_right | 10 |
| 3 | 180 | knee_flexion_left | 10 |
| 3 | 180 | knee_extension_left | 10 |
| 3 | 180 | hip_marching_left | 10 |
| 3 | 180 | left_ankle_strech | 5 |
| 3 | 180 | hip_abduction | 4 |
| 4 | 0 | knee_extension_right | 10 |
| 4 | 0 | hip_marching_right | 10 |
| 4 | 0 | right_ankle_strech | 0 |
| 4 | 0 | knee_flexion_right | 10 |
| 4 | 0 | knee_flexion_left | 10 |
| 4 | 0 | left_ankle_strech | 1 |
| 4 | 0 | hip_marching_left | 9 |
| 4 | 0 | knee_extension_left | 10 |
| 4 | 0 | hip_abduction | 2 |
| 4 | 45 | hip_marching_right | 10 |
| 4 | 45 | knee_extension_right | 10 |
| 4 | 45 | knee_flexion_right | 10 |
| 4 | 45 | right_ankle_strech | 10 |
| 4 | 45 | hip_marching_left | 10 |
| 4 | 45 | left_ankle_strech | 10 |
| 4 | 45 | knee_flexion_left | 10 |
| 4 | 45 | knee_extension_left | 10 |
| 4 | 45 | hip_abduction | 1 |
| 4 | 90 | hip_marching_right | 10 |
| 4 | 90 | knee_flexion_right | 10 |
| 4 | 90 | right_ankle_strech | 0 |
| 4 | 90 | knee_extension_right | 7 |
| 4 | 90 | knee_flexion_left | 10 |
| 4 | 90 | knee_extension_left | 8 |
| 4 | 90 | hip_marching_left | 10 |
| 4 | 90 | left_ankle_strech | 3 |
| 4 | 90 | hip_abduction | 10 |
| 4 | 135 | right_ankle_strech | 10 |
| 4 | 135 | knee_flexion_right | 10 |
| 4 | 135 | hip_marching_right | 10 |
| 4 | 135 | knee_extension_right | 10 |
| 4 | 135 | left_ankle_strech | 10 |
| 4 | 135 | hip_marching_left | 10 |

| | | | |
|---|---|---|---|
| 4 | 135 | knee_flexion_left | 10 |
| 4 | 135 | knee_extension_left | 10 |
| 4 | 135 | hip_abduction | 6 |
| 4 | 180 | right_ankle_strech | 7 |
| 4 | 180 | hip_marching_right | 10 |
| 4 | 180 | knee_extension_right | 10 |
| 4 | 180 | knee_flexion_right | 10 |
| 4 | 180 | knee_flexion_left | 10 |
| 4 | 180 | hip_marching_left | 10 |
| 4 | 180 | left_ankle_strech | 1 |
| 4 | 180 | knee_extension_left | 10 |
| 4 | 180 | hip_abduction | 5 |
| 5 | 0 | knee_extension_right | 10 |
| 5 | 0 | hip_marching_right | 10 |
| 5 | 0 | right_ankle_strech | 0 |
| 5 | 0 | knee_flexion_right | 10 |
| 5 | 0 | left_ankle_strech | 0 |
| 5 | 0 | knee_extension_left | 10 |
| 5 | 0 | knee_flexion_left | 10 |
| 5 | 0 | hip_marching_left | 10 |
| 5 | 0 | hip_abduction | 4 |
| 5 | 45 | knee_extension_right | 10 |
| 5 | 45 | knee_flexion_right | 10 |
| 5 | 45 | right_ankle_strech | 4 |
| 5 | 45 | hip_marching_right | 10 |
| 5 | 45 | knee_flexion_left | 7 |
| 5 | 45 | left_ankle_strech | 5 |
| 5 | 45 | knee_extension_left | 10 |
| 5 | 45 | hip_marching_left | 10 |
| 5 | 45 | hip_abduction | 10 |
| 5 | 90 | hip_marching_right | 10 |
| 5 | 90 | knee_flexion_right | 10 |
| 5 | 90 | knee_extension_right | 9 |
| 5 | 90 | right_ankle_strech | 1 |
| 5 | 90 | hip_marching_left | 6 |
| 5 | 90 | knee_extension_left | 5 |
| 5 | 90 | knee_flexion_left | 10 |
| 5 | 90 | left_ankle_strech | 1 |
| 5 | 90 | hip_abduction | 7 |
| 5 | 135 | right_ankle_strech | 10 |
| 5 | 135 | knee_flexion_right | 10 |
| 5 | 135 | hip_marching_right | 10 |
| 5 | 135 | knee_extension_right | 10 |
| 5 | 135 | hip_marching_left | 10 |
| 5 | 135 | knee_flexion_left | 10 |
| 5 | 135 | knee_extension_left | 10 |
| 5 | 135 | left_ankle_strech | 10 |

| | | | |
|---|---|---|---|
| 5 | 135 | hip_abduction | 9 |
| 5 | 180 | hip_marching_right | 10 |
| 5 | 180 | knee_extension_right | 10 |
| 5 | 180 | knee_flexion_right | 10 |
| 5 | 180 | right_ankle_strech | 0 |
| 5 | 180 | knee_flexion_left | 10 |
| 5 | 180 | left_ankle_strech | 0 |
| 5 | 180 | knee_extension_left | 10 |
| 5 | 180 | hip_marching_left | 10 |
| 5 | 180 | hip_abduction | 2 |