University of West Bohemia
Faculty of Applied Sciences
Department of Computer Science and Engineering

Bachelor's thesis

Arm rehabilitation with video motion detection

Pilsen, 2020 Martin Ryba

Místo této strany bude zadání práce.

Declaration

I hereby declare that this bachelor's thesis is completely my own work and that I used only the cited sources.

Pilsen, 21st July 2020

Martin Ryba

Abstract

The bachelor thesis is focused on the data collection, annotation and model training for the purposes of a simple arm rehabilitation application. Various methods of rehablitation were chosen and described with the help of a professional physiotherapist. Based on the information, data were collected and several machine learning models were trained from which the best performing one was chosen. A simple application with chosen trained model was developed to be used for the real-time feedback during the arm rehabilitation process. Finally, this application and model were tested with several volunteers.

Abstrakt

Tato bakalářská práce je zaměřena na sběr a anotaci dat a natrénování modelu pro potřeby jednoduché rehabilitace paže. S pomocí profesionálního fyzioterapeuta byly vybrány a popsány různé metody rehabilitace. Na základě čehož byla sesbírána data, která byla použita pro natrénování několika modelů strojového učení, z nichž ten nejlepší byl vybrán pro další použití. Dále byla vyvinuta jednoduchá aplikace s vybraným modelem pro rehabilitaci paže, která poskytuje zpětnou vazbou v reálném čase. Nakonec byla aplikace otestována s několika dobrovolníky.

List of Acronyms

ADL Activities of Daily Living. 9, 19, 21

AI Artificial Intelligence. 13, 14

CBT Cognitive Behavioral Therapy. 18

CNN Convolutional Neural Network. 15

CPU Central Processing Unit. 14

CV Computer Vision. 9

DL Deep Learning. 7, 10, 13, 14, 39

DOF Degrees of Freedom. 21

FN False Negative. 29

FP False Positive. 29

GPU Graphics Processing Unit. 14

GUI Graphical User Interface. 34, 35

HPE Human Pose Estimation. 7, 9, 10, 13, 15, 16, 25, 37–39

KNN K-Nearest Neighbors. 27, 28, 32, 33, 37, 38

ML Machine Learning. 13, 14, 27, 31, 32, 39, 40

MSPN Multi-Stage Pose Network. 16

MVC Model-View-Controller. 34, 35

NN Neural Network. 14, 27, 40

PAF Part Affinity Field. 16

PMA Pneumatic Muscle Activator. 12

RNN Recurrent Neural Network. 40

SVM Support Vector Machine. 28, 32, 33, 37, 38

TN True Negative. 29

TP True Positive. 29, 30

VR Virtual Reality. 7, 10, 17, 19

Contents

1 Introduction

Stroke is a significant cause of adult disability and the third most common cause of death worldwide [1, 2]. It occurs mostly among the elderly [3], where about three-quarters of all first strokes happen after the age of 65 [4, 5].

Survivors often experience paralysis or loss of physical strength on one side of the body, as well as memory loss, significantly impairing their ability to perform Activities of Daily Living (ADL). The primary treatment for these disabilities is through rehabilitation — by relearning the correct movement patterns. It has been shown that thorough therapy and task-based exercises helps with significant recovery of motor functions [6].

It is projected that by 2050, the global population of older persons (aged 60 years or older) will increase to 2.1 billion (compared to 2015's 901 million) — decreasing the number of people in working age per older person from 7.0 to 3.5 [7, 8].

Due to the aging population, it is becoming increasingly essential to develop effective rehabilitation methods to enable physiotherapists to assist more patients. Many of the modern rehabilitation methods aim to increase the effectiveness of rehabilitation by decreasing the necessary time spent with the patient before they can perform prescribed exercises on their own.

To achieve that, various mechanisms have been developed to observe or guide the patient after the initial introduction to the exercises by medical personnel. Exoskeleton robots are currently a popular choice for this purpose, although these specialized devices do not alleviate the patient with limited movement capability of the need to visit the hospital. The visits may be needed several times a week for the first few months and in some cases even for most of the patient's life [1]. Needless to say, the exoskeleton robots are also costly, require significant space, and can handle only one person at a time, thus somewhat alleviating, but not eliminating the burden on the hospitals.

One of the current research goals is to move as much of the rehabilitation therapy as possible into patients' homes. Recent advancements in Computer Vision (CV) enabled Human Pose Estimation (HPE) with unprecedented accuracy. This thesis will examine if HPE can be efficiently used for in-home rehabilitation.

2 State of the art

This chapter will briefly summarize recent development of the approaches to rehabilitation and the role of technology in it. It will mainly focus on the technologies enabling effective home rehabilitation.

The major problem to solve when using technology for rehabilitation has been the issue of motion tracking. Visual tracking (§2.1), specifically its marker-based variant, has been successfull in general motion tracking tasks, whereas robot-aided tracking (§2.2), with the option to add resistance, has been implemented for rehabilitation with great effect.

Various other options for non-visual tracking (§2.3), such as accelerometers, are still highly popular for problems where great precision of tracking is key.

Recent advancements in Deep Learning (§2.4) allowed for highly accurate marker-free tracking systems (§2.1.2), which resulted in an entire subfield of Human Pose Estimation (HPE) (§2.4.3).

The chapter concludes with briefly touching the patient motivation (§2.5), since adhering to prescribed training program is the key to success. The physiotherapist traditionally gave feedback as a way to motivate the patient. However, in home setting it can be difficult for many people to adhere to the recommended (sometimes daily) exercises for extended periods of time. Various proposed solutions to the problem, such as usage of Virtual Reality (VR) (§2.5.1) or dialogue systems (§2.5.2), are thus described.

2.1 Visual motion tracking systems

Visual analysis of the human movement is a broad endeavour that can be divided into various categories. Some of the most important distinctions are:

- usage of one or more cameras
- relience on special marks on the subject

Since the usage of one camera is given by the problem, we will assume monocular vision in further text.

Visual-based systems are inexpensive and safe for the tracked person in case of a major malfunction. On the other hand, a major drawback of these systems is occlusion, where parts of the information are not visible from the camera's point of view either due to objects in the scene or the tracked person's own body (self-occlusion).

2.1.1 Marker-based systems

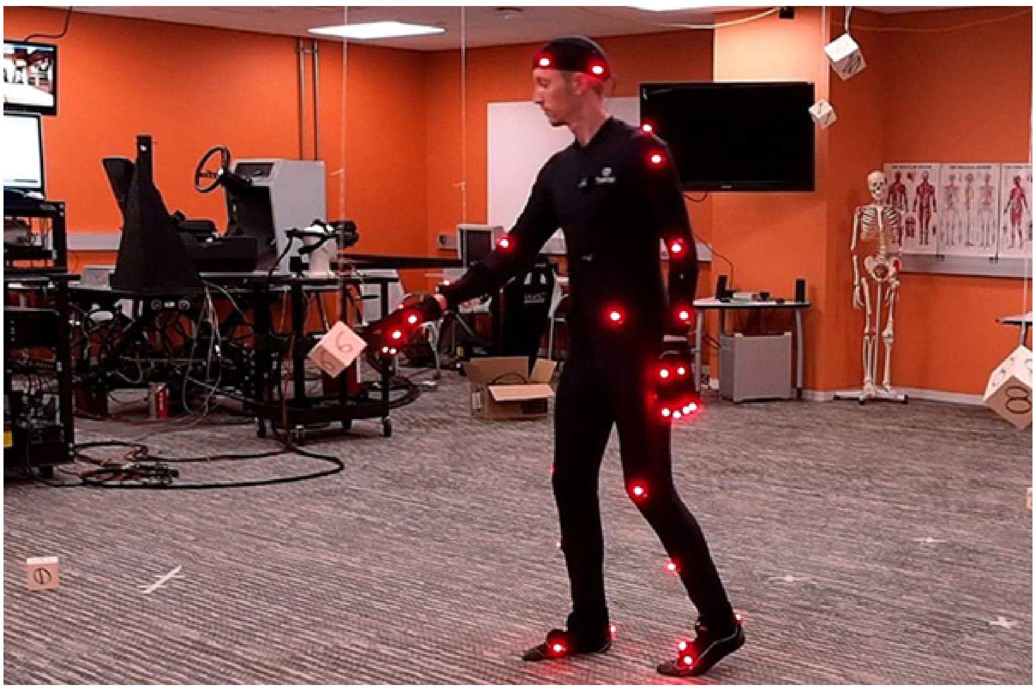

Figure 2.1: Modern active marker-based system, PhaseSpace

Marker-based systems usually utilize small reflexive markers attached to a person's joints. The unique appearance of the markers in the image allows for great precision of the system. They can be divided into passive, active and hybrid categories. Passive systems use reflexive markers, while active (Figure 2.1) use light-generating markers. Generally, infrared light is used in these systems [9]. They offer high accuracy and reusability - they can be easily used for tracking facial features, individual fingers, or various objects by reattaching. This makes them very popular for motion tracking in the entertainment industry. The downside of using a marker-based system is the space required and time to setup. They are also slightly more expensive than marker-free systems, but usually more affordable than non-visual systems.

2.1.2 Marker-free systems

Without relying on the existence of attached markers, the different strategies must be devised to effectively track the motion. This can be achieved by tracking boundaries or features (keypoints (§4.5)) of the human body. Main design decisions for such systems are structure analysis (model and non-model based) and camera configuration (single and multiple) [9]. Main advantage of these systems is their high portability, making them ideal for home rehabilitation scenarios. One the donwside, they lack the accuracy of marker-based systems, since the results are only estimations. They are also not universal - a new set of rules has to be created for each use-case, such as pose estimation, tracking facial features, etc.

2.2 Robot-aided motion tracking

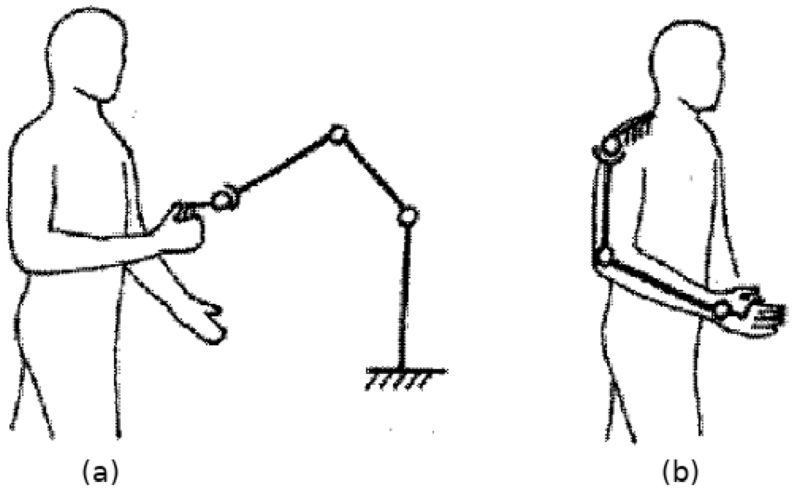

Figure 2.2: (a) End-effector robot. (b) Exoskeleton robot. Artwork by [10]

In the early days of robot-aided tracking, **end-effector robots** (Figure 2.2) were used. These robots, such as MIME [11] hold the subject's hand at one point and generate forces at the interface. The joints of this type of robots do not match the subject's, thus generating isolated movements or performing some exercise has proven difficult, if not impossible. They are, however, relatively cheap to produce and can be easily adjusted for various arm lengths.

Recently, research in robotic therapy shifted towards **exoskeleton robots** (Figure 2.2). This type of robots is attached to the patient's upper limb, closely copying its structure with aligned joints. Exoskeletons can be further divided into two categories:

Active exoskeleton can assist movements by generating force, for example, through Pneumatic Muscle Activators (PMAs) in exoskeletons such as RUPERT IV [12].

Passive exoskeleton can only provide resistive force, making them safer to use, but limiting their use to resistance and balance training. Example of a passive exoskeleton is Dampace [13].

2.3 Alternative non-visual motion tracking systems

The sensors are typically on-body and use various physical properties to track the motions. They are usually categorised as mechanical, inertial, acoustic, and magnetic based. Some of these sensors are accurate enough to detect even slight movements such as that of individual fingers. Popular among these is the usage of accelerometer because of their size and weight. On the other hand, accelerometers suffer from the "drift problem" [9] and require external correction throughout the tracking stage.

2.4 Deep Learning (DL)

Recent advancements in Deep Learning serves for implementation and training accurate marker-free tracking systems (§2.1.2), which resulted in an entire subfield of Human Pose Estimation (HPE) (§2.4.3).

The field of **Artificial Intelligence (AI)** was born in the 1950s and could be defined as *the effort to automate intellectual tasks normally performed by humans.* It encompases many subfields such as Machine Learning (Figure 2.3) as well as fields that do not involve any learning. Example of such approach would be chess program with many hardcoded rules - also known as the symbolic AI paradigm [14].

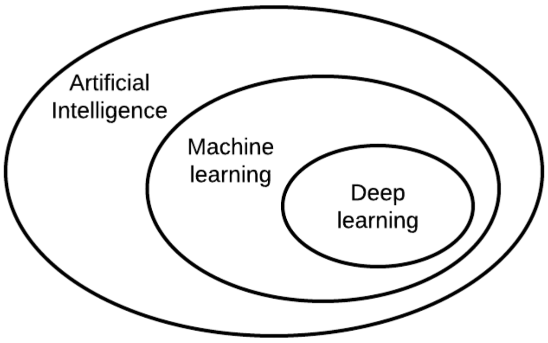

Figure 2.3: Artificial Intelligence, Machine Learning and Deep Learning

Traditional computation as well as symbolic AI systems consists of human inputting data as well as rules for operating on those data and getting output. With **Machine Learning (ML)** system, the system's task is to find the rules, that would either map given inputs onto outputs correctly (Supervised Learning), or organise the data logically (Unsupervised Learning). We say a ML system is trained as opposed to programmed. ML system attemps to learn statistical rules on a given dataset. As would be expected, the quality of the system highly depends on the quality of the training [14].

What sets the **Deep Learning (DL)** appart from other ML approaches is the focus on learning many successive layers of representations, each slightly more complex than the previous. These representations are learned via Neural Networks.

2.4.1 Deep Learning libraries

Over the years, many libraries emerged with the aim to simplify the process of implementing a Neural Network. In this section, we will briefly introduce the major ones used in modern DL.

TensofFlow was released by Google as open source in 2015 [15]. Overall, TensorFlow allows for the most control over the NN model at the cost of simplicity.

Keras is a high-level API running on top of a Deep Learning framework, such as TensorFlow, Microsoft Cognitive Toolkit or Theano. It was developed with a focus on enabling fast experimantation, making it very user-friendly. Other features include seamless running on a CPU or a GPU and support for convolutional networks, recurrent networks and combinations of them.

PyTorch was developed by Facebook in 2017 [16]. Due to it's simple debugging and TensorFlow's switching of APIs, PyTorch became popular in academia, with significant majority of papers from 2019 being implemented in Pytorch [17].

Despite the Pytorch's popularity among researchers, TensorFlow is still the dominant framework in the software industry due to its simpler deployment on various devices and potential for model optimization [17].

2.4.2 Convolution

The convolution operation inputs image and a filter (also known as kernel) — typically a 3x3 or 5x5 martix. The process of convolution consists of adding each element and its neighbors, multiplied by the kernel. Some notable examples of kernels are Identity, Edge detection, or Gaussian blur.

Each layer of Convolutional Neural Network (CNN) uses K filters on the input image to produce output of corresponding depth. Locations in the output feature map correspond to the locations in input feature map.

Convolutional layers are most often used for image recognition due to their inherent advanatages to the usual densely connected layers: whereas dense layers learn global patterns in their input feature space, convolutional layers can learn local patterns. These patterns are then translation invariant, meaning that the network is able to recognize the same pattern in different parts of the image without learning it anew.

Convolutional Neural Network can learn spatial hiearchies of patterns. First layer will typically learn small local shapes, such as edges, second will learn larger patterns made up of feature from previous layer and so on. This means that higher layers of the CNN are capable of detecting complex visual concepts, for example people and animals.

2.4.3 Pose estimation

Human Pose Estimation (HPE) models can be used to estimate either 2D coordinates of joints or full 3D model of a person from image(s). Since this thesis assumes only one camera without additional sensors or markers (monocular vision), we will concern ourselves only with 2D HPE. In further text, broader term HPE will refer to 2D HPE.

Singleperson HPE methods can either attempt to directly map input image to the joint coordinates (regression-based methods), or, in the case of detection-based techniques, to heatmaps of joint locations. Generally, heatmap representation is considered more robust [18]. [19] proposed a function to convert heatmap to coordinates. Variations of MobileNets [20, 21] have also been successfully adapted for HPE on mobile phones and other low-capacity devices [22].

Multiperson HPE adds the difficulty of the image containing unknown number of people, each of which may appear at any position or scale and be occluded by other people as well as objects.

More common top-down approach uses person detector and then passes cropped pictures to standard single person estimator. This approach is much simpler to implement, but relies heavily on the person detector, which may fail in case of occlusion. Its runtime also grows proportionally to the number of people in the image.

In contrast, modern bottom-up (Figure 2.4) approaches find all body parts in the image and then associate them together by various means, resulting in a consistent runtime even with many people present [23] at the cost of a slightly lower, yet usually sufficient, accuracy.

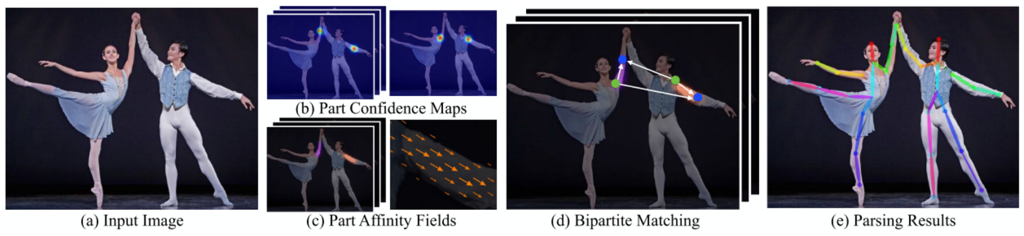

Figure 2.4: Pipeline of OpenPose, a bottom-up approach to HPE

2.4.4 Human Pose Estimation (HPE) methods

There is no shortage of well performing methods, although only a few provide real-time estimations. The following were picked for the training test because of their freely available pretrained models and consistent high scores on various datasets:

AlphaPose [24] is a top-down multiperson method specially designed to cope with inaccurate bounding boxes and redundant detections.

Multi-Stage Pose Network (MSPN) [25] is also a top-down multiperson method. It argues that quality of the detector is irrelevant beyond a certain point and proposes several improvements to the typical multistage estimation. The resulting approach outperforms many competitors.

OpenPose [23] is one of the most popular methods for general use in applications. It falls in the category of bottom-up multiperson HPEs. OpenPose first predicts a set of 2D confidence maps of body part locations and a set of 2D vector fields (Figure 2.4) of Part Affinity Fields (PAFs), encoding the degree of association between pairs. Both are then parsed by greedy inference.

After testing these popular methods on sample footage, only OpenPose was found satisfatory in both performance and accuracy. For example, shadows on a wall seem to be a significant problem even for these state-of-the-art models. Therefore, OpenPose was selected as the HPE model of choice for this thesis.

2.5 Patient's motivation

In order for the rehabilitation to be effective, patient must adhere to the assigned training regime. Some patients might feel demotivated just by the fact of having their movement impaired, others by time spent in the hospital, making further progress more difficult.

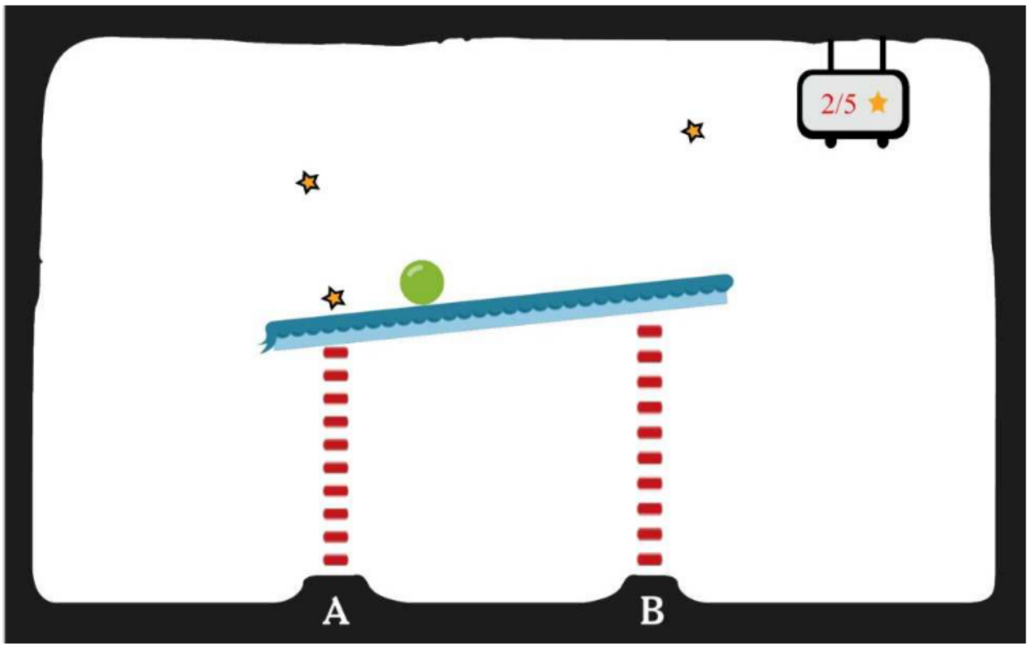

Figure 2.5: Collaborative games designed to incorporate rehabilitation exercises can motivate patients to further training

To make patients feel motivated to continue in their rehabilitation for longer period (especially in case of home rehabilitation), several approaches to make it more engaging were proposed. For example, several videogames where patients use assigned exercises to achieve in-game goals were developed [26]. One of such games, shown in (Figure 2.5), allows two players to play together (for example other patients or family members). Added social and game aspects increased the patient's motivation to continue with the training regime.

2.5.1 Virtual Reality (VR) games

Visualization of a performed task (such as putting books on a shelf instead of repeating the arm movement) has been shown to increase the effiency of training [10]. Virtual reality (VR) can easily combine such visualization with game and social aspects discussed. As VR becomes more widely available, rehabilitation programs in VR are becoming viable option for home based rehabilitation.

2.5.2 Dialogue games

Dialogue system is a special type of software designed to interact with its users by simulating human conversation through text or synthesised voice. Since the recent developments discussed in (§2.4), high-quality natural language processing has been made possible. [27] observed positive correlation between conversing with dialogue system and adherence to prescribed medication regime among 4737 patients with breast cancer. The patients were also more open to converse about deeper issues of life with cancer with the dialogue system than with human doctors.

The mental benefits were explored by Woebot [28]. In the study, 34 young adults with symptoms of having depression and anxiety were asked to engage with the dialogue system over the course of two weeks. Woebot, guided by the principles of Cognitive Behavioral Therapy (CBT), significantly reduced the symptoms.

It is fair to assume that as the dialogue system get more advancend in the coming years, their role as (always available) support will increase as well.

3 Physiotherapeutic exercises

Stroke is generally defined as *an episode of acute neurological dysfunction presumed to be caused by ischemia or haemorrhage, persisting for at least 24 hours or until death* [29].

Stroke rehabilitation is a restorative learning process aimed to speed up and maximize recovery by treating the disabilities caused by the stroke, and to prepare the survivor to reintegrate as fully as possible into normal life [30]. In practice, this means going through exercise movements similar to Activities of Daily Living (ADL). Some exercises are deliberately designed to resemble ADL as much as possible - this is especially pronounced in the rehabilitation-focused Virtual Reality (VR) games (§2.5.1).

The primary goals of rehabilitation among stroke survivors are to prevent further complications, minimize body impairment and to prevent future stroke occurrences [31]. The ability of the patients to successfully undertake ADL without assistance depends mainly on the recovery[1] of motor functions of the upper limb [33].

The rehabilitative therapy should begin as soon as the diagnosis is established and life-threatening problems are under control. Measures should begin with range-of-motion exercises and physiologically sound changes of position in bed on the day of admission and should be followed by progressively increasing levels of activity as soon as medically possible. Early mobilization also includes encouragement to resume self-care activities and socialization [30].

After the initial stage, some people recover completely without any need for further rehabilitation, while others are too incapacitated to benefit from it. Between these extremes are people with various degrees of disability. Generally speaking, patients with a high degree of disability are candidates for intense hospital rehabilitation program with several hours of exercises every day, until further recovery is achieved. Those with lesser disability are usually left with the choice of either low-intensity hospital program or home rehabilitation. The hospital program is typically more comprehensive, although the patient may prefer home rehabilitation for various reasons [30].

During the home rehabilitation, simple self-correcting exercises are recommended, as assistance by another person (i.e. a family member) might not be available. Typically, standard movements are modified to make room

[1] [32] defines motor recovery as *restoring the ability to perform a movement in the same manner as it was performed before injury.*

for self-correction. However, even with these measures, the patient's movement patterns might start to deteriorate over time (usually after getting familiar enough with the exercises to not actively think about them during a session). Regular checkups with the physiotherapist are therefore necessary to identify emerging problems.

3.1 Arm movement

Performing all of the various arm movements on daily or bi-daily basis has the rehabilitative effect necessary for the recovery of the motor functions of the upper limb. To better understand the concerns when constructing a rehabilitation regime, we will now briefly examine possible movements of the upper limb (Figure 3.1) and some typically prescribed exercises.

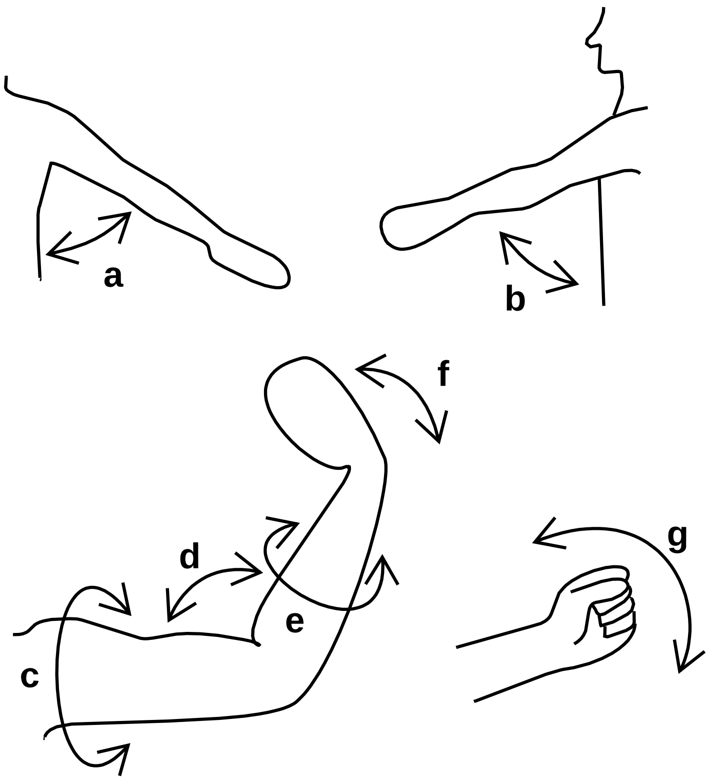

Figure 3.1: Possible movements of the human arm. (a) shoulder abduction / adduction, (b) shoulder flexion / extension, (c) shoulder rotation, (d) elbow flexion / extension, (e) pronation / supination, (f) wrist flexion / extension, (g) adduction / abduction

Wrist can perform flexion, extension, adduction and abduction motion. Additionally, radioulnar joints produces pronation and supination movement. Therefore wrist can move with three Degrees of Freedoms (DOFs) (i.e. along three axes).

Elbow allows for flexion and extension (one DOF). Elbow pain is usually caused by wrist and forearm muscles and thus various combinations of wrist movements are often prescribed.

Shoulder allows for the three DOFs and the greatest number of exercises including both individual and assisted exercises. During these exercises, it is important to keep straight back, as the shoulder movement is strongly interconnected with scapula's. Home rehabilitation varieties of recommended exercises usually involve either lying on the floor or leaning with back on a wall, as this allows patients to better sense a bad form and self-correct without supervision.

Shoulder, apart from grip strength, is often the most problematic when trying to restore the patient's ability to independently perform Activities of Daily Living (ADL). Because of this fact, the selected exercises (§3.2) revolve around shoulder movements.

3.2 Selected exercises

After considering the technical limitations of one camera (§2.1) and discussing the rehabilitative properties of various exercises with a professional physiotherapist, the following exercises were chosen for the basic application:

Bar raises The patient stands facing the camera, preferably with back and heels touching a wall (so as to prevent bending backwards in later parts of the movement), about two meters from the screen. He holds a bar (e.g. a broom handle) with both hands and overhand grip shoulder-width apart and slowly raised it without bending arms as far up as comfortable.

If one arm has a limited range of motion, the second arm will usually help raising the bar, slightly increasing this range. When both shoulders are forced out of their comfortable range, the patient typically arches his back, which is prevented by the aforementioned heels-to-wall stand. Other frequent mistakes include holding the bar with too wide or too narrow of a grip, raising shoulders at the end of the comfortable range or not holding the bar horizontally.

The aforementioned range of the shoulders can be used as a means of tracking long-term progress.

Bar rotations The patient starts in the same stance as in the previous exercise. He holds the bar with both arms straight in front of himself in a horizontal position. The patient then rotates the bar as if driving the steering wheel of an imaginary vehicle, as far as he is comfortable while keeping his back straight, and then to the other side.

The usual mistakes are similar to the first exercise, with the addition of rotating the bar significantly off-centre.

The comfortable angle with good form is a good measurement of long-term progress.

Shoulder rotations In the initial stages of rehabilitation therapy, it may be difficult for the patient to rotate his shoulder with arm fully extended. This exercise helps to avoid most common mistakes.

The patient sits sideways about one to one and a half meter from the camera - so that one elbow is fully visible in any position. He touches his shoulder with his fingers, if able. He then starts slowly rotating the arm while keeping the fingers touching the shoulder, in as big of an area as comfortable.

The most common mistake while performing this exercise is hunched back. The covered area highly depends on individual shoulder mobility, so it cannot be clearly classified as good or bad form. Individuals with limited range of motion will still benefit from performing this exercise.

The area covered by the elbow rotation in the air can be used as a measurement of progress.

In order to limit the scope of the thesis, decision was made to dedicate further text to just one exercise. For this purpose, the **bar raises** movement was selected, since it is easy to explain to participants (§4.2) through text and best suited for detection.

4 Data collection

4.1 Participants

The training (Table 4.1) data were collected by 10 volunteers (5 women and 5 men with average age of 38.70 ± 14.39, all caucasian race). Testing data (Table 4.2) were also collected by 10 voluntiers (6 women and 4 men with average age 47.50 ± 15.75, all caucasian race), where 4 of them provided different training and testing data (see the overalap of person id).

Person Id	Age	Gender
1	49	Female
2	47	Female
3	47	Female
4	58	Male
5	23	Male
6	20	Male
7	48	Female
8	20	Female
9	52	Male
10	23	Male

Table 4.1: Participants in the collection of training data.

Person Id	Age	Gender
7	48	Female
8	20	Female
9	52	Male
10	23	Male
11	70	Female
12	55	Female
13	49	Female
14	46	Female
15	70	Male
16	42	Male

Table 4.2: Participants in the collection of testing data.

4.2 Data Collection Instructions

Each subject was first instructed on the correct technique of the selected exercises (§3.2), as well as the usual mistakes (as described by the professional physiotherapist). Subjects were also instructed on the slightly slower tempo of movements prescribed by the physiotherapist, since not all of them have undergone rehabilitation before.

4.3 Video Exercises Collection

Due to the ongoing COVID-19 epidemy the participants kindly agreed to provide videos of themselves performing the exercises at home according to the instructions. Therefore the data had been gathered in a mostly uncontrolled environment. The videos could be captured on any capable device, as long as it was stabilized (i.e. notebook on a table, camera on tripod, etc.). Framerate of the device was deemed irrelevant, since the pace of the movements might vary greatly between people. Each exercise was to be performed at least 16 times, alternating each repetition between good form and doing one of the suggested mistakes (§3.2) (different each time).

4.4 Data preprocessing

The collected videos were then cut into smaller clips, each representing one full repetition of either good or bad form of a particular exercise (§3.2). It is necessary that each sample includes only the one repetition and nothing else for the model (§5) to be trained properly. For example, the occassional pause between exercises was removed. Each clip was annotated by hand as either a good or a bad example of the performed exercise.

Finally, directories 'good' and 'bad' were created, each containing respective samples. These were in turn moved into directories representing the discussed exercises (§3.2).

These clips were then passed to the OpenPose (§2.4.3) pretrained model to produce sequences of keypoint (§4.5) coordinates for training.

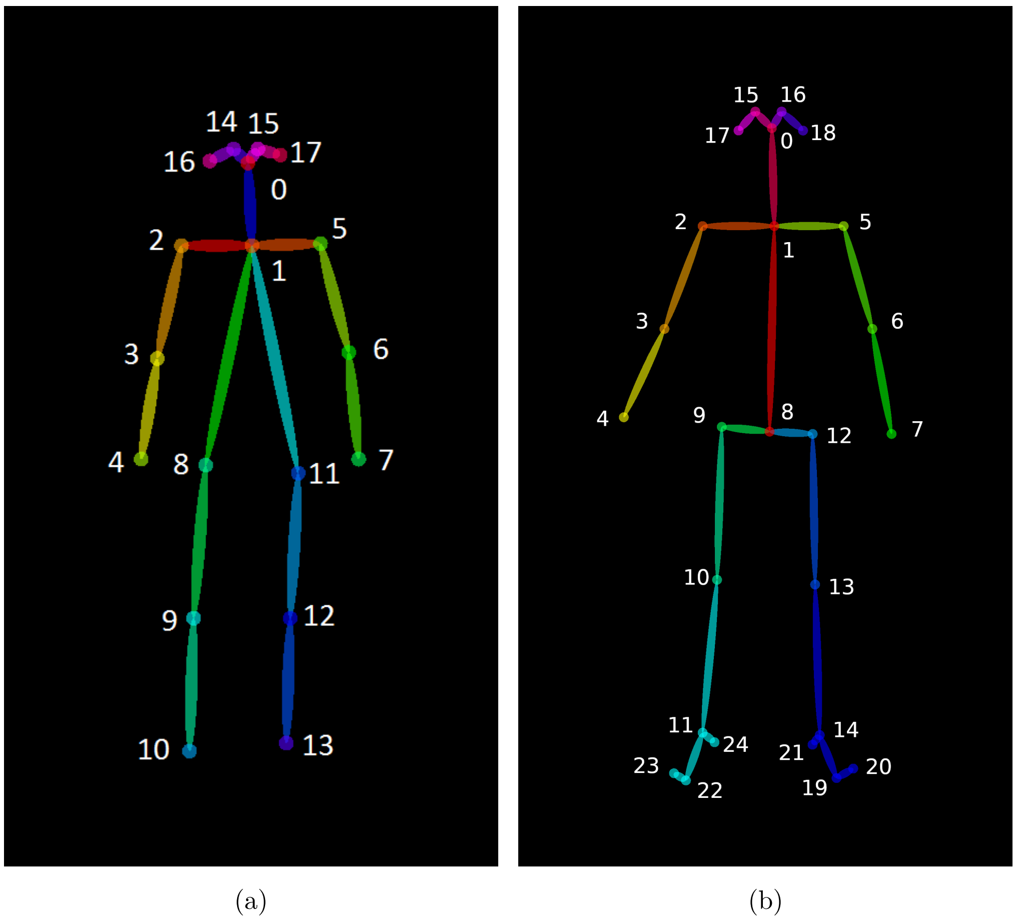

Figure 4.1: (a) standard COCO keypoint ordering (b) BODY_25, also available in OpenPose.

4.5 Keypoint detection

Output of most Human Pose Estimation (HPE) models is formatted according to the widely used COCO dataset [23]. Each keypoint consists of its x and y coordinates in the input image, scaled to the range [0, 1], as well as its confidence score. If the keypoint was not detected, it was substituted by [0, 0] with a confidence of zero.

Note that the participants did not wish to have the exercise videos published and thus only the anonymised collected coordinates are attached to this work.

This thesis only concerned with the movements of the upper limb. After discussion with the professional physiotherapist (§3), it was found that to correctly assess the quality of the arm exercises, it is necessary to track movement of head and torso as well. In terms of the COCO keypoint ordering (Figure 4.1(a)), these are:

- 0 - nose
- 1 - neck
- 2, 5 - shoulders
- 3, 6 - elbows
- 4, 7 - wrists
- 8, 11 - hips

After the discussion with the physiotherapist, other keypoints have been discarded at this stage, since they are considered irrelevant for the upper limb movement classification. Confidence scores have also been removed from the dataset, as the default thresholding in OpenPose (§2.4.3) ensured high quality of estimations anyway.

The default coordinate system places the origin into a corner of the video. During testing with basic models, it was found that this limits the user to perform the exercises only in the middle of the screen (where most training examples were performed). To combat this issue, decision was made to used relative coordinates to a point on the user's body. The neck keypoint has been selected since it is easy to detect (minimizes the chance it would not be found) and relatively motionless throughout the exercise. Even with added noise, the accuracy was not significantly impacted and classifier works well even in imperfect setting.

4.6 Balance of the dataset

Training data (Table 4.1) have been gathered in a manner that ensures maximal balance between good and bad classes. However, some clips had to be discarded. After doing so, the number of inidividual videoframes were not balanced (6461 positive / 5554 negative with difference of 14.04%) This was also due to the fact that many people speed up very slightly when knowingly performing bad form of an exercise. This difference was therefore reduced by gathering additional samples of the bad form, balancing the dataset to 6461 positive samples and 6444 negative samples (with difference of 0.26%).

The testing data (Table 4.2) weren't extra balanced, but it was collected with 4353 positive samples and 4570 negative samples (with the difference of -4.98%).

5 Training

In Machine Learning (ML), we distinguish between Supervised Learning, where the goal is to approximate a function that maps features (independent variables) onto target variable Y (dependent variable), and Unsupervised Learning, where no target variable is available [14, 34]. Since we have annotated dataset with known dependent variable, we can utilize supervised learning.

Supervised Learning can be further divided into regression and classification tasks. The regression refers to predicting a quantitative variable (e.g. a price of an item), while classification is concerned with predicting a qualitative variable (e.g. sentiment analysis). As mentioned in Data Collection (§4), we are interested in classifying the pose as either good or bad.

For the purposes of this thesis, we are only interested in discerning between good and bad form of one chosen exercise. Therefore, this is a binary classification problem. In the next sections, we will examine various binary classifiers and compare their results on the collected dataset (§4).

5.1 Binary Classifiers

There is a great number of methods for binary classification available, from the simplest approaches such as Logistic Regression to the State-of-the-Art Neural Networks. We will now introduce some of the most common, yet simple classifiers:

Logistic Regression [34] is a basic method for binary classification. It works by passing linear regression through logistic function. Given input datapoint, it calculates the probability of the datapoint belonging into the positive or negative class. Although there are variations designed to work with multiple classes, we are only concerned with the binary variant.

Through the querying the learned coefficients, logistic regression can be used to examine the contribution of each independent variable to the resulting classification. However, this method assumes linear relation between the independent variables and the dependent variable (linear separability).

K-Nearest Neighbors (KNN) [34, 35] is a lazy non-parametric algorithm, meaning that it does not require a separate training phase. When given an input datapoint, it calculates distance to all the other datapoints from the training set. K nearest points are then selected and a class with highest presence among them is assigned.

KNN is often used for pattern recognition and dimensionality reduction. Since this method lacks a training phase, prediction calculation is not reduced to a small set of learned coefficients. As a result, it's performance suffers when presented with high dimensional data, a large training dataset and/or categorical features. On the other hand, this approach doesn't assume anything about the dataset.

Support Vector Machine (SVM) [36, 37] attempts to find the optimal decision boundary. That is, a boundary where the margin from the nearest points (support vectors) of all classes is maximized.

Standard SVM assumes linear separability. For more complex data, a kernel function is used. SVM's kernel projects data into higher dimensions where they are linearly separable.Tuning hyperparameters are:

C is the cost associated with incorrect classification in SVM model. Larger C results in smaller margin for the sake of low training error, while smaller C gives us bigger margin and smoother decision boundary.

kernel when data is non-linearly separable, kernel function is used to project datapoints into higher dimensional space, where it can be separated. Common non-linear kernels are: gaussian, polynomial and sigmoid.

gamma when using kernel to project datapoints into higher dimensions, gamma controls the influence of an individual datapoint. With low gamma, the area of influence of each sample is greater than with high gamma.

degree controls the degree of polynomial kernel.

Random Forest [38, 39] combines multiple Decision Trees into one Ensemble learning model. A decision tree finds decision rules that divide the training data into smaller subsets. These decision rules are then organised into a tree structure to maximize purity of the gained subset at each step. Therefore the most important rule becomes the root node.

A single Decision Tree is however prone to overfitting [34]. A Random Forest utilises many Decision trees by constructing K decision trees, each according to a subset of the original dataset - N random datapoints. Input data are then passed to all the constructed trees and the result is averaged.

Random Forest is very stable compared to other algorithms and works well with missing values and/or bad feature scaling. However, passing the input data to a large number of decision trees might be slower than similar algorithms.

5.2 Evaluation

To accurately compare learning algorithms, various metrics can be used. Since they mostly stem from the confusion matrix (§5.2.1), we will shortly introduce it. Then follows the calcualted metrics (§5.2.2) which are derived from confusion matrics basic values.

5.2.1 Confusion Matrix

Confusion matrix [34] (Table 5.1) can be used for any classification model. Typically, rows represent actual (testing) class and columns represent the predicted class from testing data.

		Predicted class	
		P	**N**
Actual class	**P**	TP	FN
	N	FP	TN

Table 5.1: Confusion Matrix

In binary classification, the classes are Positive and Negative. This makes for 2x2 matrix containing these basic values:

True Positive (TP) is the number of Positive cases classified correctly.

True Negative (TN) is the number of Negative cases classified correctly.

False Positive (FP) is the number of cases where model predicted Positive class, but the actual class was Negative. Also known as Type 1 Error.

False Negative (FN) is the number of cases where model predicted Negative class, but the actual class was Positive. Also known as Type 2 Error.

5.2.2 Calculated Metrics

Out of these basic values, different metrics can be computed:

Precision (5.1) Out of all predicted Positive cases, how many were correct.

$$Precision = \frac{TP}{TP + FP} \tag{5.1}$$

Recall (Also known as True Positive Rate (5.2)) Out of all actual Positive cases, how many were correcty predicted.

$$Recall = \frac{TP}{TP + FN} \tag{5.2}$$

Accuracy (5.3) is the most common performance metric for balanced datsets (§4.6). It is a ratio of the number of correct predictions and total number of predictions. Doesn't work well for unbalanced dataset since it may give way to methods biased towards the majority class. Often used when every class is equally important.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{5.3}$$

1-Specificity (Also known as False Positive Rate (5.4)) Out of all actual Negative cases, how many were correcty predicted?

$$1 - Specificity = \frac{FP}{FP + TN} \tag{5.4}$$

F_1 **Score** (5.6) There is a trade-off between Precision and Recall. To evaluate both using a single value, F_β(5.5) Score is often used. When $\beta = 1$, both metrics are given equal importance. This metric is often used for unbalanced datasets. As can be seen from the equation, the F_1 Score skews towards high TP. In this work however, both positive and negative classes are given equal importance and thus F_1 Score was found to be sub-optimal to compare methods.

$$F_\beta = (1 + \beta^2)\frac{Precision * Recall}{\beta^2 * Precision + Recall} \tag{5.5}$$

$$F_1 = 2\frac{Precision * Recall}{Precision + Recall} \tag{5.6}$$

5.2.3 Metric Selection

For the purposes of model comparison, accuracy has been used. This is because the dataset is balanced (§4.6) and we place equal importance on correct classification of positive and negative cases.

5.2.4 Cross-Validation

The test set is typically gained by splitting the dataset into training and testing subsets (i.e. in a 80:20 ratio). Since the exact selection of test subset might influence the results, cross-validation is used. K-Fold cross-validation divides the dataset into K parts. In each of K iteration, one of them is used as test set, while the others comprise the training set. A mean and std. deviation of the metrics gained by each iteration then serves as a more stable model evaluation than a single train / test split. During training a 5-Fold cross-validation was used.

5.3 Models Comparison

The described methods were used to develop several models for classifying the correct exercise techniques. These models were then compared using the accuracy metric and cross-validation.

5.3.1 Feature Selection

Models are trained using coordinates of only certain keypoints, as specified in §4.5. These coordinates were converted to coordinates relative to allow for exercising in various location within the frame, as detailed in §4.4. The coordinates coresponding to the neck were then discarded.

5.3.2 Model Baseline

To make sure the model is effective, a baseline has to be first established. Since the dataset is only slightly unbalaced (§4.6), simple Majortiy class classifier has been selected as a baseline. This simple approach predicts the most frequent class in the dataset.

5.3.3 Hyperparameters

In Machine Learning, we distinguish between parameters optimized by the models themselves by training and parameters that are passed before the training starts. The latter is often called hyperparameters. To arrive at a good combination of hyperparameters, a grid search is commonly used. This approach simply creates models for all of the combinations of hyperparameters, trains them, and finally, compares them using defined metric. By doing this, we can arrive at an approximation of optimal hyperparameters.

5.3.4 Prediction Models

The table 5.2 shows accuracies of described methods 5.1. First, the simple method of logistic regression was tested. Since the accuracy score reached is only slightly above baseline, the dataset doesn't appear to be linearly separable. We can further test this with SVM with linear kernel. The reached accuracy is not very good either.

We can conclude that the dataset is not linearly separable. By intuition, the joints may move only within a certain range with good form. On both extremes of this range, bad form is detected. Since our dataset is not linearly separable, methods assuming this property will never perform well. This rules out Logistic Regression and SVM with linear kernel.

Next, we compare the other methods on the dataset in order to choose few promising candidates for optimization.

All the previous results are show in 5.2 summary table with respective model hyperparameters (§5.3.3) tunning.

Method	Accuracy	C	γ	kernel	degree	K	N	max. depth
Baseline	50.07 (±0.00)							
LR[1]	53.03 (±11.89)							
SVM	55.21 (±9.22)	1	1/18	lin.				
SVM	59.36 (±8.87)	1	1/18	poly.	3			
SVM	56.93 (±10.06)	1	1/18	gauss.				
SVM	62.29 (±9.15)	50	35	gauss.				
KNN	65.39 (±9.22)					5		
DT[2]	56.52 (±6.68)							unlimited
RF[3]	61.71 (±10.85)						100	unlimited
RF	61.83 (±10.86)						100	450

Table 5.2: Various ML models accuracy comparison (with 5-Fold Cross-Validation (§5.2.4))

[1]Logistic Regression

[2]Decision Tree

[3]Random Forest

We can see that Random Forest, SVM and KNN show good results and thus are suitable for further optimization.

After searching the hyperparamter space through grid search (§5.3.3), the best three results were achieved by KNN($K = 5$), SVM(gaussian kernel, $C = 50$, $\gamma = 35$) and Radnom Forest($N = 100$, max. depth = 450)

The best model for the user application (§6) was therefore the KNN model.

5.4 Exporting the model

In order to reuse the model in the final application (§6) without having to retrain from scratch, the model has to be exported to a file. Python provides built-in solution, the pickle module[4]. The module can serialize any object to a file, allowing persistance. This, however, is not without significant downside, as the exact mechanism depends on the installed version of Python and might not be compatible even in the case of minor version changes.

Scikit-learn[5] offers its own solution in the form of joblib module[6]. The module works similar way as pickle. Being custom made for scikit-learn models, it is more efficient with large numpy[7] arrays. Although minor changes in Python versions should not be a problem, the same version of scikit-learn is required.

Manually implementing export and import functions to a file (e.g. JSON) is perhaps the most flexible option. However, since the final application will include scikit-learn for inference anyway, joblib has been deemed as the most appropriate solution.

[4]https://docs.python.org/3/library/pickle.html
[5]https://scikit-learn.org
[6]https://joblib.readthedocs.io
[7]https://numpy.org

6 Application

For the purposes of this thesis, a simple desktop application was developed to enable testing of the trained model with non-technical users.

The aim of the application is to assess the quality of the performed exercises in the real-time and provide the user with feedback. As mentioned in (§5), we are only concerned with estimating the form as either good or bad.

Since the target user doesn't have technical background, the application had to be designed with accessibility in mind. One of the most important aspects of designing an accessible Graphical User Interface (GUI) is to be self-explanatory. The application should not require additional explanation apart from a basic overview of its purpose.

6.1 Usage of Application

Upon starting the application, an introductory screen is displayed. The screen contains a welcome message and buttons to exit the application or to select an exercise.

When the exercise is selected, the next screen shows detailed instructions about the exercise. Upon finishing reading the instructions, the user clicks the "OK" button to navigate to the final screen.

The final screen contains footage from the video device with feedback to the user and a button to navigate back to the main menu (the introductory screen).

For the purposes of this thesis, the application gives feedback in the form of colored border (green is good, red is bad) around the live webcam video. The application is expected to integrate into a bigger system, where the feedback is handled differently.

6.2 Architecture

The application follows Model-View-Controller (MVC) architecture in its design with high-level overview below (Figure 6.1). This ensures the separation of the application logic (the model) and the Graphical User Interface (GUI). The purpose of the view part of the architecture is to display relevant information in a GUI environment, making it simple to use for non-technical user. Communication between view and model is done by the controller part, which reacts to the events in GUI (e.g. user clicks a button) and updates model accordingly. The controller also requests data from model to redraw view (e.g. calculating a new score to display).

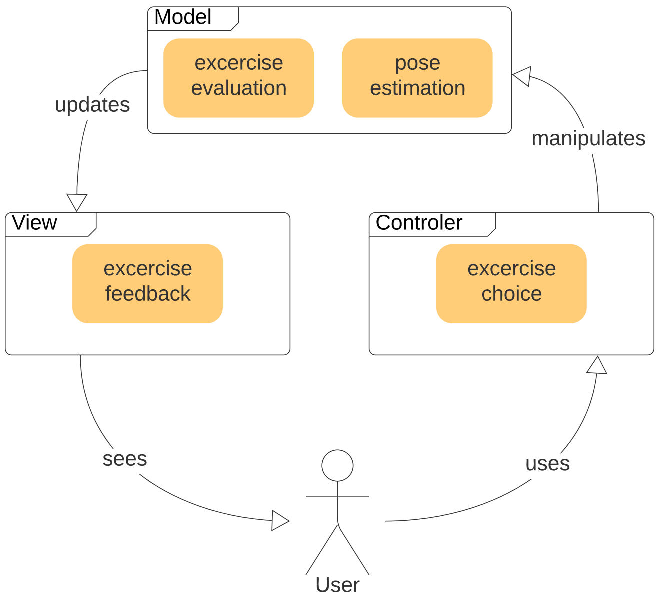

Figure 6.1: MVC

The view layer contains one class for every screen (sometimes also called stage - particularly in JavaFX terminology) in the application. Each screen contains various elements (called widgets) such as buttons, labels and images. The screen class itself is a subclass of widget, which allows for creation of custom widgets to reuse in the application. However, since the design of this application is simple (i.e. does not need reusable components), basic elementary widgets were sufficient.

In MVC architecture, there is usually one controller per screen in view layer. However, since the standard Python module for GUI programming, tkinter[1], allows the usage of lambda expressions, the most simple interactions (e.g. changing to other screen) do not require dedicated controller class for that view. The application does contains a main controller, MainApplication (Figure 6.1), which handles changing the view class and initializing the proper dedicated controller for it.

At the heart of every MVC application lies the model layer, which handles its internal logic. The model should be made in such a way to be fully

[1]https://docs.python.org/3/library/tkinter.html

functional even without the other two layers. It can be a collection of library functions to be called, up to a full collection of classes with their internal states and public interface. In the application, this calling is handled by the controller classes. In the developed applicatoin, the model classes handle the image processing part of the dataflow. The responsibilities are: getting keypoints (§2.4.3) from the input image, predicting class from keypoints (§4.5), etc. Most of the model classes were written as wrappers around library objects, which allows to swap these libraries in a modular fashion by writing a new wrapper.

6.3 Dataflow in the application

Dataflow of tha application follows the high-level sequence diagram (Figure 6.2) with the detailed flow described below.

Navigation to different screens is handled by the MainApplication class. Navigation buttons send request to the controller with the name of the screen to be drawn and optional additional arguments to pass to its constructor (such as which exercise was selected by the user in previous screen). The MainApplication creates the new view, and, if possible, the corresponding controller. Old screen is then destroyed and replaced with the new one. Views and controllers may implement destroy method to perform clean-up beyond standard behaviour of tkinter (e.g. closing the videofeed)

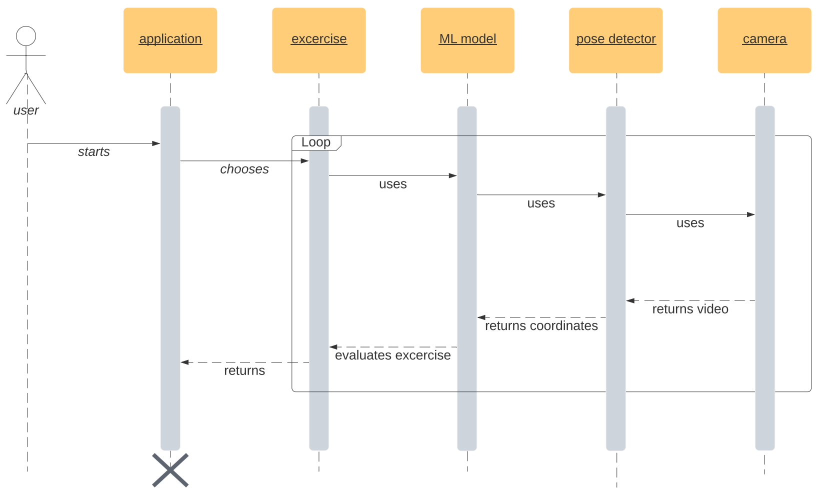

Figure 6.2: Data Flow

Upon navigating to the final screen, the video device is opened and an evalutation loop starts. After getting next frame from the device, it is passed to HPE wrapper, which returns selected keypoints (§4.5) in the same ordering as training data (§4). These keypoints are then used as an input for the fitted model, which predicts the class of the data. After the classification is complete, the image is given colored border based on the class. Finally, the controller updates the image displayed by view.

6.4 Testing the application

The developed application primarily utilizes the optimized K-Nearest Neighbors (KNN) method, since it was the best model performing on the training dataset (Table 5.2), but testing included also the another two best performing methods (SVM and Random Forest). The trained model object is loaded directly from a file (§5.4) into the application, which simplifies the implementation.

The application was tested with 10 volunteers (§4.1) and the test data was collected (Table 4.2). The participants were given basic instructions about how to use the application (§6.1) and instructed how to perform the excercise (§4.2). Each participant was then asked to perform four repetitions of the **bar raises** exercise (§3.2). While performing the exercises, participants were randomly asked to make one of the described mistakes (§3.2).

The test data (Table 4.2) from the experiment videos were captured in a manner similar to the data collection (§4) of the training data (Table 4.1). This allowed us to use them as testing set and compare the three best performing methods (Table 6.1).

6.4.1 Frame-wise testing

The captured videos were annotated and used to test the trained models on frame-by-frame basis. The reached results were following:

Method	Cross-Validation accuracy	Testing accuracy
KNN	65.39 (±9.22)	70.28
Random Forest	61.83 (±10.86)	69.02
SVM	62.29 (±9.15)	63.99

Table 6.1: Accuracy comparison of the final models during cross-validation (Table 5.2) and on the testing data.

The table 6.1 shows that the K-Nearest Neighbors (KNN) method outperforms the other algoritms on never-before seen data. The accuraccies on unseen data are slightly above average training accuracy, yet still within the boundaries of standard deviation.

6.4.2 Sequence-wise testing

Since the frame-wise classification is too fine-grained for the user, the overall feedback during the whole repetition was also evaluated. If during each repetition, the model responded with at least 60% (intuitively chosen) of the frames correctly classified, it was considered a successful classification. Any prolonged misclassification was considered an error. In most cases, the overall feedback was often apparent by observing the frame-wise classification. This was later verified with the captured test set.

Method	Frame accuracy	Sequence accuracy
KNN	70.28	72.50
Random Forest	69.02	72.50
SVM (gauss. kernel)	63.99	55.00

Table 6.2: Comparison of accuracies between frame-wise and sequence-wise testing.

It was soon discovered that two limiting factors of the models are the requirement of somewhat level camera (i.e. uneven surface might hinder detection) and keeping the suggested distance from the camera.

Since the accuracy highly depends on the quality of Human Pose Estimation (HPE) output (§2.4.3), it comes at no surprise that the biggest problems were caused by errors in the pose detection. For example, poorly lit clips were often changing between being classified as good and bad technique.

The biggest hurdle of the trained model was the lower end of the **bar raises** movement (§3.2). In this position, model often misclassified as bad form. This suggests an issue with the collected data, where the movement was not completed to a full range of motion by all participants.

7 Conclusion

Human Pose Estimation, Action Recognition and rehabilitation are all unique and difficult challenges. In this thesis, it was shown that Machine Learning (ML) can be used to develop effective in-home rehabilitation option without the need for specialized accessories.

After several discussions with a professional physiotherapist, we present the general process of rehabilitation among post-stroke patients in (§3). A comprehensive overview of the mechanics of the human arm was presented (§3.1) and three often prescribed exercises were described in (§3.2).

Various motion detection techniques were presented in (§2). To detect arm movement in video without any special accessories, Deep Learning was introduced in (§2.4) together with the modern methods of Human Pose Estimation (HPE) (§2.4.3).

To correctly classify the quality of the performed exercise, video data from 10 volunteers were collected (§4). These collected datapoints were then annotated while consulting with physiotherapist (§4.4).

Based on the gathered data, various Machine Learning methods were tested in (§5.3). Three methods with the highest accuracy score were then optimized to maximize this metric in (§5.3.3).

Finally, an application was developed to allow simple usage of the trained models in (§6). This application was then presented to another 10 volunteers to test the best-performing model. The testing data were also collected and used as a final evaluation of the trained models in (§6.4).

7.1 Future work

As shown in the results during testing, presented ML methods can predict with reasonable accuracy (§5.3). However, the intended use-case would require higher accuracy to be robust. In the this section, some options intended to overcome the shortcomings of the presented solution are proposed.

Used method of data annotation was too coarse. As detailed in (§4.4), the collected videos were passed to OpenPose to convert to keypoint coordinates. However, this method gives only estimates of the coordinates.

Using either a high-quality (but slow) model just for preprocessing, or creating the pose data manually would significantly increase the overall quality of the dataset.

Another issue of the dataset is that the labels were assigned based on clips of entire exercises. However, in many of the clips the mistake occured only in certain parts of the movement. This led to the result where even the well-performed parts of the exercise were labeled as a bad form.

This leads to an option to improve the model while simultaneously getting more useful feedback. For future work, it would be helpful to annotate on a lower level, only parts of the movement where the mistake occured. This would also allow us to annnotate with the exact mistake done, thus allowing the predictions to give helpful feedback to the user.

Another option to improve performance would be to use Neural Network (NN). NNs have proved themselves very useful in solving problems beyond the scope of traditional Machine Learning methods. However to design and train a Neural Network would probably require significant amount of data and hardware power beyond the scope of a thesis.

Usage of Neural Network would also allow properly analyzing the movement as a sequence istead of just individual frames. Various Recurrent Neural Networks (RNNs) have long been a saple of the field of action recognition.

Bibliography

[1] C. Warlow, C. Sudlow, M. Dennis, J. Wardlaw, and P. Sandercock, "Stroke," *Lancet*, vol. 362, no. 9391, pp. 1211–1224, 10 2003.

[2] T. Bryndziar, P. Šedová, and R. Mikulík, "Incidence cévní mozkové příhody v Evropě -systematická review," *Ceska a Slovenska Neurologie a Neurochirurgie*, vol. 80, no. 2, pp. 180–189, 2017.

[3] E. J. Benjamin, C. Paul Muntner, V. Chair Alvaro Alonso, F. S. Marcio Bittencourt, M. W. Clifton Callaway, F. P. April Carson, F. M. Alanna Chamberlain, A. R. Chang, M. Susan Cheng, F. R. Sandeep Das, F. N. Francesca Delling, M. Luc Djousse, M. S. Mitchell Elkind, F. F. Jane Ferguson, F. Myriam Fornage, F. Lori Chaffin Jordan, F. S. Sadiya Khan, B. M. Kissela, M. L. Kristen Knutson, T. W. Kwan, F. T. Daniel Lackland, F. T. Tené Lewis, J. H. Lichtman, F. T. Chris Longenecker, M. Shane Loop, P. L. Lutsey, F. S. Seth Martin, F. Kunihiro Matsushita, F. E. Andrew Moran, F. E. Michael Mussolino, F. O. Martin, A. Pandey, M. M. Amanda Perak, M. D. Wayne Rosamond, F. A. Gregory Roth, F. K. Uchechukwu Sampson, F. M. Gary Satou, F. B. Emily Schroeder, F. H. Svati Shah, F. L. Nicole Spartano, A. Stokes, D. L. Tirschwell, F. W. Connie Tsao, V. P. Chair Elect Mintu Turakhia, F. B. Lisa VanWagner, F. T. John Wilkins, F. S. Sally Wong, F. S. Salim Virani, and C. Elect, "Heart Disease and Stroke Statistics-2019 Update: A Report From the American Heart Association," *Circulation*, vol. 139, pp. 56–528, 2019. [Online]. Available: http://ahajournals.org

[4] V. L. Feigin, C. M. Lawes, D. A. Bennett, and C. S. Anderson, "Stroke epidemiology: A review of population-based studies of incidence, prevalence, and case-fatality in the late 20th century," pp. 43–53, 1 2003.

[5] C. Sudlow and C. Warlow, "Comparable Studies of the Incidence of Stroke and its Pathological Types," *Stroke*, vol. 28, no. 3, pp. 491–499, 3 1997.

[6] R. W. Teasell and L. Kalra, "What's new in stroke rehabilitation: back to basics." *Stroke*, vol. 36, no. 2, pp. 215–7, 2 2005. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/15637318

[7] “World Population Ageing,” United Nations - Department of Economic and Social Affairs, Population Division, New York, Tech. Rep., 2015.

[8] Český statistický úřad, “Proměny věkového složení obyvatelstva ČR - 2001-2050,” Tech. Rep., 2019. [Online]. Available: www.plzen.czso.cz

[9] H. Zhou and H. Hu, “Human motion tracking for rehabilitation—A survey,” *Biomedical Signal Processing and Control*, vol. 3, no. 1, pp. 1–18, 1 2008. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1746809407000778

[10] A. Frisoli, L. Borelli, A. Montagner, S. Marcheschi, C. Procopio, F. Salsedo, M. Bergamasco, M. C. Carboncini, M. Tolaini, and B. Rossi, “Arm rehabilitation with a robotic exoskeleleton in Virtual Reality,” in *2007 IEEE 10th International Conference on Rehabilitation Robotics, ICORR’07*, 2007, pp. 631–642.

[11] C. G. Burgar, P. S. Lum, P. C. Shor, and H. F. Van Der Loos, “Development of robots for rehabilitation therapy: The Palo Alto VA/Stanford experience,” *Journal of Rehabilitation Research and Development*, vol. 37, no. 6, pp. 663–673, 2000.

[12] S. Balasubramanian, H. R. Wei, M. Perez, B. Shepard, E. Koeneman, J. Koeneman, and J. He, “Rupert: An exoskeleton robot for assisting rehabilitation of arm functions,” in *2008 Virtual Rehabilitation, IWVR*, 2008, pp. 163–167.

[13] A. H. Stienen, E. E. Hekman, F. C. Van Der Helm, G. B. Prange, M. J. Jannink, A. M. Aalsma, and H. D. Van Kooij, “Dampace: Dynamic force-coordination trainer for the upper extremities,” in *2007 IEEE 10th International Conference on Rehabilitation Robotics, ICORR’07*, 2007, pp. 820–826.

[14] F. Chollet, *Deep Learning with Python.* Manning Publications Co., 2018.

[15] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng,

and G. Research, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems,” Tech. Rep., 2015. [Online]. Available: www.tensorflow.org.

[16] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. D. Facebook, A. I. Research, Z. Lin, A. Desmaison, L. Antiga, O. Srl, and A. Lerer, “Automatic differentiation in PyTorch,” Tech. Rep., 2017.

[17] H. Ho, “The State of Machine Learning in 2019,” 2019.

[18] Y. Chen, Y. Tian, and M. He, “Monocular human pose estimation: A survey of deep learning-based methods,” *Computer Vision and Image Understanding*, vol. 192, p. 102897, 3 2020.

[19] D. C. Luvizon, H. Tabia, and D. Picard, “Human pose regression by combining indirect part detection and contextual information,” *Computers and Graphics (Pergamon)*, vol. 85, pp. 15–22, 12 2019.

[20] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” 4 2017. [Online]. Available: http://arxiv.org/abs/1704.04861

[21] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 1 2018. [Online]. Available: http://arxiv.org/abs/1801.04381

[22] B. Debnath, M. Orbrien, M. Yamaguchi, and A. Behera, “Adapting MobileNets for mobile based upper body pose estimation,” in *Proceedings of AVSS 2018 - 2018 15th IEEE International Conference on Advanced Video and Signal-Based Surveillance.* Institute of Electrical and Electronics Engineers Inc., 2 2019.

[23] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, “OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields,” Tech. Rep., 2019. [Online]. Available: https://gineshidalgo.com

[24] H.-S. Fang, S. Xie, Y.-W. Tai, C. Lu, S. Jiao Tong University, and T. YouTu, “RMPE: Regional Multi-Person Pose Estimation,” Tech. Rep., 2018. [Online]. Available: https://cvsjtu.wordpress.com/rmpe-regional-multi-person-pose-estimation/

[25] W. Li, Z. Wang, B. Yin, Q. Peng, Y. Du, T. Xiao, G. Yu, H. Lu, Y. Wei, and J. Sun, "Rethinking on Multi-Stage Networks for Human Pose Estimation," Tech. Rep., 2019. [Online]. Available: https://github.com/megvii-detection/MSPN

[26] L. Vanacken, S. Notelaers, C. Raymaekers, K. Coninx, W. van den Hoogen, W. Ijsselsteijn, and P. Feys, "Game-based collaborative training for arm rehabilitation of MS patients: a proof-of-concept game," pp. 1–10, 2010.

[27] B. Chaix, J. E. Bibault, A. Pienkowski, G. Delamon, A. Guillemassé, P. Nectoux, and B. Brouard, "When chatbots meet patients: One-year prospective study of conversations between patients with breast cancer and a chatbot," *Journal of Medical Internet Research*, vol. 21, no. 5, pp. 1–7, 2019.

[28] K. K. Fitzpatrick, A. Darcy, and M. Vierhile, "Delivering Cognitive Behavior Therapy to Young Adults With Symptoms of Depression and Anxiety Using a Fully Automated Conversational Agent (Woebot): A Randomized Controlled Trial," *JMIR Mental Health*, vol. 4, no. 2, p. e19, 6 2017. [Online]. Available: https://mental.jmir.org/2017/2/e19/

[29] R. L. Sacco, S. E. Kasner, J. P. Broderick, L. R. Caplan, J. J. Connors, A. Culebras, M. S. Elkind, M. G. George, A. D. Hamdan, R. T. Higashida, B. L. Hoh, L. S. Janis, C. S. Kase, D. O. Kleindorfer, J. M. Lee, M. E. Moseley, E. D. Peterson, T. N. Turan, A. L. Valderrama, and H. V. Vinters, "An updated definition of stroke for the 21st century: A statement for healthcare professionals from the American heart association/American stroke association," *Stroke*, vol. 44, no. 7, pp. 2064–2089, 2013.

[30] W. B. S. P. W. D. Glen E. Gresham, M.D., *Post-Stroke Rehabilitation*, 1995. [Online]. Available: https://books.google.cz/books?hl=en&lr=&id=5GgQYAPWyLAC&oi=fnd&pg=PA1&dq=post+stroke+rehabilitation&ots=F1bQWYhrAe&sig=4xP5C826RsiwQ9q2zdha9cEmUQk&redir_esc=y#v=onepage&q=poststrokerehabilitation&f=false

[31] P. W. Duncan, R. Zorowitz, B. Bates, J. Y. Choi, J. J. Glasberg, G. D. Graham, R. C. Katz, K. Lamberty, and D. Reker, "Management of Adult Stroke Rehabilitation Care: a clinical practice guideline." *Stroke*, vol. 36, no. 9, pp. 100–43, 9 2005. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/16120836

[32] M. F. Levin, J. A. Kleim, and S. L. Wolf, “What do motor "recovery" and "compensationg" mean in patients following stroke?” *Neurorehabilitation and Neural Repair*, vol. 23, no. 4, pp. 313–319, 5 2009. [Online]. Available: http://journals.sagepub.com/doi/10.1177/1545968308328727

[33] J. M. Veerbeek, G. Kwakkel, E. E. Van Wegen, J. C. Ket, and M. W. Heymans, “Early prediction of outcome of activities of daily living after stroke: A systematic review,” *Stroke*, vol. 42, no. 5, pp. 1482–1488, 5 2011. [Online]. Available: https://www.ahajournals.org/doi/10.1161/STROKEAHA.110.604090

[34] T. Hastie, R. Tibshirani, G. James, and D. Witten, *An Introduction to Statistical Learning.* New York: Springer, 2013.

[35] T. M. Cover and P. E. Hart, “Nearest Neighbor Pattern Classification,” *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[36] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, no. 3, pp. 273–297, 9 1995. [Online]. Available: https://link.springer.com/article/10.1007/BF00994018

[37] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “Training algorithm for optimal margin classifiers,” in *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory.* New York, New York, USA: Publ by ACM, 1992, pp. 144–152. [Online]. Available: http://portal.acm.org/citation.cfm?doid=130385.130401

[38] T. K. Ho, “Random decision forests,” *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, vol. 1, pp. 278–282, 1995.

[39] ——, “The random subspace method for constructing decision forests,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, 1998.