Master Thesis

TWENTE UNIVERSITY

DEVELOPMENT OF AN AUTOMATED EXERCISE DETECTION AND EVALUATION SYSTEM USING THE KINECT DEPTH CAMERA.

Frodo Muijzer

FACULTY OF ELECTRICAL ENGINEERING, MATHEMATICS AND COMPUTER SCIENCE
BIOMEDICAL SIGNALS AND SYSTEMS

EXAMINATION COMMITTEE

Prof.dr.ir. H.J. Hermens

29-01-2014

Preface

This master thesis gives a detailed description on how the Microsoft Kinect camera can be used for automated rehabilitation exercise evaluation in a non-supervised setting. The research was done under the authority of Roessingh Research and Development (RRD) and forms the conclusion of the master curriculum Biomedical Engineering at the University of Twente. Daily supervision is in the hands of Harm op den Akker and Thijs Tönis, both PhD students at the Telemedicine group of Roessingh Research and Development and the Remote Monitoring and Treatment group of the University of Twente. Ronald Poppe, a postdoctoral researcher at the Human Media Interaction group of the University Twente is the external advisor. Last, Hermie Hermens, professor in Telemedicine and chairman of the Remote Monitoring and Treatment group of the University of Twente, is the graduate professor.

Abstract
Due to a growing number of chronically ill patients, there is an increasing demand for automated rehabilitation exercise detection and evaluation systems which can be used in a non-supervised out-of-clinic setting. This report described the development and implementation of a proof-of-principle exercise detection and evaluation framework. The objective was to find out whether the affordable Microsoft Kinect depth camera can be used for such an exercise evaluation system. Microsoft developed the Kinect depth camera to enable control of specially designed games via body movements. Unfortunately, the Kinect cannot track subtle movements. In Chapter 3 of this thesis it is shown that out of 109 realistic rehabilitation exercises, 98 would certainly not be suitable for evaluation with the Kinect depth camera without significant adaptations.

In order to detect and evaluate an exercise, the exercise has to be taught to the computer system, either via automated learning, or via explicitly defining the parameters. For this project, the latter was chosen, because it provided the context information needed for evaluation. Unfortunately, no generally accepted method exists to parameterize an exercise. Therefore, the concepts of a method used to notate dances, Labanotation, were used to develop a new parameterization method. This method (described in Chapter 4) parameterizes an exercise by first defining the relevant body parts, then dividing the whole exercise into segments of a specific duration, and for each segment describing the movements of each relevant body part in terms of horizontal and vertical translations and rotations.

Chapter 5 gave a method to convert the parameterized exercise into an exercise playlist. The chapter also described how to convert joint positions, measured by the Kinect, into the translations used in the parameterization (horizontal, vertical and rotation). Next a method was given to compare a single measured translation to an arbitrary element from the exercise playlist. Finally, the difficult issue of when to advance to subsequent items in the playlist was described. At the start of the exercise, this comparison can be made between the first measured items, and the first items from the exercise playlist. Because the detection algorithm might fail to detect movements, the system is able to advance even if not all previous items have been matched. It can advance when the larger part of the previous items could be matched, or when the current measured translations form a good match to a future part of the exercise playlist. Missed exercise specification elements are marked, enabling evaluation of the movements the user failed to make. Chapter 6 described the implementation of the detection and evaluation system.

Chapter 7 discussed the protocol and results of experiments carried out to test the performance of the system. Unfortunately, the results of these experiments were not positive. The main issue lies outside the scope of the implementation: the subpar skeleton tracking performance of the Kinect SDK. Solely based on this, it can be stated that the Kinect depth camera cannot be used for automated rehabilitation exercise evaluation without alteration of the exercises, or exercise specific workarounds.

Despite negative experiment results, the developed Labanotation based parameterization method provided a good balance between a too cumbersome quantitative notation and a too vague text-based notation. The method was suitable for both the specification and detection of the movements, enabling straightforward comparison between the exercise specification and user performance.

Samenvatting

Door het toenemende aantal chronisch zieken, is er een groeiende behoefte aan systemen die revalidatieoefeningen kunnen herkennen en evalueren in een thuissituatie zonder professionele ondersteuning. Deze masterthesis beschrijft een haalbaarheidsonderzoek naar een raamwerk voor een herkennings- en evaluatiesysteem dat gebruik maakt van de Microsoft Kinect dieptecamera. De Kinect is door Microsoft ontwikkeld om computerspellen te besturen middels lichaamsbewegingen. Helaas is het aantal lichaamsdelen dat de Kinect kan herkennen beperkt, en herkent het subtiele bewegingen niet. In hoofdstuk 3 van deze thesis is aangetoond dat, zonder aanpassingen, slechts 11 van de 109 realistische revalidatieoefeningen geschikt zouden zijn voor evaluatie met de Kinect.

Alvorens het computersysteem een oefening kan herkennen en evalueren, moet het systeem bekend gemaakt worden met de oefening. Er zijn twee manieren om een oefening in te leren, middels automatisch leren of via het toekennen van expliciete parameters. De laatste methode is gekozen, omdat deze de contextinformatie geeft die nodig is voor de evaluatie. Helaas bestaan er geen algemeen geaccepteerde methoden om een oefening om te zetten in parameters. Daarom zijn concepten uit de dansnotatie Labanotation gebruikt om een nieuwe parameterisatie methode te ontwikkelen. Deze methode (beschreven in hoofdstuk 4) parameteriseert de oefening door eerst de relevante lichaamsdelen te definiëren, vervolgens wordt de oefening opgedeeld in segmenten van een specifieke duur. Voor elk segment worden de bewegingen van elk relevant lichaamsdeel omschreven in de termen: horizontale en verticale translaties en rotatie.

In hoofdstuk 5 wordt beschreven hoe een afspeellijst gemaakt wordt van een geparameteriseerde oefening. Het hoofdstuk beschrijft verder hoe de lichaamsposities, gemeten door de Kinect, vertaald worden in de termen van de parameterisatie methode (horizontale en verticale translatie en rotatie). Vervolgens wordt omschreven hoe een set van gemeten translaties vergeleken kan worden met een willekeurig element uit de afspeellijst. Als laatste wordt beschreven hoe het systeem de voortgang in de afspeellijst kan bepalen. Aan het begin van de oefening wordt de vergelijking natuurlijk gemaakt tussen het eerste element uit de afspeellijst en de eerste meetwaarden. Omdat het systeem niet altijd alle bewegingen juist zal detecteren, kan de afspeellijst ook doorlopen worden wanneer het merendeel van de bewegingen herkend is, of wanneer de huidige bewegingen overeenkomen met een later deel van de afspeellijst. Elementen uit de afspeellijst die niet herkend zijn, worden gemarkeerd om evaluatie van gebruikersfouten mogelijk te maken. In hoofdstuk 6 wordt de implementatie van het hierboven behandelde systeem beschreven.

Hoofdstuk 7 beschrijft het protocol en de resultaten van de experimenten die uitgevoerd zijn om de prestaties van het systeem in kaart te brengen. Helaas waren deze resultaten negatief. Het grootste probleem lag buiten het bestek van dit onderzoek, namelijk, de middelmatige lichaamsherkenning van de Kinect camera. Gebaseerd op alleen de herkenningskwaliteit, kan al gesteld worden dat de Kinect camera geen geschikt hulpmiddel is voor het automatisch herkennen en evalueren van onaangepaste rehabilitatieoefeningen.

Ondanks de negatieve experimentresultaten, bood de, op Labanotation gebaseerde, parameterisatie-methode een goede balans tussen een onwerkbare kwantitatieve notatie en een onduidelijke tekstuele notatie. De methode leent zich zowel voor het omschrijven van de oefeningen als voor het opslaan van de herkenningsresultaten. Dit maakt het vergelijken van de gebruikersuitvoering met de omschrijving eenvoudig.

Table of contents

1 Introduction

1.1 Physical therapy and rehabilitation

Every day, many people are limited in their activities of daily life due to a severe trauma. To regain functionality, or cope with the loss of functionality, intensive rehabilitation is required. Initially, the patient will be supervised at a rehabilitation center, but after 6 to 12 months, the patient will visit the rehabilitation center less frequent. In most cases, this does not mean rehabilitation is “finished”, the patient should continue to do exercises. Unfortunately, the lack of supervision and motivation (Jolly et al., 2007) while training at home, makes rehabilitation at home less effective.

The total costs of healthcare take up a larger percentage of the gross national product (GNP) each year, for example: 17% of GNP in the US today compared to 5% 60 years ago (“OECD Health Data 2010,” 2010). Therefore, instead of increasing the number of the visits to the rehabilitation center to increase the outcome, authorities are looking for ways to decrease the amount of visits, in order to save money. Telemedicine – the remote delivery of healthcare via ICT – is one of the promising ways to decrease the health care expenses without decreasing the outcome.

Since a few years, patients can do rehabilitation exercises at home, using telemedicine. For example via a web portal that shows them relevant training videos. But to be a good substitute for the face-to-face contact with the physician at the rehabilitation center, these telemedicine applications need to be able to provide direct feedback to the patient about their performance. To measure performance and give automated feedback, detection of posture and movement is required. There are many systems that can detect posture and movement of a patient, but they are too complicated to be used in a home setting, such as multi-camera tracking systems (Pastor, Hayes, & Bamberg, 2012), or lack specificity, like accelerometer-based systems.

1.2 Kinect

Recently, Microsoft released the Xbox Kinect, a depth camera that allows users to control a computer game via body movements and postures. Because the Kinect is very affordable and easy to setup, it could be an ideal tool for detection of posture and movement in a home-setting. The Kinect uses an infra-red projector / camera to measure depth, and is equipped with a normal video camera. Via the depth image, it can separate the subject from the background, which makes automated subject tracking and analysis much more reliable. To promote use of the Kinect outside of the gaming industry, software development kits (SDKs) for the Kinect are made available. These SDKs give access to the movement data of the people that are being tracked by the Kinect camera. The movement data is presented via the position in space of the main joints, in essence generating a moving “stick figure”.

1.3 Assignment

In this master thesis, research was done to find out whether the Microsoft Kinect is a suitable tool for automated exercise detection and evaluation.

The main research question is: **“How can the Microsoft Kinect camera be used for automated rehabilitation exercise evaluation in a non-supervised setting?”**

In this research question we define the "non-supervised" setting as a training location outside of the rehabilitation center, without the presence of professional care givers. For example, this can be at home, or in a community center.

Before this question can be answered, several sub-questions need answering:

- What are the pose and movement detection capabilities of the Kinect depth camera when tracking a single person in a non-supervised exercise setting?
- What type of rehabilitation exercises can be evaluated using a Kinect depth camera in a non-supervised setting?
- Which measurable body movement parameters can be used to evaluate the performance of an exercise that is part of a non-supervised training scheme for rehabilitation patients?
- How can the measured body movement parameters be automatically detected from the motion data recorded with a Kinect depth camera?
- How can the detected movements be compared to the intended exercise, in order to be able to evaluate performance?

This explorative research forms the starting point for an automated system that can provide exercise detection, performance evaluation and performance improvement feedback for many exercise types. As a proof of principle, the detection and evaluation components of this system are implemented for a single representative example exercise. For the implementation a generalized framework was developed that can be used to recognize and give feedback on various types of exercises. With the tools of this framework, new exercises can be entered into the system, without the need of rewriting the software. To enhance the exercise performance of the patient, a feedback loop is needed. This loop consists of the software giving feedback on errors made by the patient, and the patient acting on this feedback. In the proof-of-principle software, this feedback loop was not implemented. In a vision at the end of this thesis is shown how such an extension could be integrated into the framework.

1.4 Context and approach

From 2010 to 2012, Roessingh Research and Development (RRD), together with multiple partners, developed the "ConditieCoach" (CoCo, or "ConditionCoach"). CoCo is an ICT service for self-management of physical fitness of elderly and chronically ill patients. CoCo offers online individual exercise therapy via a web portal. This web portal consists of an individualized training program, illustrated by a set of relevant training videos, chosen from a database of over 200 training videos. Each exercise available in CoCo is accompanied by a short explanation.

The research to find out how the Microsoft Kinect camera can be used for automated rehabilitation exercise evaluation in a home setting is divided into tasks that relate to measurement and tasks that relate to analysis. The tasks are also divided into two stages: A = Preparation research and B = Implementation (see Figure 1 for an overview of the individual tasks and their order).

In the first stage (Chapter 2), the properties of the Kinect are researched (Figure 1: A1, A3), to find out which types of movements can be detected. For example, the Kinect application programming interface (API) does not include the finger joints, making it impossible to evaluate e.g. grasping exercises. Paragraph 2.1 discusses these properties of the Kinect. The next paragraph will discuss the evaluation of exercises, for example what measurable parameters could be used to judge exercise performance (A2, A4). The detailed properties of the Kinect (A1, A3) combined with information on evaluation of exercises (A2, A4) forms the basis for a set of rules that can indicate if the Kinect is a suitable evaluation tool for a specific exercise. In Chapter 3, these rules are applied on all exercises in the CoCo database, and one target exercise is chosen that is feasible and relevant to evaluate (A5).

The second stage of the research involves the design and the proof of principle implementation of the automated evaluation for the target exercise. To make sure the system can be extended to contain all feasible exercises from the CoCo database, a method is defined to parameterize the exercises (Chapter 4). This parameterization method is described, but not implemented. The data model underlying the method is implemented, and has to be specified manually for the target exercises (B1).

After the target exercise is parameterized, an algorithm is designed (Chapter 5) and implemented (Chapter 6) to detect the exercise parameters from the movement data (B3). This algorithm reads the exercise specification and compares the patient's performance to the specification (B2). The deviations between the measured performance and exercise specification are the input for the automated evaluation algorithm (B4). Augmented with metadata from the exercise specification, this algorithm can judge the impact of the errors made during the exercise performance.

To test the evaluation algorithms, several healthy persons have performed the target exercise, both correctly and with some deliberate mistakes (B5). These sessions are recorded with the Kinect depth camera and processed by the prototype implementation. Via a set of predefined performance indicators, the performance of the prototype is evaluated (Chapter 7).

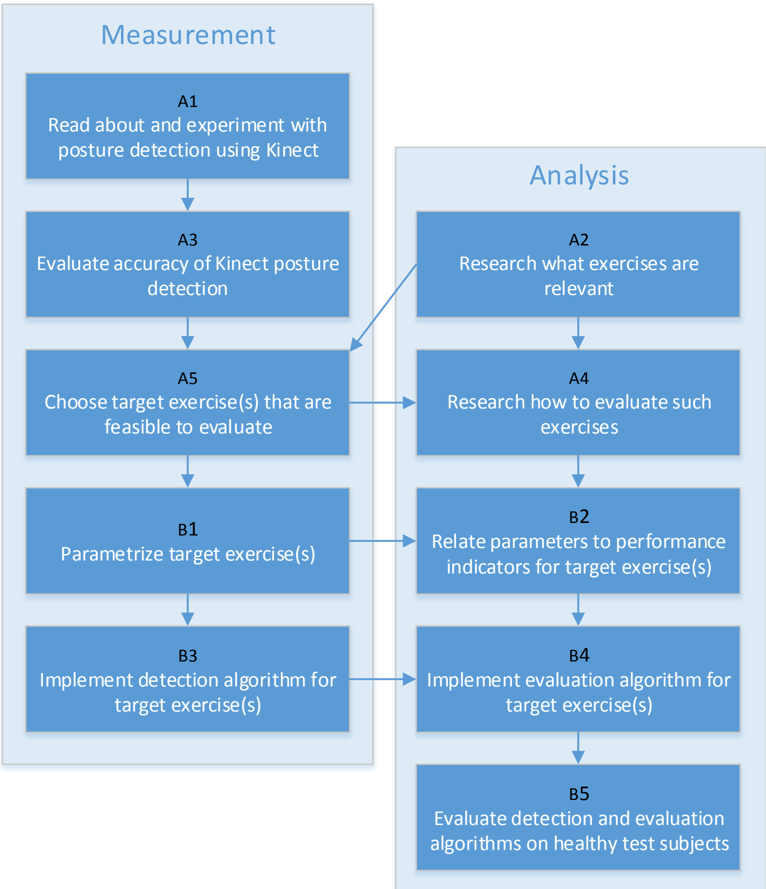


Figure 1: Scheme of the approach, items marked with A are related to the preparation research, and the items marked with B are related to the implementation of the prototype.

2 Context and background

In this chapter we present technical and practical information on the Kinect depth camera such as detection accuracy. Second, general information on exercises and evaluation of exercises is discussed.

2.1 The Kinect depth camera

The Kinect depth camera is one of the first widely available and affordable camera's that can detect depth, i.e. the distance from the camera to an object. Microsoft co-developed this camera together with PrimeSense (PrimeSense, 2011), to make a robust contactless user interface for their Xbox 360 gaming computer. The contactless user interface is offered by linking system actions to postures and gestures of the user. Therefore the posture and movement of the user need to be tracked. Conventional cameras can be used to track a user, but they are easily disturbed when there is no significant visual difference between the user and the background, i.e. a person in a grey sweater in front of a grey wall. A depth camera does not have this limitation, it can easily detect that the wall is further away from the sensor than the user, and in this way discern between the user and the surrounding objects. The depth information also greatly improves detection accuracy for limbs that are moving towards or away from the camera.

2.1.1 Technical properties of Kinect

Figure 2 shows a "see trough" image of the Kinect depth camera, with the major components marked. The *IR Emitter / IR Sensor* combo is used to measure the distance between objects and the sensor, the *Color Sensor* records normal video and the *Microphone Array* is used as an directional microphone, which can either sense the direction of a sound source, or "listen" to sound from a specific direction. Last, the sensor can be tilted 27° with use of the *Tilt Motor* to get the subject in view. Rotation is also possible, albeit manually.

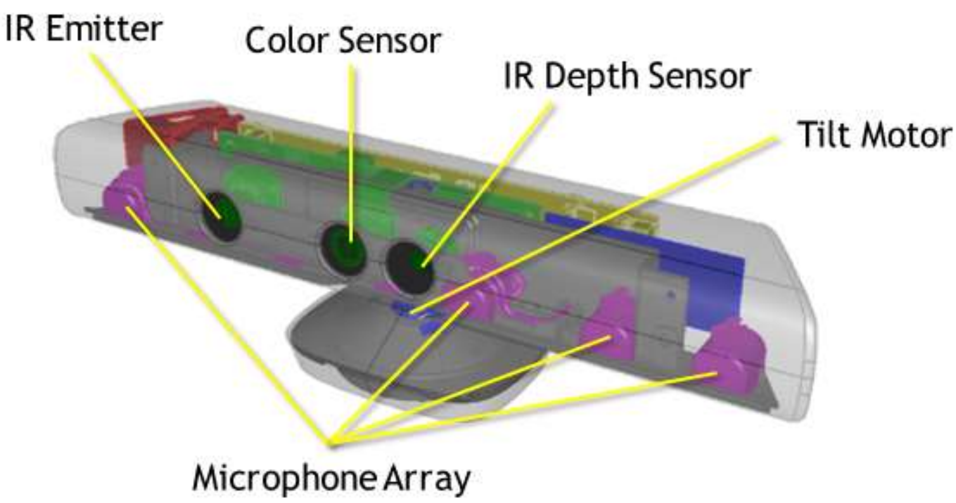


Figure 2: See trough image of the Kinect depth camera with the major features marked. (Microsoft, 2012)

Currently, three closely related devices are sold commercially, all compatible with the PrimeSense OpenNI software. PrimeSense sells their own camera, called the "Carmine", Asus sells the "Xtion", and Microsoft sells two versions of their Kinect. The "Kinect for Windows" and the "Kinect for Xbox". The Xbox version, as its name suggests, is only meant for use with Microsoft's Xbox 360 game console, whereas the Windows version is meant to be used with Windows PCs.

Even before the release of the “Kinect for Windows”, Microsoft released the Kinect Software Development Kit (SDK) (Microsoft, 2013a) which gives access to the raw image and depth videos, but also to the pose and movement data that is extracted from those raw videos. The Microsoft Kinect SDK can be used together with the “Kinect for Xbox” as well, but not with the PrimeSense Carmine or Asus Xtion. For those sensors, PrimeSense released the “Natural Interaction” SDK (Currently: OpenNI 2.0). This SDK works with both versions of the Microsoft Kinect as well, even though Microsoft officially does not support the use of OpenNI with their sensors.

Table 1 lists the main specifications of the three closely related sensors. The relevant differences are:

- The Kinect needs an external power supply
- The Kinect has an microphone array to detect the direction of a sound source
- The Asus Xtion does not have a normal camera (only depth)
- The Kinect for Windows and Carmine support the “near” mode, which changes the range from 80cm-4m to 40cm-3m.

	MS Kinect for Xbox	**MS Kinect for Windows**	**PrimeSense Carmine 1.08**	**PrimeSense Carmine 1.09**	**Asus Xtion Pro**	**Asus Xtion Pro Live**
Release date	Nov 2010	Feb 2012	Aug 2012		Apr 2011	Jul 2011
Intended use	Gaming	Commercial, consumer	Commercial, development		Development	
Range	80cm-4m	40cm – 3m	80cm – 3.5m	35cm – 1.4m	80cm – 3.5m	
SoC	PrimeSense PS1080-A2					
Introduction Price	$ 150	$ 250	$ 200		$ 190	$ 270
Resolution / Frame rate RGB	1280x960 / 12fps 640x480 / 30fps		1280x960		n.a.	1280x1024 / 30fps
Resolution / Frame rate depth	640x480 / 30fps		640x480 / 30fps 320x240 / 60fps			
Accelerometer	3-axis, 2G range, 1° resolution		n.a.			
Automatic Tilt	1-axis, ±27°		n.a.			
Field of view	43° vertical 57° horizontal		45° vertical 58° horizontal			
Audio	4 microphones, 16KHz		2 microphones			
Power use	12 watt (External PSU)		2.5 watt (USB Powered)			
Dimensions	30.5 x 7.5 x 6 cm		18 x 3.5 x 5 cm			
Weight	1.3 kg		0.3 kg			
SDK	MS Kinect SDK / OpenNI + NITE		OpenNI + NITE			

Table 1: Comparison of the different PrimeSense based depth sensors (sources: (Asus, 2012; IFixit, 2011; iPiSoft, 2013; Microsoft, 2012; PrimeSense, 2011, 2012)).

The depth images received from a structured light 3D-scanner such as in the Kinect camera are the result of an algorithm that performs dense 3D image acquisition using structured light with a pattern of projected infrared points. The deformation of a speckle pattern projected on the scene, with respect to a reference pattern, reveals information about the distance of the objects and results in a calibrated depth mapping of the scene (Elteren & Zant, 2012). Figure 3 shows the world, seen through the IR sensor of the Kinect. The speckle pattern is analyzed in the Primesense processor integrated in the sensor, to create a depth map of the whole image. For each point, the distance between that point and the sensor is stored and sent to the PC. The unaltered infra-red and color videos, and the audio streams are also sent to the PC. The latency of these streams, including the depth map, is roughly 45ms (PrimeSense, 2011). All the streams together nearly fill the bandwidth of the USB 2.0 interface. Therefore only a single sensor can be connected to an USB controller (most PCs have multiple controllers) and recording / processing of the streams generates a high load on the PC.

Figure 3: Dot pattern as seen by IR camera on Kinect (left: full frame, right: detail of pattern).

2.1.2 Software Development Kits

Currently, there are two SDKs that enable skeletal tracking using the Kinect: MicroSoft's own Kinect SDK, and PrimeSense's OpenNI + NITE. Other markerless motion tracking software packages exist, but these must be trained for a specific use case, such as OpenCV, or require a multi-camera setup, such as Organic Motion OpenStage.

The Microsoft Kinect SDK and OpenNI + NITE are made for the same purpose: tracking a skeleton using a depth camera based on PrimeSense technology. Compared to OpenNI + NITE the Microsoft Kinect SDK does have some advantages and downsides (see Table 2):

Microsoft Kinect SDK	PrimeSense's OpenNI + NITE
Closed source	Open source
Fully supported in C++, C#, partly in Visual Basic	Fully supported in C++, partly supported in C#
Windows only	Windows, OS X and Linux support
Tracks persons without requiring an initial pose	Requires "initial pose"
Complete and up-to-date documentation	Good documentation for OpenNI, but NITE documentation is outdated
Only functions with Kinect, and forces use of "Kinect for Windows" sensor for executables	Works with all Primesense based sensors
Tracks up to 6 persons, but only the first two have a complete skeleton	Fully tracks 6 persons
Tracks up to 20 joints	Tracks up to 24 joints

Table 2: Comparison between the MS SDK and Primesense OpenNI / NITE software.

In terms of accuracy of the skeleton tracking of a single person (the usecase in this project) the differences between the two software packages are minor. Although the initial pose, holding both hands in the air, required by the OpenNI + NITE software, can cause serious problems for rehabilitation purposes. For example, many CVA patients will have serious issues striking the initial pose, due to hemiplegia (Pastor et al., 2012).

In research there is a bias towards using open-source software. This means that most research projects used the Kinect together with OpenNI and NITE software. Thus OpenNI / NITE enables to take advantage of research projects of which the source code was made public. Unfortunately, these research projects all use C++, which is not very suitable for unexperienced programmers. Especially because the documentation of OpenNI / NITE is less coherent and up to date than Microsoft's Kinect SDK documentation. Support for the relatively easy-to-learn C# language, and the better documentation were the main reasons to choose the Microsoft Kinect SDK for this project.

2.1.3 Skeletal Tracking

For the skeletal tracking to work reliably, the full body has to be in the field of view of the Kinect camera. The relatively narrow vertical field of view of 43° greatly limits the area in which the user can move around.

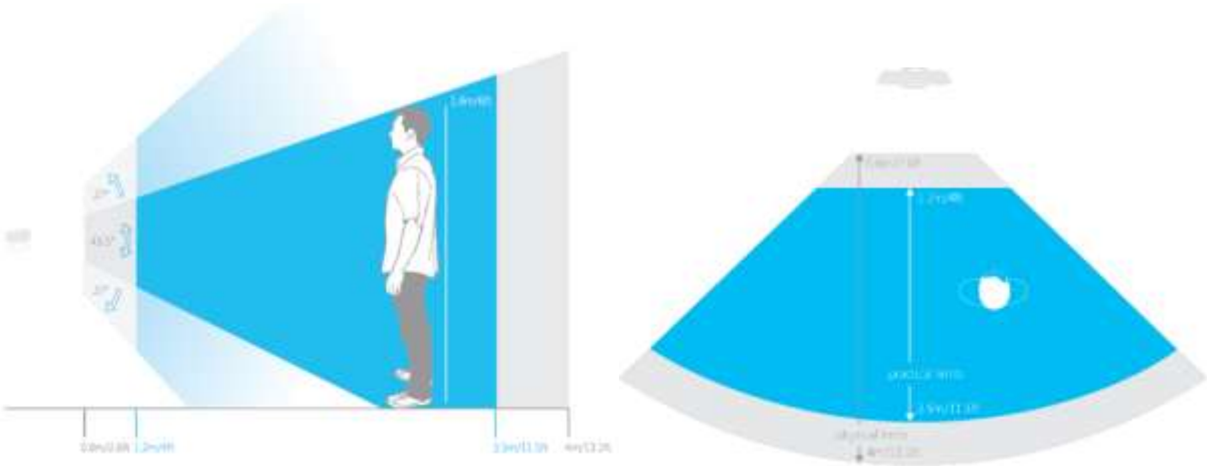


Figure 4: Working range and field of view of a Kinect depth camera. As can be seen in the left view, the movement range towards or away from the sensor is very limited. The right image shows that the movement range to the left or right is much larger (Microsoft, 2013a)**.**

Figure 4 clearly shows this limitation. In the picture on the left, the dark shaded area represents the area in which reliable depth data is available. The person, 180cm in length, is standing as close to the sensor as possible. Nevertheless, he can only move one step back, before he is standing too far from the sensor. The right image shows that the horizontal plane allows for free movement. Unfortunately, the software cannot track a person not facing the sensor, thus the Kinect is not suitable for free walking exercises. When the person is detected and the skeleton is tracked, the position of the joints listed in Table 3 and depicted in Figure 5 are available.

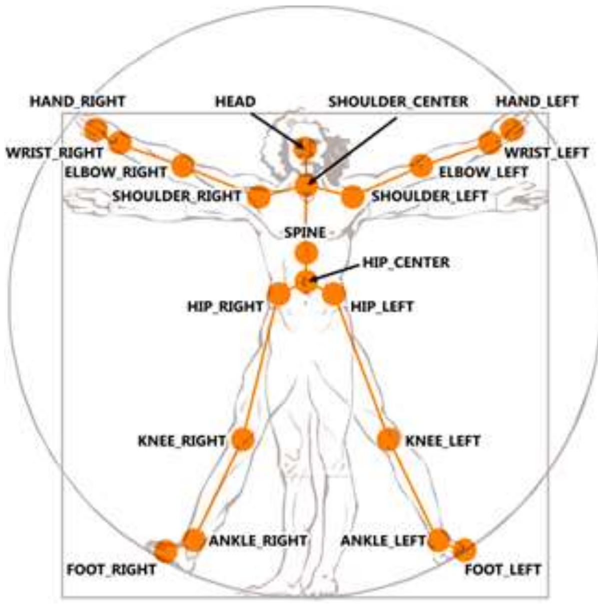


Figure 5: MS Kinect SDK joints (Microsoft, 2013a).

Member name	Description
AnkleLeft	Left ankle
AnkleRight	Right ankle
ElbowLeft	Left elbow
ElbowRight	Right elbow
FootLeft	Left foot
FootRight	Right foot
HandLeft	Left hand
HandRight	Right hand
Head	Head
HipCenter	Center, between hips
HipLeft	Left hip
HipRight	Right hip
KneeLeft	Left knee
KneeRight	Right knee
ShoulderCenter	Center, between shoulders
ShoulderLeft	Left shoulder
ShoulderRight	Right shoulder
Spine	Spine
WristLeft	Left wrist

Table 3: JointType Enumeration (MS Kinect SDK).

The Microsoft SDK has two states: recognized and tracked. Up to 6 persons can be recognized, these 6 get a unique ID, and a location of the Hip Center joint. If a person re-enters the scene, the old ID is coupled to this user. This recoupling of the old ID is not guaranteed to work reliably, thus for person identification other technologies should be used, for example the SHORE project by Fraunhofer (Ruf, Ernst, & Küblbeck, 2011). Up to two persons can be in the tracked state, for those two, the full set of joint positions is given, including the orientation of the bones in between the joints. When the OpenNI + NITE software is used, 2 extra bones become available: Collar Left and Right. The Collar bones are in most cases redundant to the Shoulder joints, but could be useful to track movements in which the torso remains static, but the shoulders move, for example when moving the shoulders forwards.

Figure 6: Joint orientation information hierarchy, the properties of a bone are stored in the parent, which is displayed towards the left.

Positions and orientations of joints can be given in two ways: hierarchical and absolute. The absolute representation uses the global Kinect camera coordinates (y-axis is upright, the x-axis is to the left, and the z-axis faces the camera). Hierarchical representation gives orientation relative to the parent joint. The Hip Center joint is highest in this hierarchy, the full tree is given in Figure 6, and an example is given in Figure 7.

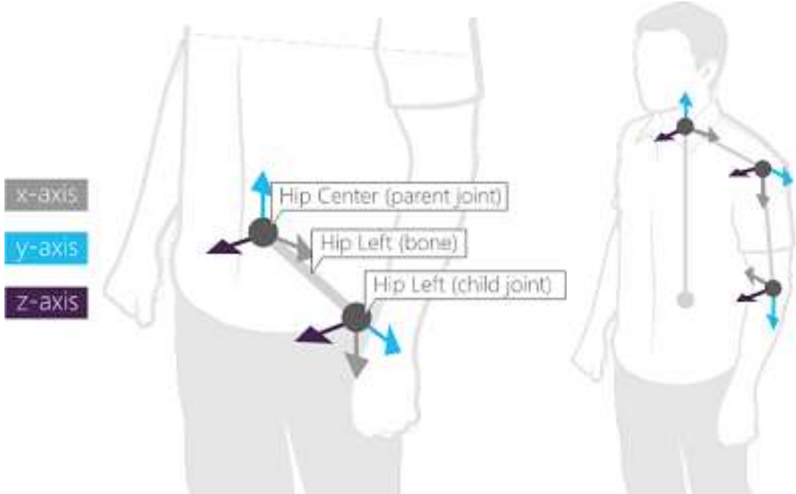


Figure 7: Schematic view of the relative bone and joint orientations. Mark that the orientation of the axis differs per joint (Microsoft, 2013a).

Next to joints, the orientation of the bones in between the joints is given. Bone rotation is stored in a bone's child joint. For example, the rotation of the left hip bone is stored in the Hip Left joint. The rotation of bones is used extensively for avateering: creating a virtual textured character that follows the movements of the tracked person.

2.1.4 Detection accuracy and capabilities

The detection accuracy of the Kinect should be looked on in two ways, first the sensor has a certain technical accuracy, limited by the technology chosen. And second, the accuracy of the human motion detection greatly depends on the optimization of the advanced software that converts the raw sensor data to moving stick figures.

Technical accuracy

As stated in the previous chapter, the Kinect sensor has a 1280x1024 RGB sensor, and a 1280x1024 infrared sensor (Khoshelham & Elberink, 2012). Both can record with a frequency of up to 60 frames per second, but due to bandwidth limitations, this frame rate can only be achieved at reduced resolutions. At the "default" frame rate of 30 fps, both RGB and depth cameras output a 640 x 480 pixels image. This sensor resolution corresponds to a theoretical effective resolution of ca. 2 millimeters for objects nearby, to a maximum of 4 cm at the maximum distance (Khoshelham & Elberink, 2012). Obviously, this theoretical resolution is limited by optical imperfections. The lens is not perfect, and shows some distortion, roughly 1.5% at the far corners.

For the depth image, the relation between the sensor resolution and the resolution of the resulting depth image is not straightforward. The depth image is the result of a triangulation process in which the shift and scaling of the observed infrared speckle pattern is calculated. The speckle pattern is observed with the infrared camera. Multiple pixels are needed to "know" the shift and scale of the pattern. This means the resolution of the depth image is much lower than the resolution of the infrared camera. How much lower, depends on several factors:

- The amount of infrared light naturally available at the scene

Infrared light present at the scene, lowers the contrast of the speckle pattern, making it more difficult to detect. Thus it is wise to avoid direct sunlight on the scene.

- The reflection properties of the objects in view

Both materials that reflect IR light in a distorted fashion (like a glass bottle), or do not reflect at all (like a furry carpet), severely hamper the depth detection accuracy (Dutta, 2012). For most fabric types, and for human skin, this is not an issue.

- The position of the objects in the viewpoint

The accuracy of the Kinect depth image is better when the object to be tracked is placed in the center of the frame. This has multiple reasons, the two most important are: the optical distortion is increased at the edges of the frame, and second, the angle of the projected IR beams is smaller at the edges, decreasing the chance on direct reflection. Indirect reflection (scattering) either decreases the amount of IR light that reaches the sensor, or worse, interferes with the IR patterns from other objects.

- Objects casting shadows

Shadows are a problem for the structured light depth detection principle. Because the IR projector and sensor are not at the same physical position, objects cast two shadows. In Figure 3 (page 15), the person is holding a pen. Next to the pen, at the right side, a black shadow of the pen can be seen. This spot is where the structured light was blocked by the pen, and obviously no depth information is available. The second shadow is not visible in this image, because it's the area directly behind the pen. This area received the structured light, but could not reflect this to the camera, because the pen was blocking the path to the IR sensor. The result of this shadowing, is a "halo" around objects that are closer to the sensor. It can be detected that this halo is not part of the object, but its depth information is missing for this halo. These shadows, combined with the low resolution, result in a poor accuracy of small objects (Dutta, 2012).

Park et al. have looked at the accuracy of the Kinect depth camera in great detail (Park, Shin, Bae, & Baeg, 2012). Their "uncertainty ellipsoid map", shown in Figure 8, is an illustration of how the accuracy deceases further away from the sensor. The ellipsoids are much larger for a larger *Z*. Away from the center, the ellipsoids are wider as well, but this effect is less pronounced.

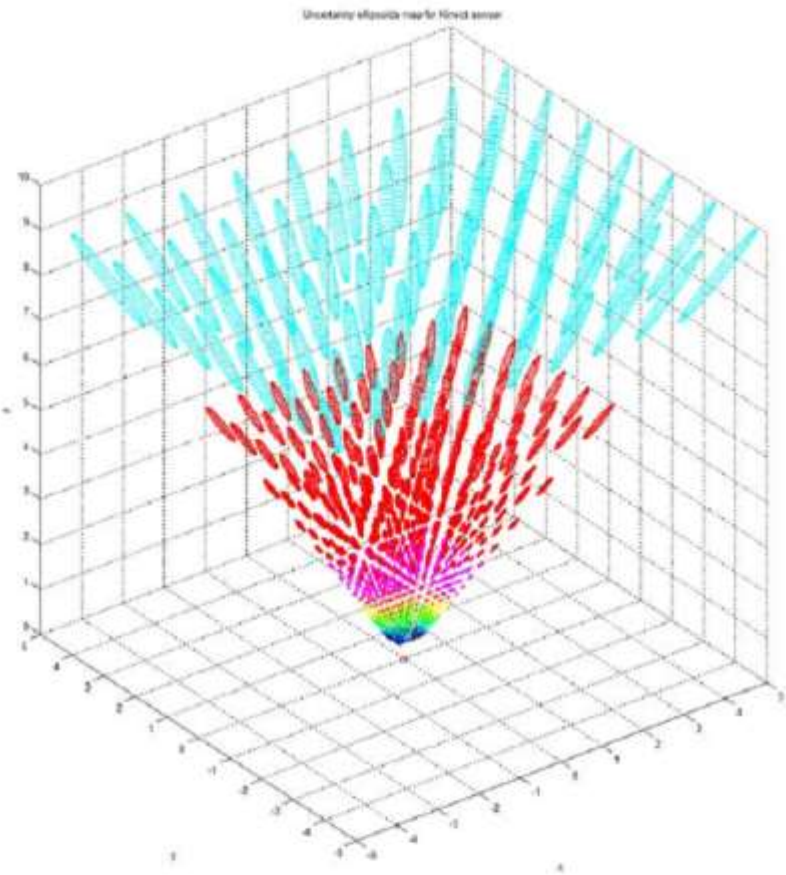


Figure 8: Uncertainty ellipsoid map in the entire measurable Cartesian space (Park et al., 2012)**.**

Practical accuracy

For the detection framework discussed in this thesis, the raw Kinect Depth data will not be used. The framework uses the skeletal movement data generated by the Software Development Kits. As described in the introduction, the software bundled together with the Kinect, uses the depth data to generate a moving stick figure of the persons in view. Because the depth data is the major input for the human movement detection, its accuracy is still relevant. As stated in the previous paragraph, the depth resolution is much lower than the horizontal / vertical resolution. This reflects on the accuracy of the movement model. This model is most accurate when the user moves within a vertical plane, parallel to the sensor, as nearby as possible while keeping the full body in the field of view of the sensor. The movement model accuracy is hampered when:

- Detailed depth data is needed

A person that is tracked with the Kinect doesn’t need to be standing in a plane parallel to the sensor, because the depth image can be used to calculate the angle between the plane and the sensor. But the more the person is standing perpendicular to the sensor, the narrower its silhouette becomes, greatly reducing the accuracy of the movement data.

- Body parts are occluded

Depth data can also be required when body parts move in front of each other. For example, when the tracked person moves his hand in front of his torso, the Kinect SDK will be able to track this, if the distance between the hand and torso is large enough (approximately 5 cm).

Even when the depth data is accurate enough, there are situations in which this data is of little use: when body parts are too close to each other, or when the silhouette is not clear.

- Body parts are joined

When two body parts are so close to each other, that there is no detectable gap, tracking of these body parts is severely hampered. In most cases, the software will try to guess the positions of the body parts which are joined. But the algorithm is easily fooled, for example by moving your arms from above your head, downwards along your body, and then moving them further such that eventually your left arm is on the right, and your right arm on the left. The software will have a hard time detecting this movement, and might conclude incorrectly that your arms have become much shorter, but that the left arm is still left and vice versa. It's hard to create a workaround for these false detections, because the algorithm is very unpredictable in these edge cases.

Figure 9: man in cape

- Silhouettes are vague

By far the most important input for the skeletal movement detection, is the silhouette. If this silhouette does not resemble a human being, detection will fail. Silhouettes get obscured when the user is wearing very loose clothing, for example the man in Figure 9 cannot be tracked reliably because the silhouette of his arms is obscured by the cape.

Silhouettes are also obscured when the user is holding a large object. The software then must decide whether this object is foreign or part of the body, but is incapable of doing this reliably and consequently. A way to circumvent this limitation, is to use opaque objects. This has been done by Pastor et al.; the authors used a transparent table, in order to let the patients rest their hands on the table, without interfering with detection accuracy (Pastor et al., 2012).

The normal modus of the Kinect SDK relies on the silhouette for skeleton tracking. This only works if the person is standing at some distance away from other objects. In the "seated modus" of the MS Kinect SDK (version 1.5 or later) (see Figure 10), the software relies on movement, and is thereby able to discern between the moving person and a static chair (Microsoft, 2013b). In this seated modus, only the arms, shoulders, neck and head are tracked. Another difference from the normal modus, is the type of initiation. Normally the MS Kinect SDK will start tracking an object that resembles a human, even if it remains static. In the seated modus, the object has to move before it will be recognized by the Kinect SDK.

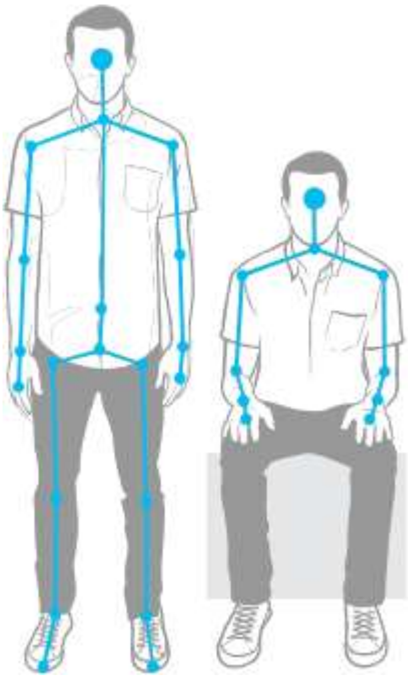


Figure 10: Normal and seated tracking modus, showing 20 compared to 10 joints (MS Kinect SDK (Microsoft, 2013b)**).**

Latency of the skeleton model greatly depends on the processing power of the PC. The raw image stream has a latency of ± 45ms, whereas the skeleton latency ranges from 100 to 200ms, depending on the resolution and number of tracked persons, with peaks up to 500ms (Livingston, Sebastian, Ai, & Decker, 2012).

The Kinect Skeleton Tracking incorporates 20 joints to represent the human movement. The number of joints in a real body is much larger. Some significant omissions are:

- Lack of fingers

  The Kinect model only tracks the wrist and hand, no fingers. It can detect a hand "grip", which can be used to grasp / drag something in a virtual interface. (See Figure 11)

- Only three joints represent the spinal cord

  Because the spinal cord is represented by a fixed set of joints, realistic bending of the back is not possible.

Figure 11: Hand Grip (Microsoft, 2013a)**.**

- Facial expressions are neglected

  Eyes and mouth are not part of the skeleton model, omitting a large part of the normal human interaction. Since version 1.5, the Microsoft Kinect SDK has a separate "Face Tracking" module, which analyses the 2D position of 87 points of the head which can be used to generate a virtual face mask. This functionality is not used for this project.

- All joints are simple ball and socket joints

In reality, some joints, such as the shoulders, are complex groups of joints that allows much more types of motion than a ball and socket joint. Chang et al. have shown that quality of tracking the hand and elbow is much higher than tracking of the shoulder movement (Chang et al., 2012).

2.2 Automated posture and motion detection methods

The Kinect SDK used in combination with the Kinect depth camera determines orientation and position of 20 joints. With this data, a realistic representation of human movement can be given. But interpretation of this movement is not straight-forward. To interpret motion, recognition of postures and movements is essential. The moment in time and the context in which these postures and movements are performed, determine the meaning of these movements. For example, for a system that is controlled via gestures, the interpretation system needs to be able to discern reliably between all available gestures, and needs to be able to detect the moment at which these gestures were performed. In context of this thesis, the system does not need to be able to discern between all movements from all exercises, because it is known beforehand which exercise the patient is about to perform. However, it is essential to be able to detect if the sequence of the movements was correct. Contrary to gesture detection, for the exercise detection and evaluation, it is essential to be able to detect movements that were performed incorrectly, and detect what the patient did instead of the correct movement. Without information on incorrect movements, it is not possible to give feedback to the patient about what he or she has done wrong.

Automated recognition / interpretation of motion can be divided into two methods: Learning and Parameterization.

- Learning

As the name says, a “learning” recognition system, “learns” itself. For this, it needs a **reference** set. To learn a system to be able to discern between 20 gestures, it needs to “see” at least one performance of each gesture. When it is then presented with a new recording, it determines which of the reference performances comes closest to the new performance. This matching can be done by searching for cross-correlations (Chang et al., 2012) between the new recording and each reference recording. When the number of reference recordings grows, this will take a significant amount of processing power. For large reference sets, methods like Hidden Markov Models (Brucker, 2012) and Neural Networks give a much better performance than a linear cross-correlation search. Large reference sets are important to get robust recognition. If only a single reference performance is available, a match can only be made if the new performance is very similar to the reference performance. Similar not only in movements and timing (signal), but also in all other properties, such as the posture of the performer (noise). Increasing the number of reference recordings of the same performance, increases the variance in properties which are not relevant to the performance (noise), but does not increase the variance in the performance movement (signal), thereby increasing the “signal to noise” ratio (as long as all performers perform the movement correctly!). With a higher number of reference recordings, the system is able to recognize the performance longer despite an increased number of random artifacts.

Before a new recording can be fed into a learning system, it has to be normalized. By normalizing, properties that are specific for a certain user, can be removed. Common ways of normalization are in time and dimension. For normalization in time, the reference and new recordings are resampled such that their duration is equal. In this way, the performance speed has no influence on the detection. For normalization in dimension, the reference and new recordings are scaled such that for example the height of the performer is *one*, this makes recognition robust for performers with different height. Normalization of the orientation is another common form of normalization in dimension, in which the recording is rotated such that each user's body makes the same angle to the camera, ruling out differences in the global orientation. What aspects can be normalized, depends on the purpose of the learning system: after normalization in time, recognition of a movement that was performed too slowly, is no longer possible.

The output of a learning system can only be the quality of a match with one or more reference recordings. This means that for every feature that has to be recognized, one or more dedicated reference recordings are required. To detect a perfect performance of an exercise, the number of required reference recordings is limited, but to be able to evaluate the performance, a much higher number of reference recordings is needed. The higher number of recordings is needed because evaluation not only requires to recognize what went according to plan, but also what went wrong. Thus of each error that the system should be able to evaluate, one or more recordings needs to be present in the reference set.

- Parameterization

An alternative to automated learning, is to parameterize the movements. The parameters can either describe static postures or dynamic efforts resulting in movement. A description of a series of static postures describes a movement by identifying the position of several body parts at known intervals in time. The movement in between these defined postures (also called "key frames") is not defined. Instead of describing static postures, a movement can also be described by identifying the changes between the postures at known intervals. The parameters then define the effort needed to go from one posture to another. For example, moving hands above the head in a static parameterization will describe two static postures, the first with the hands along the body, and the second with the hands above the head. The effort based parameterization of the same movement will only have one step: move hands upwards. In most cases it's not practical to use an exclusively effort based parameterization method, because it lacks an initial posture. Without a defined starting point, the result of any effort is undefined as well. Another issue with an exclusively effort based parameterization can be drift. If the effort based parameterization contains many steps, and in each step a small error is made, the end result of the whole movement can differ significantly from the intended movement. The static postures do not require input from a previous step, and therefore maintain their accuracy. By adding a static initial posture to an effort based parameterization method, any kind of movement can be described fully. Static postures (key frames) can also be added at longer intervals to deal with the drift, at the cost of increased complexity.

Compared to the learning systems, parameterization has the substantial advantage that it is context aware. If a certain parameter describes that the hands move upwards, and this movement was not recognized, it can be concluded that the hands did not move upwards. Whereas when a learned reference was not recognized, little can be concluded, because it was not known what the meaning of the reference was. For evaluation, being able to recognize errors made by the patient is essential. Parameterization is much more suited to do this, and is therefore chosen as method for the design in this thesis.

This is not to say parameterization doesn't have downsides. The most difficult aspect of this method is to create a suitable set of parameters. There is no straightforward method to define any sort of human movement in a structured parameterized way. Chapter 2.3 will go into detail on this topic. Even though, creation of the parameters is very complex, it only has to be done once. Contrary to the reference sets for the learning system, parameters can be adjusted. Patient-specific aspects can be taken into account, for example by reducing the range of motion of a certain joint. Such alterations would not be possible using a learned reference recording.

2.3 Notation of human postures and motion

"Teaching" exercises to a computer can be done in two ways, via automated learning, or via explicitly defining the parameters. For this project, the latter is most relevant, because it can easily be extended into an automated evaluation system. Unfortunately, the literature on systems that have implemented a way to easily add new exercises to the system, fail to explain how they implemented this. They just mention that there are "interactive tools for assisting the therapist with creating new exercises" (Camporesi, Kallmann, & Han, 2010), or give a method without context: "Accumotion recognition algorithm is based on multiple kinematics evaluation functions based on taking the dot products of a target bone position and the user bone position" (Fujimura, Kosaka, & Robert, 2012).

In many papers there is information on the modalities that are taken into account while determining the parameters needed to describe the exercises. For example, Jack et al. use range, speed, fractionation (independence of (finger) movement) and strength of movement (Jack, Boian, & Merians, 2000) to describe the movements. To learn the "Reactive Virtual Trainer" new exercises, Van Welbergen and Ruttkay have developed a method in which a specific path in time is described for each "key body point" (Welbergen & Ruttkay, 2008). This combines both position and speed accuracy into the evaluation. This concept is well visualized in their paper (Figure 12).

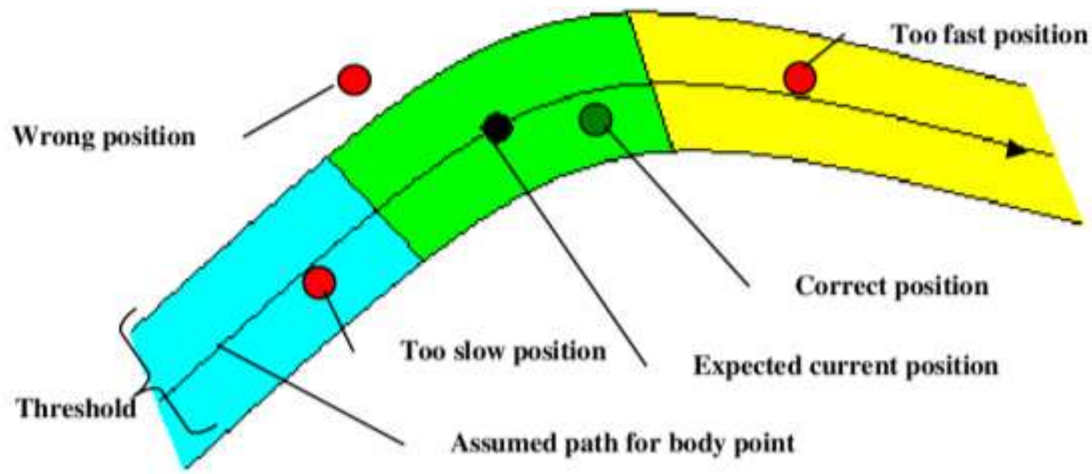


Figure 12: Assumed motion path of a body point, with expected, early, late and wrong positions (Welbergen & Ruttkay, 2008).

For their exercises, the "key body points" were the four extremities: hands and feet. For the repetitive but simple exercises they targeted, this was sufficient, but for the much more complex set of exercises present in the CoCo database, more “key body points” (or more appropriate for the Kinect: key body **joints**), need to be defined. The number is practically limited by the 20 joints available in the Kinect SDK Skeleton model.

A few publications exists in which a universal movement notation is developed, specifically aimed at evaluation of exercises (Lu & Jiang, 2013; Ukita, Kaulen, & Röcker, 2014). Unfortunately, these publications were published after this stage of the research was completed.

2.3.1 Dance notation

In contrary to the world of rehabilitation, in dance and music, extensive notation “languages” exist. One of the first successful attempts for a universal “dance notation”, was the Labanotation developed by Ann Hutchinson Guest (Guest, 1977) based on the Laban Movement Studies by Rudolf Laban (1879-1958). Despite being one of the more successful notations of today, neither Labanotation, nor any other dance notation can be called a “standard”, such as the staff notation used to write down music. Dance notations are not popular because they are not intuitive in use (Kahol & Tripathi, 2006), and complex to learn. The complexity is evident, when looking at a small part of the Labanotation of the “Autumn Quartet”, in Figure 13.

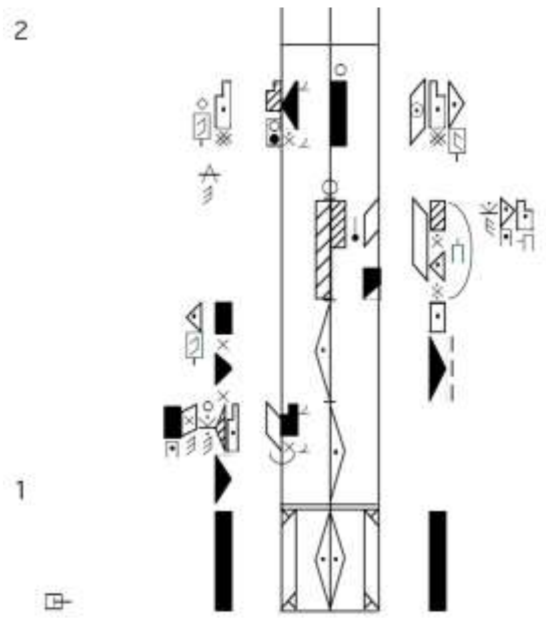


Figure 13: Start of the "Autumn Quartet" (Extract from Wordpress blog by Michael J. Morris).

Despite the complexity, Labanotation, or its derivatives, “Kinetographie Laban” and “Motif”, are relevant to this research, because it is one of the very few standardized ways to describe motion, that is compatible to every kind of dance, and as such also for almost any type of movement. Labanotation and the movement analysis rationale behind it, has been used for experimental research in revalidation therapy (Foroud & Whishaw, 2006).

Labanotation describes movements by describing the effort and movements that are needed to get to the desired posture. This is essentially different from the “keyframe animation” that is commonly in use for human movement animation and analysis on a computer, in which only the end positions are described.

Labanotation symbols are placed on a staff, and read from the bottom to the top. The center of the staff represents the transference of weight of the body, and to the left and right, the left and right parts of the body are represented. The transference of weight column records every change in the center of weight, including which body part carries the center of weight (usually the legs). The columns for transference of weight, legs, body, arm and head are always drawn, more columns can be added when required, for example, a column for the feet is added when a foot should make a movement that is not logical in respect to the movement of the legs (e.g. turning them outwards).

The length of the staff is directly related to time. Thus when a movement takes much time, the symbol will be stretched over a large part of the staff.

To indicate in which horizontal direction a movement takes place, a basis of 9 directions is used: Place, Forward, Backward, Left, Right, Left forward, Right forward, Left backward and Right backward (see Figure 14). Three vertical directions are discerned: Up, Middle and Down. These are indicated by the shading of the symbol for the horizontal direction. If required, a more detailed direction can be given by adding “pins”. These are particularly useful to indicate a single body part has a movement relative to another body part, instead of relative to the body as a whole.

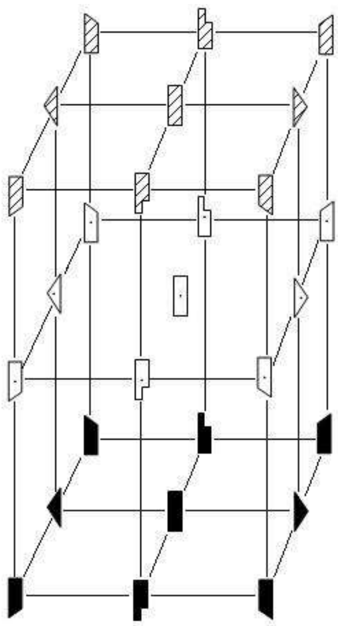


Figure 14: Labanotation direction symbols (Griesbeck, 1996).

When a direction symbol is placed in any column other than the “center of weight” column, it indicates the movement of that body part is relative to the point of attachment. See Figure 15 for a visualization of the arm movement and respective symbols.

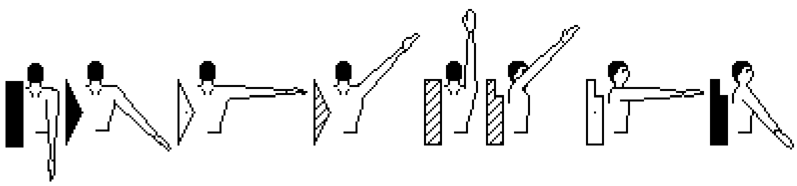


Figure 15: Arm gestures and the direction symbols (Griesbeck, 1996).

Via symbols in the “center of weight” column, five situations can be indicated:

1. Hold: nothing changes, can be indicated by a dot.
2. Shift: weight carrying body parts do not change, but center of weight does, for example by bending the knees to lower center of mass.
3. Transfer: change weight carrying body part, for example during walking. Switch to another body part can be indicated by the adding the logo of that body part to the “center of weight” column (see Figure 16 for the logos).
4. Jump: while in the air, no body part carries the weight, and the “center of weight” column is empty.
5. Turn: turns are indicated via skewed rectangles. Most turns take place around the vertical axis.

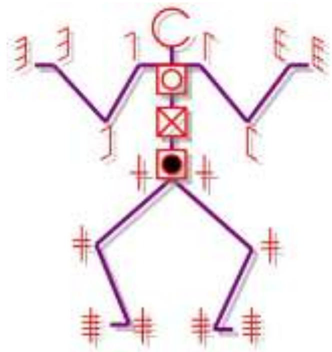


Figure 16: Labanotation: Signs for parts of the body *(By Huster via Wikimedia Commons).*

The position and length of the direction signs indicate the quantity, but Labanotation also allows indication of the quality of a movement. For example, a cross indicates that a movement should be made in a shortened or contracted way. In six levels, the amount of contraction can be indicated. Other space measure qualities are: extension, folding, unfolding, joining and spreading. Next to the space measure, quality can also be indicated via accents, for example: weighty, gentle, strong, relaxed, emphasized, etc.

If independent movement of multiple body parts should take place simultaneously, this is indicated via a large vertical bow, joining all symbols of the simultaneous movement.

The last group of symbols consists of paths and floor plans. For example to indicate that the whole body is moving in a continuously larger circle. The floor plans are essentially a map indicating the movement of the whole body. These are particularly useful if interaction between multiple people takes place.

Use of Labanotation to describe exercises

If the Labanotation is to be used to describe rehabilitation exercises, the physician entering the exercises would need more information on the notation than given in the previous sections. However, it is not realistic to expect a physician to become a master in the Labanotation before he or she can use the system. Because the Labanotation can capture virtually any movement with this limited set of symbols, the notation does indicate what modalities are considered of importance when a movement has to be captured on paper. Without using the Labanotation symbols, it is still possible to extract the same information from an exercise description, and as such use the concept of the language, without the language specific syntax. This would mean, first start by defining the timing of the exercise. A new time segment starts when movement is static. Then analyze the direction of the “center of weight” in time. Next decide which body parts perform specific movements that need separate notation. Divide this motion into horizontal and vertical translations or a rotation. And determine the duration of each movement.

Along with the description of the exercise, there must be room for metadata to personalize the exercise. For example, the required movement speed can be dependent on the age of the patient. For evaluation, it is also important to define the restrictions on movements. Errors in movements of certain body parts might have much lower impact than errors made in the movement of other body parts.

3 Selection and detailed analysis of exercise

3.1 Introduction

The previous chapter gave information on the capabilities of the Kinect depth camera and discussed ways to parameterize a movement via the Labanotation. In this chapter, a set of exercises is presented, which are all suitable for unsupervised home training. The knowledge on the technical capabilities of the Kinect is applied on the set of exercises to give an indication of the types of exercises for which the Kinect depth camera is potentially a useful tool to perform automated exercise detection and evaluation. In the last sections a single target exercise is chosen, to support development and testing of the actual detection and evaluation system. This exercise is described in detail.

3.2 Available exercises

To get a good view of the type of exercises which are suited to be included in a home training program, the exercise database behind the home rehabilitation system “ConditieCoach” (CoCo) is analyzed. CoCo is developed by RRD, together with multiple partners, from 2010 to 2012. In these years, over 500 patients used CoCo as an experimental addition to their rehabilitation program.

CoCo consists of three parts (Tabak et al., 2013):

1. Activity monitoring by use of a Smartphone and movement sensor
2. Online individual exercise therapy
3. Telemonitoring and feedback

Part 2 consists of an individualized training program, illustrated by a set of relevant training videos, chosen from a database of over 200 training videos. This database of training videos forms an excellent basis to find out what type of exercises could be evaluated with a Kinect depth camera.

CoCo is divided into four main “care paths”: COPD, Acute Hip (hip surgery after trauma), Conservative Hip (planned hip surgery) and Oncology. For each path a specific set of exercise videos is available, but a single exercise can be part of multiple paths. Each path contains exercises in multiple categories, for example: thorax mobilization, relaxation and breathing techniques. Each exercise available in CoCo is accompanied by a short explanation (see the screenshot in Figure 17). This explanation text contains roughly the same information as spoken by the “actor” in the videos. Each explanation contains the same set of sections: *purpose*, *performance*, *attention points*, *extra information* and *number of repetitions (doses)*.

Figure 17: Screenshot of the CoCo web portal showing a training video from the care path "Hip Conservative".

To clarify what kind of information is given, the explanation of the “turning of torso” exercise is taken as an example.

Purpose contains an explanation of the purpose of the exercise, for the example exercise, it is explained that COPD causes the torso to stiffen, and that this exercise helps to loosen up the torso.

The *performance* section contains important information for the evaluation of the exercise. It contains the starting position, plus the movements needed. For the example, it indicates that the patient should sit on a stool, facing a mirror, and with both hands in the neck.

In the *attention points* important remarks are given to prevent the patient from making errors while performing the exercise. These are the points that should also be noted by the automated evaluation system. For the example exercise, the following things are important:

- Keep upright
- Do not move your hips
- Do not pull your neck
- Keep your elbows facing outwards

Extra information contains remarks on the performance, such as an alternative to the main movements, or a work around to cope with handicaps. For the example exercise, it lists that the arms could also be crossed on the shoulders.

Last, *doses* (repetitions) indicates how many times the exercise should be repeated. Unfortunately, these explanations are static, and not personalized. As a result, *doses* usually lists that the patient should adhere to the number of repetitions indicated by the therapist.

3.3 Suitability of exercise for automated detection

In the previous paragraph, an overview of the CoCo home training program is given. In this paragraph, the exercises of CoCo are evaluated with respect to the capabilities of the Kinect depth camera. The outcome of this evaluation (Appendix 10.3) will indicate for every exercise in the CoCo database, if the Kinect would be a suitable tool to evaluate it. Several key indicators are taken into account, to decide whether the exercise in question would be suitable for automated detection or evaluation with a Kinect. The final outcome of this analysis can be threefold:

A. The exercise is not suitable for detection
B. Performance of the exercise can be detected, but not evaluated
C. Performance of the exercise can be detected and evaluated

Incorporating the Kinect depth camera for exercises in group B can be useful to measure the adherence to the training program, but the Kinect depth camera cannot be used to evaluate if the patient did the exercises correctly, nor can it give feedback to the patient to improve his performance.

To give some information on why an exercise falls in category A, B or C, a "check" is given for each key indication. The following aspects are considered "key factors":

- Incomplete model
  - The detection algorithms have a simplified human model. This model lacks the hands, facial expressions and torso details, and has simplified shoulder joints.
- Fine movements
  - Although the resolution of the outcome of the detection algorithms is high, the accuracy can be limited. For example, loose clothing will severely reduce the accuracy. Therefore, the Kinect is not suitable to detect fine movements.
- Contact objects
  - The Kinect depth camera is triggered by blobs that have equal distance from the sensor. Consequently, when a person is holding a large object close to his / her body, that object becomes "part" of the person and will confuse recognition.
- Occlusion problems
  - Due to limited depth resolution, tracking of body parts that are in front of other body parts is limited. If the person holds his hands together on his belly, the Kinect cannot discern between the belly and hands, but when the hands are held 20cm in front of the belly, the Kinect will be able to discern between the hands and belly.
- Viewpoint problems
  - The Kinect measures depth from a single point. It cannot look through objects, it only knows silhouettes, and the distance from each point. The smaller the silhouette, the lower the accuracy. When the person is standing sideways to the sensor (with his right arm facing the sensor, and the left arm pointing away from the sensor), the silhouette gives little information on the pose.

- Orientation information
    - Absolute position outcome of the detection algorithms is much more accurate than orientation outcome. When, for example, the person is standing with his arms pointing forwards, and hand palms facing downwards, the simplified human model might indicate exactly the same pose than when the person is standing with hand palms facing upwards. In that case, the orientation of the arm joints is incorrect for at least one pose.

Number		Name	Incomplete model	Fine movements	Contact objects	Occlusion problems	Viewpoint problems	Orientation information	Detection possible	Evaluation possible	Comments
Heup Ac.		Oefeningen in zit									
HA17		Hakken/tenen optrekken	▼	▼	▬	▲	▲	▬	✗	✗	
HA18		Knieën strekken	▬	▬	▬	▲	▲	▲	✓	✗	Movement of foot cannot be evaluated
HA19		Knieën heffen in zit	▲	▬	▬	▲	▲	▲	✓	!	Chair can disturb detection
HA20		Spreid/sluit benen	▲	▬	▬	▲	▲	▲	✓	!	Chair can disturb detection

Table 4: A small part of the CoCo evaluation table (Appendix 10.3 contains full version).

Table 4 shows a small portion of the CoCo evaluation results table, the full table can be found in Appendix 10.3. The table contains the following information: the first column lists the exercise code used for reference in CoCo. The next column contains a small green patch, the shading of this patch is based on the number of patients that performed this exercise. During the years that CoCo was running, the number of prescriptions per exercise were logged. The darkest patches represent the most frequently prescribed exercise. When this column is left blank, no data was available about prescription frequency. The next column lists the Dutch title of the exercise. The six following columns match with the "key factors" given in the previous paragraph. A red downward arrow indicates that there will be problems during detection or evaluation. Yellow bars indicate potential issues, and green upwards arrows indicate that no problems are expected. The next two columns are "Detection possible" and "Evaluation possible". A red circle with a cross indicates that either detection or evaluation is not possible at all, a yellow explanation mark indicates potential issues, and a green check indicates no expected problems. The last column contains remarks on issues specific to the exercise, for example that the use of a chair can distort the detection and evaluation. In such a case, it is wise to look for another tool to keep balance, which occludes less from the human body.

Of the total of ca. 200 videos, 109 were analyzed, the others were left out because they were obviously unsuitable for automated detection and evaluation, such as behavior change and psychological exercises. Table 5 shows that out of the 109 videos analyzed, 41 were fully detectable, and only 11 were suitable for automated evaluation.

	Fully	Party	Not
Detection possible	41	36	26
Evaluation possible	11	42	58

Table 5: Analysis results of 109 exercise videos.

3.4 Selection of a target exercise

In paragraph 3.3, a method is defined to analyze whether the Kinect is a good tool to detect and evaluate a specific exercise. To develop and test the detection and evaluation framework conceived in this thesis, a "target exercise" is needed. This exercise should meet the following demands:

- detectable and evaluable (green checks in the last two columns of Table 4)
- relevant (higher number of prescriptions)
- automated detection and evaluation should have an added value

Number	Name	Incomplete model	Fine movements	Contact objects	Occlusion problems	Viewpoint problems	Orientation information	Detection possible	Evaluation possible	Comments
C13	Strekken van romp	▬	▬	▲	▲	▲	▲	✓	✓	Respiration cannot be detected
C11	Krachtoefening rechte buikspieren	▲	▲	▲	▬	▬	▬	✓	✓	
C12	Krachtoefening - schuine buikspieren	▲	▲	▲	▲	▬	▬	✓	✓	
HA33	Zijwaarts lopen	▲	▲	▲	▲	▬	▲	✓	✓	range is limited
HC10	Zit, buigen heup	▲	▲	▲	▬	▲	▲	✓	✓	
HC05	Rekken van spier buitenzijde bovenbeen	▲	▲	▲	▬	▲	▲	✓	✓	
HC14	Stand, been zijwaarts bewegen	▲	▲	▬	▲	▲	▲	✓	✓	Chair can interfere
HC36	Uitvalspas	▲	▲	▲	▲	▲	▲	✓	✓	
HC46	Stand op één been, bewegen met armen	▲	▲	▲	▲	▲	▲	✓	✓	
HC47	Stand op één been, bewegen met het andere been	▲	▲	▲	▲	▲	▲	✓	✓	
HC61	Huppelen	▲	▲	▲	▲	▲	▲	✓	✓	

Table 6: CoCo evaluation table, filtered for "Suitable for evaluation".

Table 6 shows the CoCo evaluation table, but then filtered for exercises that are expected to be both detectable and evaluable. Of these exercises, only four are "relevant", because only four have been subscribed a significant number of times during the CoCo trials. These four are C11, C13, HA33 and HC05. Of these four, a visual summary is given, followed by a discussion on the added value of automated evaluation.

- C11: "Strength exercise rectus abdominis"

Exercise C11 requires the Kinect SDK to be in "seated" modus. The Kinect SDK probably has a hard time detecting the crossing of the arms, but the effect of the exercise remains the same when the hands are lain on the shoulders without crossing the arms. Automated evaluation is a useful addition for this exercise, because the exercise can easily be done too fast or too superficial. Both aspects can be measured well with the Kinect SDK.

- C13: “Stretch of torso”

Exercise C13 also requires the Kinect SDK to be in “seated” modus. The quality of the execution of this exercise largely depends on the breathing technique used, which cannot be detected or evaluated with the Kinect SDK. Evaluation on whether the exercise is done too fast or too shallow is possible, therefore automated evaluation can still be useful, but not to the extent of that of C11.

- HA33: “Walk sideways”

Detection and evaluation of exercise HA33 is very simple, but the field of view of the Kinect limits the number of sideways steps that can be made before leaving the frame. An automated detection and evaluation system can evaluate if the exercise was performed too fast, or whether the steps are too small or too large. Because the exercise is very straightforward, the added value of such an automated system seems low.

- HC05: “Stretch of exterior upper leg muscle”

Automated detection and evaluation of exercise HC05 is possible and useful. During performance, it is important to keep stretch on the muscles in the upper leg. The automated detection and evaluation system can be used to check whether the patient held the last state long enough, and whether he or she bent their upper body far enough. Unfortunately, detection of crossed body parts is not reliable with a Kinect depth camera. Therefore evaluation of the crossing of the legs can be difficult.

3.4.1 Conclusion

Exercise HC05: “Stretch of exterior upper leg muscle” is chosen to be the target exercise, because it can be used to test the full skeleton model of the Kinect, and not the seated modus. For the target exercise, a more elaborate description is given in the next paragraph.

3.5 Detailed description of the target exercise

The full (translated) exercise description of the target exercise from the CoCo database is:

Introduction

The aim of this exercise is to extend the muscle on the outside of the upper leg. Maintaining the length of this muscle is essential for the movements that you have to make while walking.

Performance

Initial posture: standing upright, behind a chair, both feet flat on the ground.

Move your healthy leg across the front of your affected leg, and place both feet next to each other. Keep your knees straight while doing so. Now start moving your hips sideways in the direction of your affected leg, while moving your upper body in the opposite direction. You should feel stretch on the outside of your upper leg. Maintain this stretch for several seconds, before moving back to the initial position slowly.

Attention

Compensation movements, Sufficient stretch on the outside of the upper leg:

To increase the stretch in the outside of the upper leg, you can move the arm of the affected side above your head, towards the healthy side.

Keep looking forward.

If you need more support, you can grab the back of the chair with your hands.

4 Parameterization of exercises

4.1 Introduction

As stated in paragraph 2.3, few tools are available to make a structured, standardized and universal notation of an exercise. Traditional verbal descriptions are too vague for use in an automated system, and automated learning techniques lack context, making it very complex to indicate which aspects of the posture are important for the exercise.

Despite being tailored for the dancing world, Labanotation does offer a way to describe any type of human motion in a structured manner. Labanotation is an effort-based notation, only the actions that cause the change in posture are notated, for example “move arm upwards”. Body parts that do not perform a significant action, are not described. Therefore, the notation always contains as little information as possible. Labanotation fulfills most requirements for a structured exercise notation, but has one major downside: it is complex. Fortunately, the notation itself is not needed, only the structure / rationale behind the notation. Therefore a step-by-step plan is developed, which will help to give all information needed to register the full exercise, without the need of extensive knowledge on the Labanotation. This chapter will describe this plan, including the data model to support it. This data model will be the input for the automated detection and evaluation system design, discussed in Chapter 5.

4.2 Parameterization of the target exercise

In order to detect and evaluate the target exercise chosen in the previous chapter, it needs to be parameterized. Intuitively, the exercise can be defined by the three postures in which the patient should be static for a certain time. The starting state, with both feet next to each other, state 2, in which the legs are crossed, and state 3, in which the upper body is bend to the side. The patient starts in state 1, then proceeds to state 2. From state 2, the transition to state 3 is made slowly, and state 3 is held for some time, before returning to state 2 slowly. In Figure 18, these three states are depicted.

Figure 18: Main states of CoCo exercise HC05.

For each of these three states, and for the transition between these states, the description gives several aspects that should be fulfilled before the exercise is performed correctly. In order for the automated system to check these aspects, the translation into parameters must be made. These parameters have the joint positions (x,y,z) of the Kinect SDK as input. A possible set of parameters is given in Table 7, as an illustration. These parameters are not tested in real life.

Aspect	**Parameter**
• *State 1: Normal stand*	
Check on straightness	Absolute range z-coordinates Hip_Center and Shoulder_Center < β
Check on distance between feet	Abs(Ankle_right(x) - ankle_left(x)) < β
Keep for certain time (stability)	Sum of absolute distance traveled by Hip_Center and Shoulder_Center < β
• *Transition state 1 → 2*	
Swing bad foot in front of good foot	Ankle_right(z) + π - ankle_left(z) > β AND Check for x-legs
Keep knees straight	2/(Ankle_right(z) + Hip_right(z)) + π > Knee_right(z) AND 2/(Ankle_left(z) + Hip_left(z)) + π > Knee_left(z)
Keep upperpart straight	Abs(Shoulder_center(z) – Spine(z)) < β AND Abs(Shoulder_right(z) – Shoulder_left(z) < β
• *State 2: Stand with feets crossed*	
Both feet next to each other	Abs(Ankle_right(z) - ankle_left(z)) < β
Whole body straight	Absolute range z-coordinates Hip_Center and Shoulder_Center < β
Keep for certain time (stability)	Sum of absolute distance traveled by Hip_Center and Shoulder_Center
• *Transition state 2 → 3*	
Move hips in direction of affected leg	Hip_center(x) - (2/(Ankle_right(x) - ankle_left(x)) < β
Move shoulders in direction of sound leg	Shoulder_center(x) - (2/(Ankle_right(x) - ankle_left(x)) < β
Movement speed should be slow	Sum of differential of Hip_Center and Shoulder_Center
• *State 3: legs crossed, upper body bend*	
Keep for certain time (stability)	Sum of absolute distance traveled by Hip_Center and Shoulder_Center
• *Transition state 3 → 2*	
Move hips towards neutral	Hip_center(x) + (2/(Ankle_right(x) - ankle_left(x)) < β
Move shoulders towards neutral	Shoulder_center(x) + (2/(Ankle_right(x) - ankle_left(x)) < β
Movement speed should be slow	Sum of differential of Hip_Center and Shoulder_Center

Table 7: Parameters of exercise HC05 (Greek letters are variables that are patient dependent).

Table 7 gives a good basis to build an evaluation algorithm. Each parameter can be explicitly programmed into the system. After defining patient specific threshold ranges, the performance can be evaluated. However, when another exercise is to be added to the system, the whole process of defining, implementing and testing all parameter checks needs to be done again. This implies "reinventing the wheel" many times, which is a serious downside. Furthermore, it is hard to say beforehand whether a certain parameter is actually implementable into the detection and evaluation system. To obtain a universal parameterization structure, a method is proposed, which follows the structure of the Labanotation. This method is given in section 4.3 in the form of a step-by-step decision model. All aspects of Laban that are not applicable to automated detection and evaluation of exercises are omitted. Examples of omissions are: expression accents (angry, sad) and interaction between multiple dancers. The method conceived in section 4.3, should enable anyone to get all parameters that make up a simple Labanotation, without actually knowing the Laban methods. In the real Labanotation, the target exercise, would be as displayed in Figure 19.

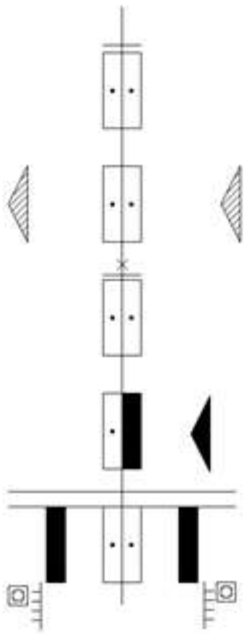


Figure 19: Labanotation of target exercise.

4.3 Development of a parameterization framework based on Laban movement analysis

Laban is based on four components: Body, Effort, Shape and Space (Foroud & Whishaw, 2006). Body and Space cover the kinematics of the movement, and Effort and Shape the non-kinematic aspects. Initially, only the Kinematics are relevant for the detection and evaluation system, because these describe the movement of the body parts, and the movement of the person itself. Effects of the non-kinematics described in Effort and Shape, such as anger and rhythm, are not relevant for the exercises.

Labanotation starts with an initial pose, indicated by the symbols below the double horizontal line. This remainder of the notation only describes changes to this initial pose, that have to performed, for example "move hand upwards". The concept of describing an absolute position is different from only detecting changes in position. To limit the complexity of the initial proof of principle system, it has been limited to detection of change in position. This means that the system requires a single initial starting position. The first movement described, always describes the changes to this initial position.

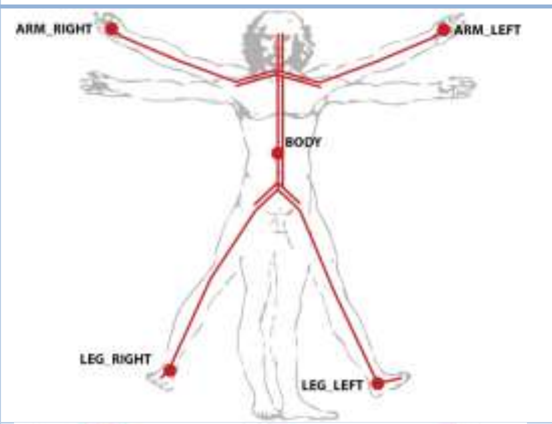
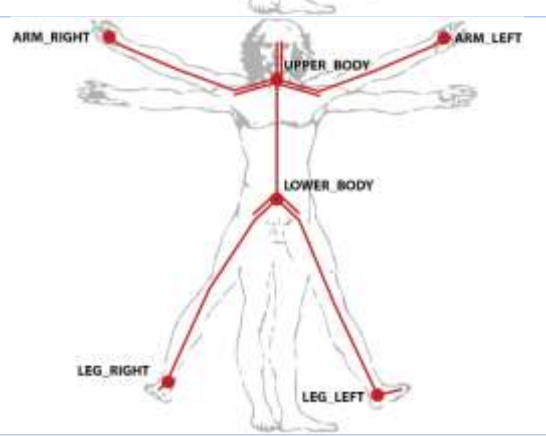
The initial position is standing with feet next to each other, toes and face facing forward, and arms hold along the body (not touching it), hand palms facing towards the body.

Because only the changes to an initial position are stored, it is not straight forward to get the absolute position at any stage of the exercise, other than the initial stage. For example, to get the absolute posture at stage 10, first stages 1 to 9 have to be processed. Each processing step will add some error, thus the calculated absolute posture at stage 10 can deviate significantly from the intended posture. For detection and evaluation of the exercise, absolute postures are not essential, but they can be of great use for visual feedback to the user. For example to show an animated avatar that performs the exercise.

4.3.1 Choosing the right joint model

Before the movement of the body parts can be registered, it has to be known which body parts perform relevant movements. It's important to use the least complex joint model and never to describe movement of body parts that are not relevant to the exercise. Not only would this add non-information, it would actually force the person into performing this non-relevant movement exactly as described. For example, if the head is described as "place" (= no movement) throughout the exercise, this means the person is not allowed to look around, even though this has little influence on his quality of performance with respect to the exercise.

In most cases, a subset of the full joint model can be used, but in some cases it's more practical to define new body parts. For example the "arms" are not defined in the full joint model (only the hand, wrist and elbow). Three extra joint models are defined, which range from limited to nearly as complete as the full model. These models are shown and explained in Table 8.

	Joint model	Comments
Basic v v v v v v v v v v		One of the key concepts of Labanotation is to use as little detail as possible. The most basic joint model is displayed left, it consists of a single "body", the arms and the legs. Describing the movement of these 5 elements suffices to describe for example normal walking. The double lines indicate a rigid structure, i.e. when an arm moves, its point of attachment is at the outer side of the rigid structure.
		One step more complex, a division between upper and lower body is made. This complexity is required to describe a bending movement. For example bending forward.

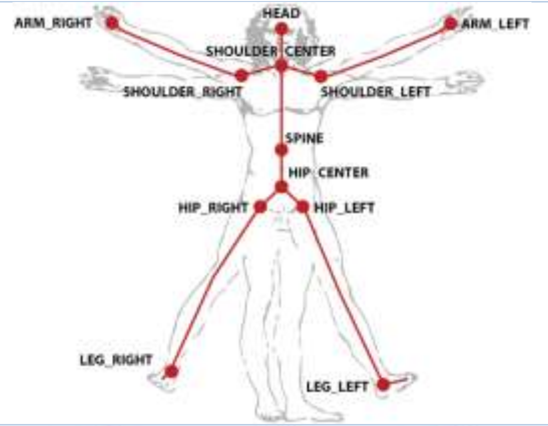
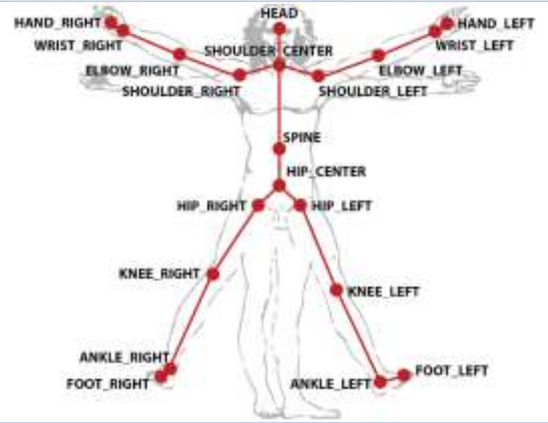
<table>
<tr>
<td>v
v
v
v
v
v</td>
<td>
</td>
<td>The next step towards the full model, has simplified arms and legs, but the full set of joints in the torso. Trough description of the movements of these joints, a movement like nodding with the head can be described.</td>
</tr>
<tr>
<td>v
v
v
v
Complex</td>
<td>
</td>
<td>The most complex model, is by definition the full joint model of the current MS Kinect SDK. Future versions of the MS Kinect SDK will support more joints, for example an extra “thumb joint”, increasing the complexity of the “full joint model”.</td>
</tr>
</table>

Table 8: Joint models

In every description, the basis of the body has to be present. This is the “origin” of the body, and describes movement of the “whole” body. In the Kinect skeleton joint model, the “hip center” joint represents the “origin” of the body.

For the target exercise, the full joint model is not needed, the simplest joint model does not support the bending of the upper body to the side, and therefore the second body model is chosen. The relevant body parts are the arms, legs, upper body, and support (always present).

4.3.2 Timeline and durations

As can be seen in Figure 19, each body part has its own column in the Labanotation. This means that each analysis step has to be performed separately for each body part, thereby always starting with the “support”.

The first thing that has to be analyzed *per body part* is the timeline. The timeline is very important, because the distance travelled is directly coupled to the time. If, for example, the “upwards” movement of the arm has a duration of 15 seconds, and the body part’s normal speed is 10 cm/s, the expected traveled distance of the arm is 150 cm. Obviously, the arm cannot go upwards that far.

The timeline is made up by segments. In essence each motion segment starts at a point at which no body part performs any movement, and lasts till the next moment at which no body part performs any movement, just like the example at the beginning of section 4.2. A single segment can be repeated, for when the exercise contains repetition, or a set of segments can be repeated (A → B → C → B → C → B → C). In the flow chart of Figure 20 the steps needed to identify the segments are depicted.

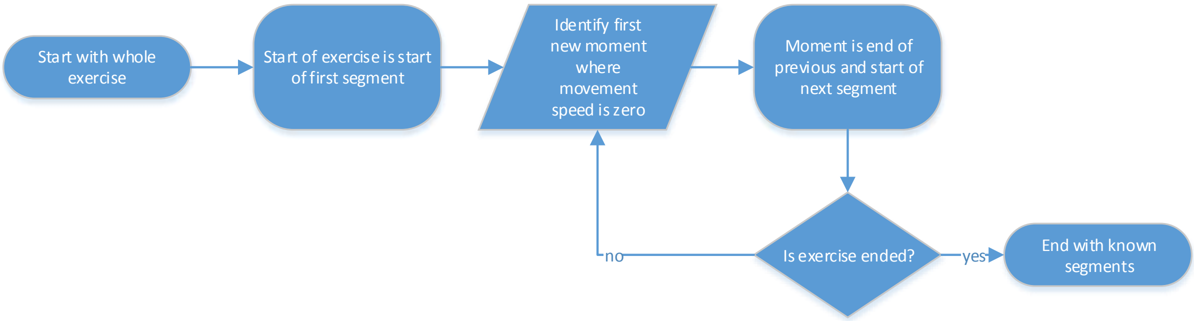


Figure 20: Steps to be taken in order to define the segments that form the timeline of an exercise. Has to be applied for each body part.

The flow chart of Figure 20 must first be executed on the "support", but because most exercises are performed in place, the "support" will have a single segment, equal to the duration of the whole exercise. When a body part has a clear moment for which the movement speed is zero (usually a turning point, at which an extreme was reached), this marks the start of a new segment. It can be that another body part has segments which are shorter than those of the other body parts. Either the movements of that body part start later, or end earlier, than of the body part with longer segments. In the first case, the movement specification must include a delay, to make it start later. In the second case, the movement duration can be shorter than for the other body part, thereby not filling the full segment. If movement "A" starts before the start of movement "B", and "A" also stops before the stop of movement "B", then the segment will start at the start of "A" and end at the end of "B". In this way, a single timeline is made for all body parts. Not each body part has to be active in each segment, but segments may not overlap, and are always consecutive. The steps visualized in Figure 21 are used to identify the duration of the movement for each body part.

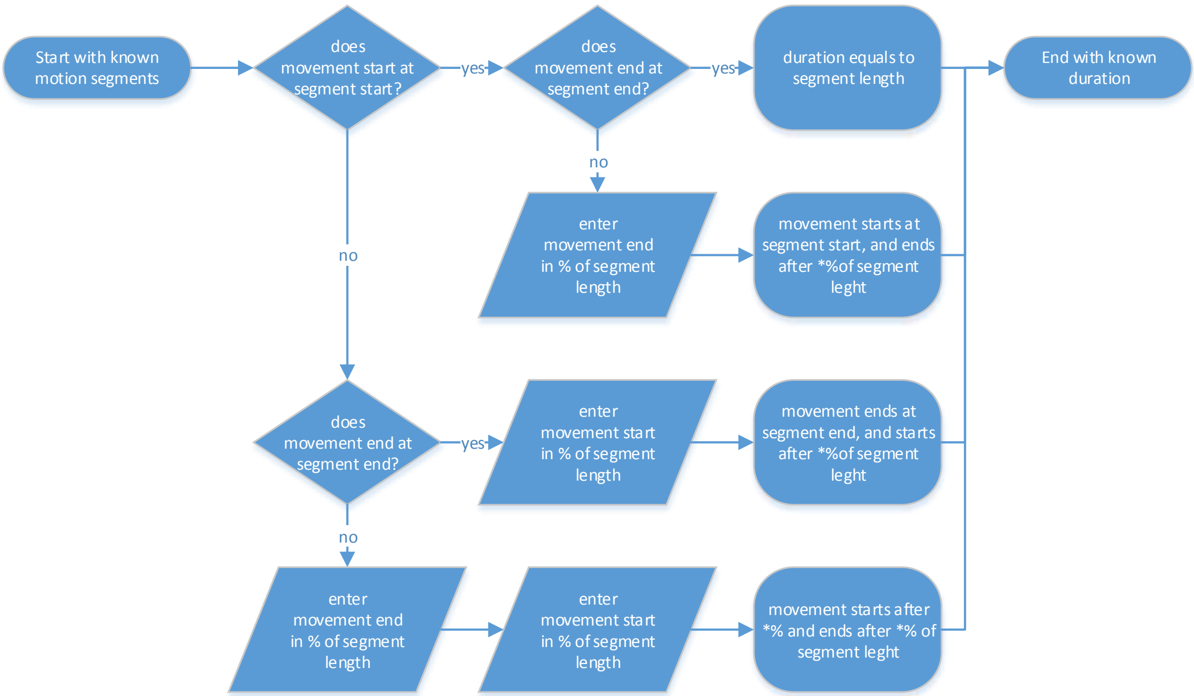


Figure 21: Steps to be taken in order to define the duration of each movement for each body part.

4.3.3 Movements

The movements that are used to define the timeline and durations, evidently need to be parameterized themselves. To identify movement, three “types” of translations are available: Horizontal Translation, Vertical Translation and Rotation. The first two are also used to create a special type of translation: the Relation.

Horizontal Translation

A horizontal translation, is a movement of a body part in a horizontal plane. Figure 22 shows the steps needed to define the horizontal movement direction. The distance the body part travels during this movement, is not related to the direction, but to the duration of the movement, as explained in the previous paragraph. When there is no movement, this results in a “horizontal direction = place”. There are eight other horizontal directions predefined to notate movement in the Labanotation:

- F = Forward
- B = Backward
- L = Left
- R = Right
- LF = Left forward
- RF = Right forward
- LB = Left backward
- RB = Right backward

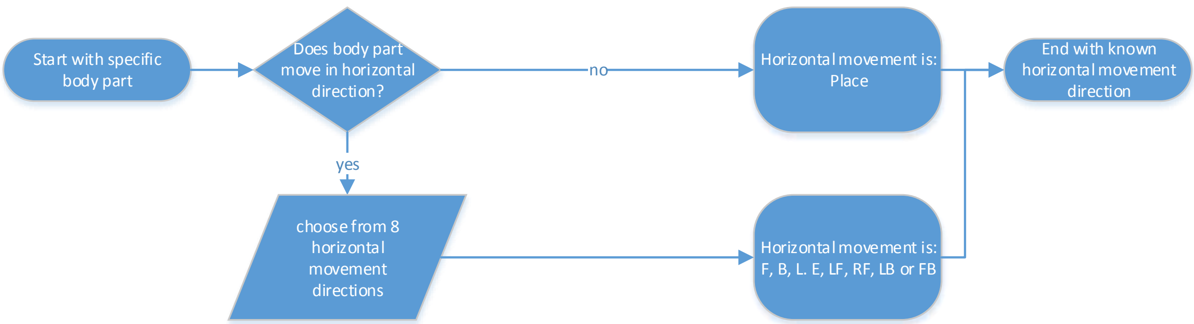


Figure 22: Steps to be taken in order to define the horizontal direction of a movement.

Many movements do not take place exclusively in the horizontal plane. Therefore, the horizontal movement, can be adjusted with a vertical direction modifier.

Vertical Translation

For movement in the vertical plane, only three options are available: up, middle and down. Middle is the neutral position, thus remaining in the horizontal plane. An upwards movement of the foot in place (thus bending the knee) would have horizontal direction place, and vertical direction “up”. The simple steps to define these three options are depicted in Figure 23.

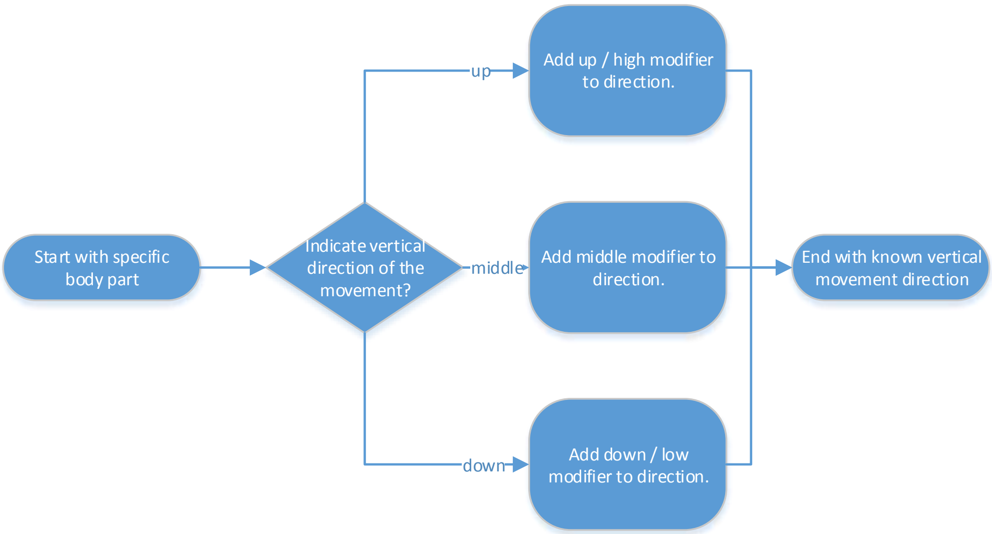


Figure 23: Steps to be taken in order to define the vertical direction of a movement.

Identify rotation

The last type of translation is used to describe a rotation. For example, rotating the hand such that the hand palm faces downwards instead of upwards. The imaginary rotation axis is drawn trough the parent bone. For the example of the hand, the rotation axis is the underarm. The direction of the rotation can be identified, and the amount of rotation, in four steps: 90°, 180°, 270° and 360°. The amount of rotation therefore is NOT related to the duration of the movement, contrary to the horizontal and vertical translations. The steps needed to identify the rotation, and depicted in Figure 24

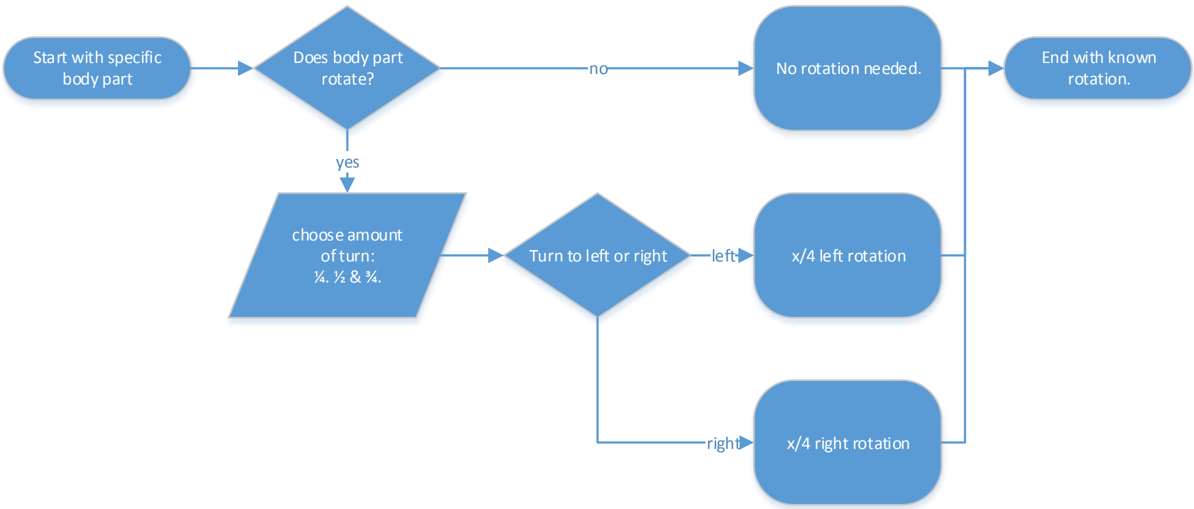


Figure 24: Steps to be taken in order to define the rotation of a movement.

Identify relation

A special type of translation, is the “relation”. Relations are used to indicate interaction between two body parts (or foreign objects). For example, “left hand” is “up” in relation to “head”, indicates the person should hover his left hand above his head. Such a relation is easier to comprehend for a real person, than a series of horizontal and vertical translations, which guide the hand to the place above the head. Unfortunately, the opposite is true for automated detection using software. The horizontal and vertical translations used to define a relation, are the same as the separate horizontal direction and vertical directions described above.

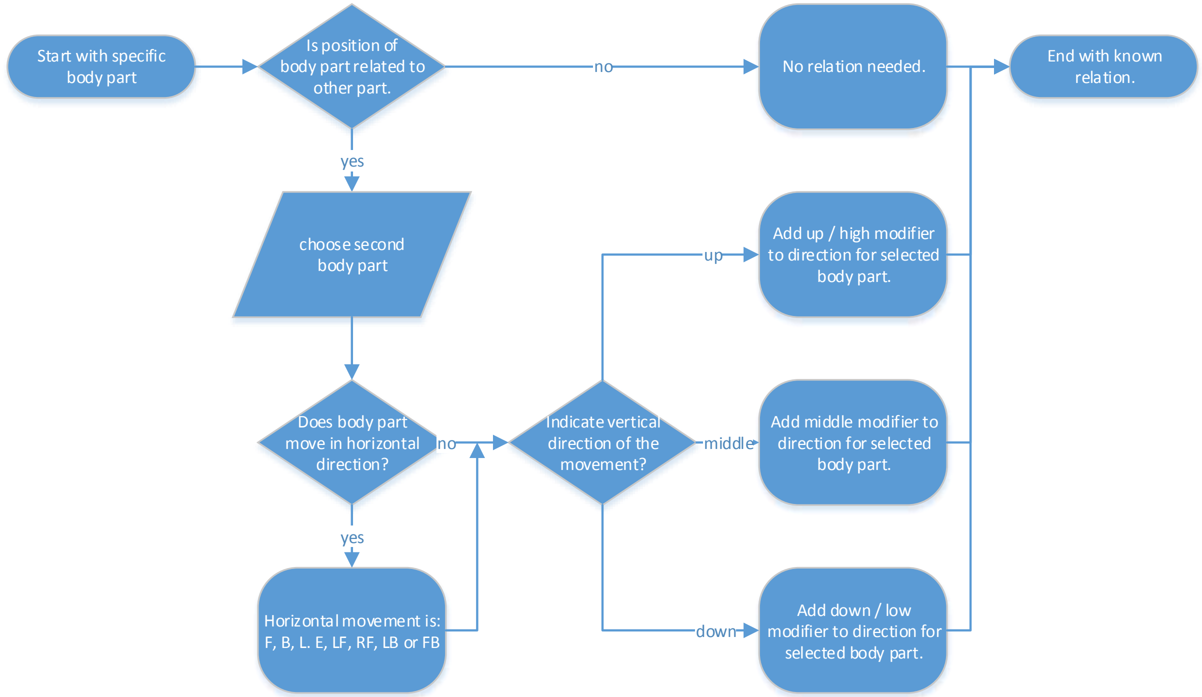


Figure 25: Steps to be taken in order to define a movement of a body part via the relation in position to another body part.

4.3.4 Accents and space measurements

If no special notice is given, all movements are performed "normally". This means that the movement is performed at an "average" speed, and the movement takes up the whole duration. Thus, there is a direct relation between the duration of a movement and the distance the specific body part should travel. If the movement should be performed faster or slower than "normal", this can be identified via accents. A movement can have three "speeds": normal, slow and fast. What the exact values of these speeds are, depends on the type of movement, the body part and the person performing the exercise. The speed could be made a function of the age or sex of the patient.

4.3.5 Personalization

Next to the static parameterization of the exercise, a dynamic component is needed as well: "personalization". In the previous paragraph, personalization based on age and sex is already mentioned. These two factors can influence the speed at which the exercises are performed. The timeline can be dependent on the properties of the patient as well, for example in the number of repeats, or the time the exercises takes. A more complex type of personalization, is to modify the actual movements based on the properties of the patient. For example, the exercises for the acute hip patients in the CoCo database, are all aimed at improving the affected side. Therefore, these exercises need to be mirrored if the patient is affected at the other side.

If patients are unable to perform a certain exercise due to handicaps, the exercise should also be adapted. For example, if a patient misses an arm, the exercise specification should not demand that both arms are raised. Such modifications of the exercises are beyond the scope of this research.

4.4 Data model and conclusion

The exercise properties, the timeline, the movements with their durations, and the personalization options together form a hierarchical structure. This structure is used for the data model, in which the exercises will be communicated and stored. On top is the exercise, with certain parameters, such as the name and category. Each exercise also has a joint model and information on personalization. Exercises have one or more segments, these segments are portions of the timeline. Each segment contains multiple body parts. For each body part, all the movement types are given (direction, rotation etc.). Each modifier has its own duration. This results in the following data model:

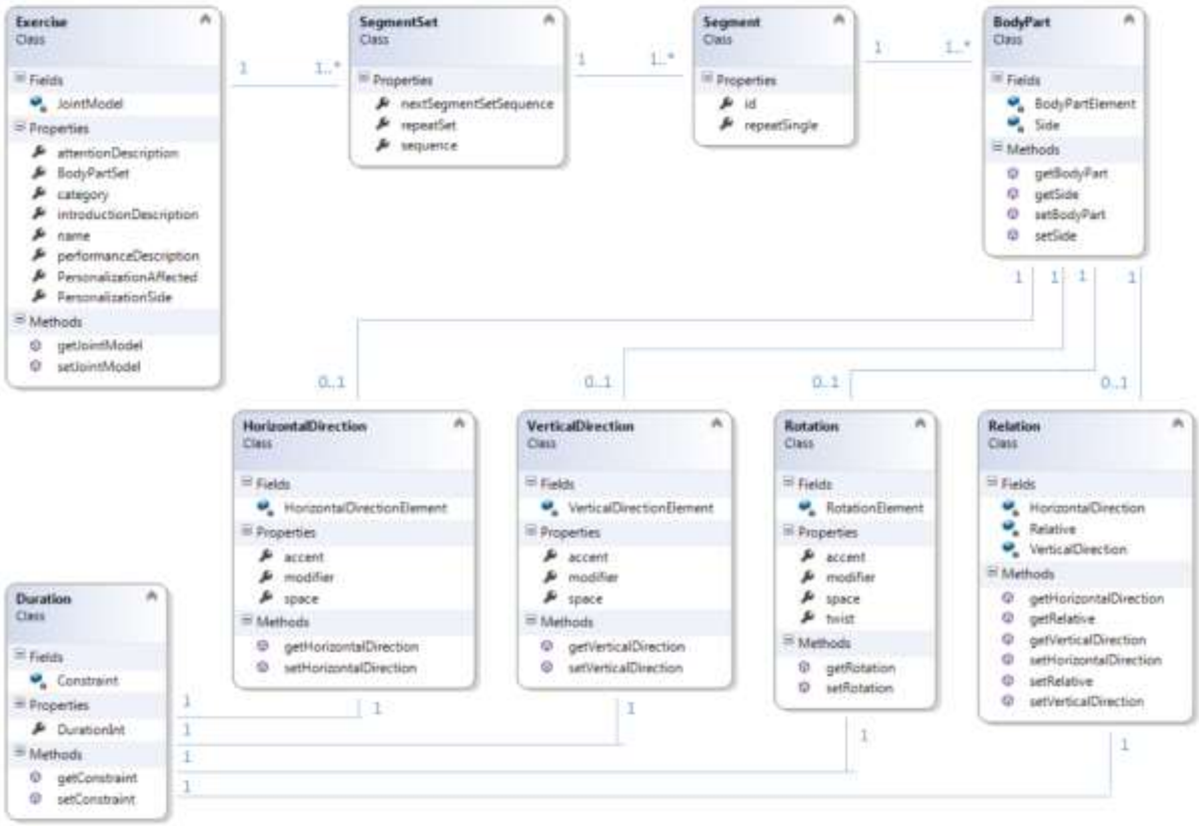


Figure 26: Class diagram of parameterization data model.

This model will be converted to an XML and Class data model, for easy implementation and communication between the software and some web site. All aspects from the data model will be input for the logic that is developed in the next chapter (Chapter 5), and implemented in Chapter 6.

5 Automated evaluation of an exercise

5.1 Introduction

In the previous chapter, the data model was described in which the exercises are parameterized. This chapter will discuss a system to read such a parameterized exercise, compare it to skeleton movement measured by the Kinect depth camera and evaluate the performance of the exercise.

The system to detect and evaluate the exercises can be divided into three components:

- Read and interpret the parameterized exercise.
- Measure and process skeleton data from the Kinect depth camera.
- Compare measured movement and compare this to the exercise in order to interpret and evaluate the movements.

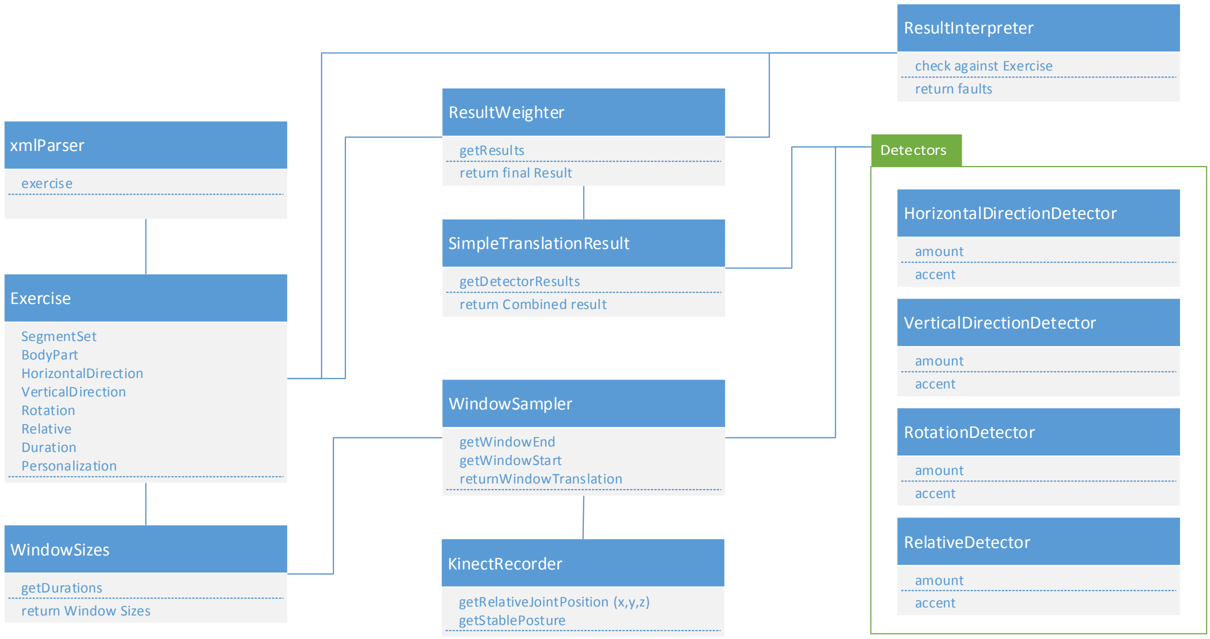


Figure 27: Simplified class overview of the detection and evaluation system.

Figure 27 shows a simplified class diagram of a possible implementation of the detection and evaluation system. The left blocks deal with parsing of the exercise specification, the "ResultIterpreter" deals with the interpretation, and the remaining blocks deal with the measurement. The next three paragraphs follow this division.

5.2 Processing of parameterization

5.2.1 Exercise specification playlist

Exercises are specified using a hierarchical data model based on the Labanotation. This data model is shown in Figure 26. But a real life performance of an exercise happens in a sequential instead of a hierarchical manner, therefore the exercise specification needs to be converted to a sequential format. This sequential "playlist" contains all successive segments, and takes the "repeat" and "next segment" information in to account. When a certain segment has to be repeated 5 times, it will be placed in the playlist 5 times. The exercise playlist indicates at each moment in time what each body part must do. Because not every body part performs an action all the time, the relevant number of body parts can change depending on the segment. If a body part is not available at a certain time, this means that this body part does not perform a relevant action at that time. The actions of this body part at such a moment are ignored. When it is important that the body part is kept at a constant position, this specification should explicitly state this, for example by indicating the Horizontal direction is “Place”.

5.2.2 Sample window lengths

Based on the duration of all translations in the exercise specification, a suitable sampling window duration is chosen. The duration of a window is the time in between two samples of skeleton positions. These two positions are subtracted from each other, to get the relative movement of the window, which is the translation at the given time. The shortest window at which the movement data is sampled should be at least as short as the shortest duration of the translations in the exercise specification. In practice even a shorter window is needed, because sampling of the windows is not perfectly in sync with the performance of the actions. As discussed in the next paragraph, the system can sample using multiple window sizes. The longer windows are all an exact multiplication of the shortest window to keep implementation simple. The duration of the longest window is related to the duration of the longest translation. The other windows are equally divided between the shortest and longest window, rounded to be multiples of the shortest window.

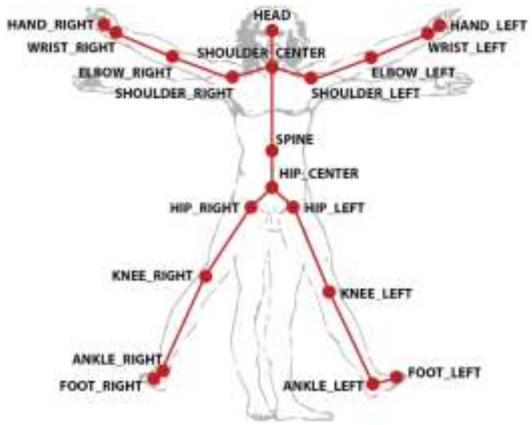


Figure 28: All joints of the Kinect skeleton model.

5.2.3 Relevant Body Parts

In paragraph 4.3.1 the different joint models which can be used to describe the movements, are discussed. Not all joints in these models are part of the Kinect skeleton model, therefore a translation between the body part names used in the exercise specification, and the Kinect “compliant” joints must be made. Body parts without a direct equivalent are: Body, Upper Body, Lower Body, Arm and Leg. The “Body” is by definition the most important body part in the Labanotation hierarchy, and the equivalent exists in the Kinect hierarchy as well: “Hip Center”. Movement of the “Upper body” is related to movement of the following Kinect joints: Shoulder Center/Left/Right and the Head. The “Lower body” relates to Hip Center/Left/Right and the Spline. For the upper and lower body, the central joints (Hip Center and Shoulder Center) represent the movement best, because these are in the center of mass of a relatively ridged part of the body. “Arm” relates to the left or right Wrist, Hand and Elbow. And last, the “Leg” relates to the left or right Knee, Ankle and Foot. For the arms and legs, the movement of the extremities is more important than the movement of the elbow or knee, because the movement is limited in the parent joint (hip/shoulder), and elbow or knee are close to the parent joint.

Most of the body parts are present at both the left and the right side of the body. In the exercise specification the following identifiers are available to control which side is meant: both, left, right, affected and non-affected (see paragraph 4.3.5). These modifiers are ignored for body parts that have no left and right version: Hip center, Spine, Shoulder center and head. For the other body parts, the option “both” will place both the left and right version in the list of body parts that need to be tracked (the relevant body part list). For the options “left” and “right” evidently, the left or right version of the body part is placed in the list. The “affected” and “non-affected” options are substituted by left or right, based on the “affected” field in the personalization options (see paragraph 4.3.5). Summarized: the body part parser reads through the whole exercise specification. It places each body part which has an active role in the exercise in the relevant body part list, but only after it is translated to a specific Kinect compliant joint.

5.3 Measure and process skeleton data from the Kinect depth camera

In order to compare the movement of the patient with the exercise playlist discussed in the previous chapter, these movements first need to be detected and formatted in the same “language” as used for the playlist. For a given duration, for each body part, the system should be able to detect the following aspects: amount and direction of the horizontal en vertical translations, rotations, and movement relative to another body part.

5.3.1 Sample windows

In paragraph 5.2.2 the calculation of the window lengths based on the durations of elements in the exercise specification is discussed. Even for very short elements in the exercise specification, the native sample interval (inverse of sample rate) of the Kinect will be much shorter than the desired interval between two samples which are used to calculate the translations. In other words, the sample rate of the translation detection is much lower than the sample rate of the source Kinect data. The simplest way to convert the sample rate is to discard unneeded samples. A large disadvantage of subsampling by omitting samples, is the sensitivity for high-frequency noise. See for an example Figure 29, if in this signal, the two sample positions would be at x=2 and x=8, the result would be an increase of 22, but the full signal clearly shows a decreasing instead of an increasing trend. This problem is overcome by low pass filtering of the source signal with a cutoff frequency of half the final sample rate. This removes fast transients in the position data, and thus “smoothens” the movements.

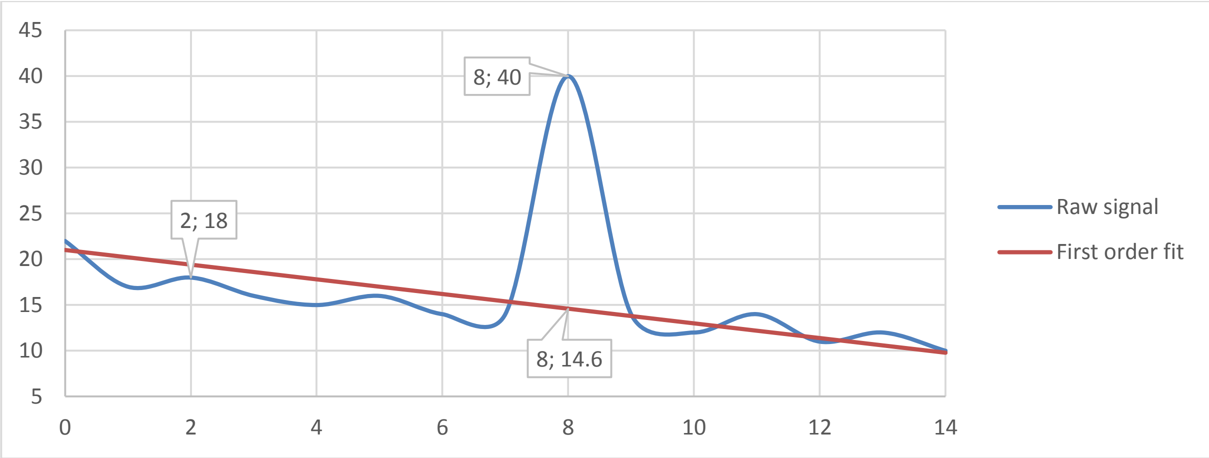


Figure 29: Signal with downwards trend, and a single noise spike.

The duration and start time of the measurement of a single translation should match the duration and start time of the action which is specified in the playlist. Obviously, the position in time in this playlist is not known beforehand, and therefore the desired window length to sample the translation is undefined. Two ways to deal with this are: to combine multiple short windows to generate longer windows, or use overlapping windows of multiple lengths. Both methods provide a way to measure translations that have a longer duration, while ignoring shorter "noise-like" translations. The "overlapping windows of multiple lengths" option is more complex to implement, but has the advantage that the window lengths do not need to be multiples of each other, and allows the cutoff frequency of the low pass filter to be specific to the window size. The last advantage becomes greater if the difference in length of the shortest and longest window becomes larger.

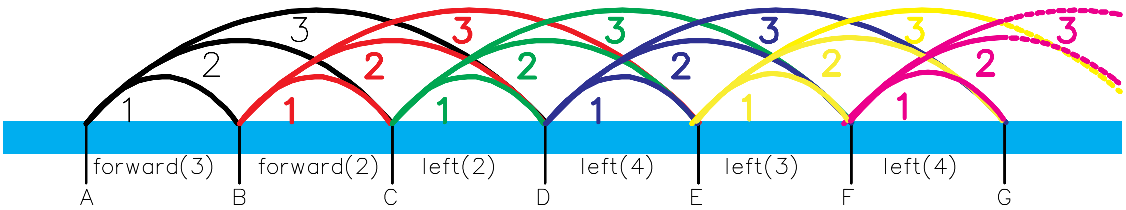


Figure 30: Example of sampling using three different window sizes, with overlap each with a horizontal translation direction and an amount between brackets.

Figure 30 shows an example of sampling with three different window sizes. Only the shortest window needs to be sampled, because the longer window lengths are multiples of the short window length. As such, the result of the longer windows can be calculated afterwards. The horizontal part of the result of each short window is shown in the figure, including the amount in brackets. The results of the longer windows can be calculated by summing the amount, and averaging the angle, as explained in paragraph 5.3.3.

5.3.2 Data preprocessing

Besides low-pass filtering the source position data, other forms of preprocessing can be of use. To limit required processing power, translation samples which are almost zero, can be put at exactly 0. Next to applying a threshold, preprocessing can also process trends. If a translation retains its direction and amount over multiple windows, the chance that it is measurement noise, is low, even if the amount per window is low. Therefore such a "long term" translation should have a greater impact during the data interpretation. Detecting whether a translation is monotone, can be done by calculating the change in amount and direction between the current and the previous sample of the same body part and convert this to percentage. The percentage change in angle is calculated by: $\frac{MD^{-1}}{|\theta_2-\theta_1|} * 100\%$, with "MD" being the maximal allowed deviation. If both the percentage change in amount and angle are above a predefined threshold, the translation can be considered monotone.

5.3.3 Translation recording

In the previous paragraph information is given on combining amount and direction of multiple windows. This can only be done after the amount and direction of a single window are known. The next paragraph discusses how to convert the position samples into a specific direction and amount of translation.

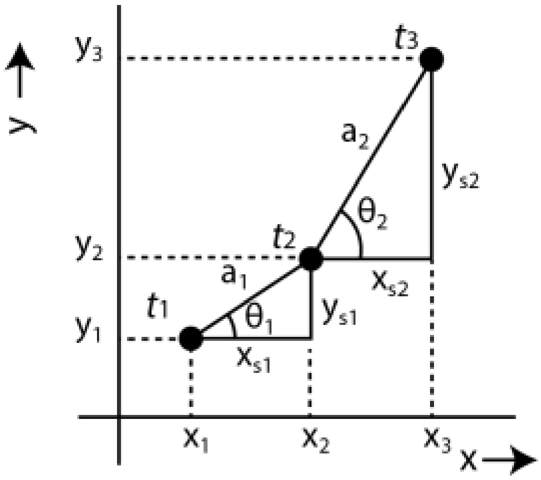


Figure 31: Schematic representation of three position points in 2D.

Figure 31 shows a 2 dimensional representation of three consecutive samples of a single body part. T_1, T_2 and T_3 are sampled exactly one window length in time from each other. These three position samples are converted into translation samples, which form the input for the rest of the analysis. A translation sample is the difference between two position samples. The three position samples result in two translation samples. For example, the result of the first translation in Figure 31 is ($x_{s1} = x_2 - x_1$) ($y_{s1} = y_2 - y_1$) ($z_{s1} = z_2 - z_1$) (the z-axis is not shown in Figure 31).

From the 3 dimensional translation sample, the amount and direction of the translation are calculated. The relation between the amount and duration of a translation is given via the "normal speed" (see paragraph 4.3.2). The direction relates to the eight horizontal directions (plus the static "place"), three vertical directions ("up", "middle" & "down") and two rotation directions ("left" and "right"), see paragraph 4.3.3. Because the amount and direction are specific for a single type of translation in the Labanotation, only a 2-dimensional plane or a single axis is needed to calculate them. For the horizontal directions, the X and Z axis are used, and the Y axis is used for the vertical directions. Calculation of the amount of the horizontal translations is done using the Pythagorean Theorem. An example is given in Figure 31, the first amount in this graph is $a_1 = \sqrt{{x_{s1}}^2 + {y_{s1}}^2}$. The direction of the horizontal translation is based on the angle of the translation, from the same figure, the first angle is $\theta_1 = \tan^{-1}\frac{z}{x}$. Each measured angle is related to a direction from the Labanotation. For the horizontal translations, the "right" direction is defined to be 0 / 360°, which relates to a translation with positive X values, "forward" is defined to be 90°, and relates to negative Z values. Figure 32 shows all 8 directions possible for the horizontal translations. The angles listed in this figure relate to the boundaries of each direction, for example, when a measured angle is between 22.5 and 67.5°, the direction "Front Right" is coupled to this angle.

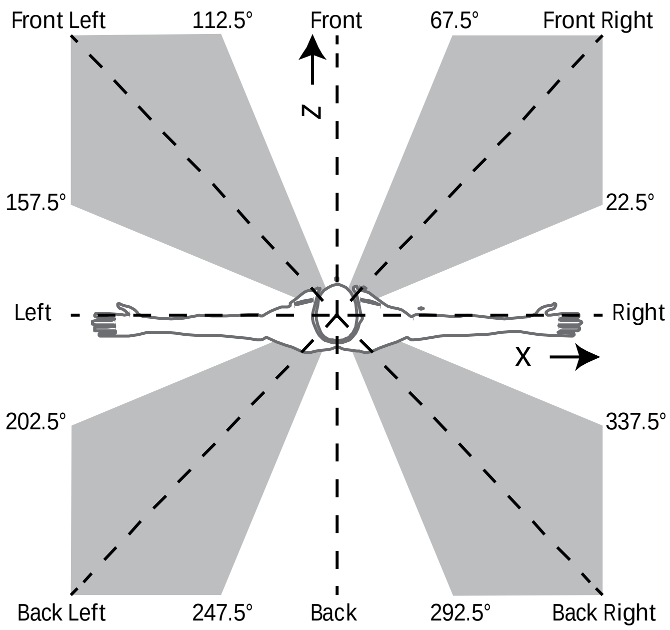


Figure 32: The basic horizontal directions drawn on the z and x axis.

The calculation of the amount and direction of a Vertical translation differs from the Horizontal translations. The amount of the vertical translation is directly given by the y-axis of the translation. Vertical translations can have only three directions: "up", "middle" and "down". These directions are determined via a predefined threshold on the y-axis, if the translation (relative change in position) exceeds this threshold, it is counted as "up" and if the translation is smaller than minus-threshold it is counted as "down". All movements in between the positive and negative threshold are counted as "middle".

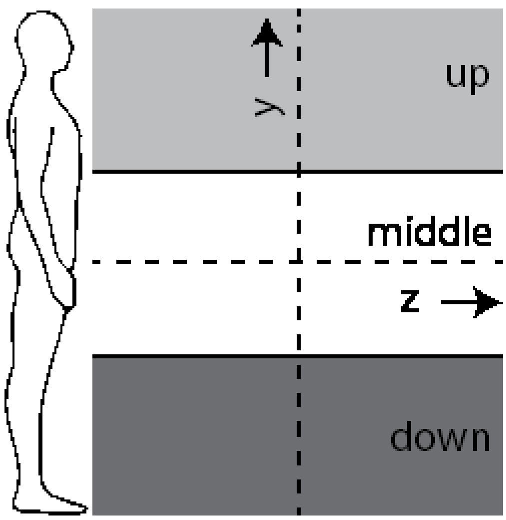


Figure 33: The basic vertical directions drawn on the y and z axis.

The calculation of the amount and direction of the rotation translations differs from that of the horizontal and vertical translations. Rotations can only take place in two directions: "clockwise" or "counter-clockwise" ("right" or "left"). The amount of rotation relates to the number of degrees of rotation, for example if the hand is rotated from facing down, to facing up, the amount is 180°. Joint positions on themselves do not give any information on the orientation of the body parts. Luckily, the skeleton model of the Kinect also contains information of the orientation of "bones". These orientations are stored in the parent joint (see Figure 6 and Figure 7 in paragraph 2.1.3). Translations are solely based on relative changes, therefore the hierarchical model is not important. The amount and direction follows directly from the change in the sign and quality of change between two samples of the "bone orientation". Figure 34 shows an example of a "clockwise" rotation of amount θ_r of the wrist.

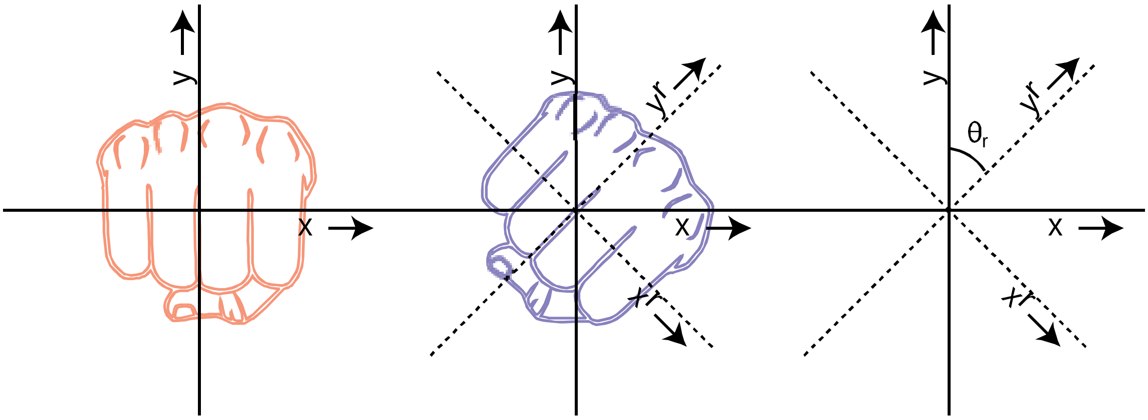


Figure 34: Schematic view of the rotation of the arm, seen from the front of the fist.

Relative translations are only defined via horizontal and vertical directions. Instead of the relative change in position of a single body part over time, the relation is calculated by measuring the relative position of one body part compared to another. For example, if the hand has a relative translation with the head of "horizontal = place", "vertical = up", the x-position and z-position of the hand and head should be in the same range, and the y-position of the hand should be higher than that of the head. The definition of the horizontal and vertical directions is the same as for the separate translations discussed above, but the concept of *amount* is more difficult. In the Labanotation, the duration of the translation defines the distance traveled, thus the longer a translation lasts, the higher its amount. For a relative translation, the maximal "amount" is reached when the relative positions are exactly as defined, for the example, the maximum would be when the x and z positions are exactly the same, and the y-position is higher. Beyond a certain threshold of deviation between the specified and actual relative positions, the amount is counted as zero.

5.3.4 Comparison between exercise specification elements and detection results.

When the exercise playlist and the result of the translation samples are known, these have to be compared in order to evaluate the exercise performance. Table 9 shows the concept of comparing the exercise specification with the detection results. In the left column, the specification is shown, and the three columns on the right show respectively the short, medium and long window. It's important to notice that this table only refers to a single body part and a single translation type. In reality each relevant body part would have such a table for each translation type in use. Each element in the exercise specification has a certain duration. Therefore the element must be matched against a detection result with the same duration.

<table>
<tr><th>Exercise</th><th>Window short</th><th>Window medium</th><th>Window long</th></tr>
<tr><td>Forward (9)</td><td>Forward (10)</td><td rowspan="2">Forward (8)</td><td>…</td></tr>
<tr><td rowspan="3">Side (8)</td><td>Down (4)</td><td rowspan="3">Side (15)</td></tr>
<tr><td>Up (7)</td><td rowspan="2">Side (8)</td></tr>
<tr><td>Side (4)</td></tr>
<tr><td>Up (10)</td><td>Middle (10)</td><td rowspan="2">Middle (6)</td><td rowspan="2">…</td></tr>
<tr><td>Side (5)</td><td>Side (5)</td></tr>
</table>

Time ↓

Table 9: Example of the specification and results of three different window sizes for a single body part (the amount of the translation is given between brackets).

In Table 9, the first and last two exercise elements have a short duration, and will be matched against the short window. The second element lasts longer, and will be matched against the long window. Strictly looking at matching window size causes problems when the exercise is performed too fast or slow. To (partly) overcome this problem, the matching detection window gets an increased weight, but the other detection windows are not ignored. This means it is easier to get a match from a detection window that has the same length as the specification element, but when a longer detection window has a much better match it will still win. In Table 9 this can be seen for the second element: the long window, which matches in length, has the right direction but a much higher amount. The middle window, however, makes a perfect match, and is therefore counted as a successful performance of this element of the specification. If no match can be made, the exercise performance lacked the required translations. In this case, the translation with the highest amount at the moment of the failing match is counted as the action the user performed instead. In Table 9 the specification element "up" is not found in the detection results, the highest detection result at that time was "middle", thus the user kept the body part at the same height, instead of moving it up.

In the previous example, comparisons are made based on directions. In practice, a continuous direction indication in angles is used, because this makes comparison much easier (the difference between "forward" and "forward left" is vague, whereas the difference between 90° and 45° is perfectly clear). Figure 35 shows an example with at the left a graph with two translations (three 2D position samples), and at the right the average of these two translations. The amount for the average translation is calculated via the Pythagorean Theory: $\sqrt{(x_1 + x_2)^2 + (y_1 + y_2)^2} = a_m$, and the angle is calculated by averaging θ_1 and θ_2.

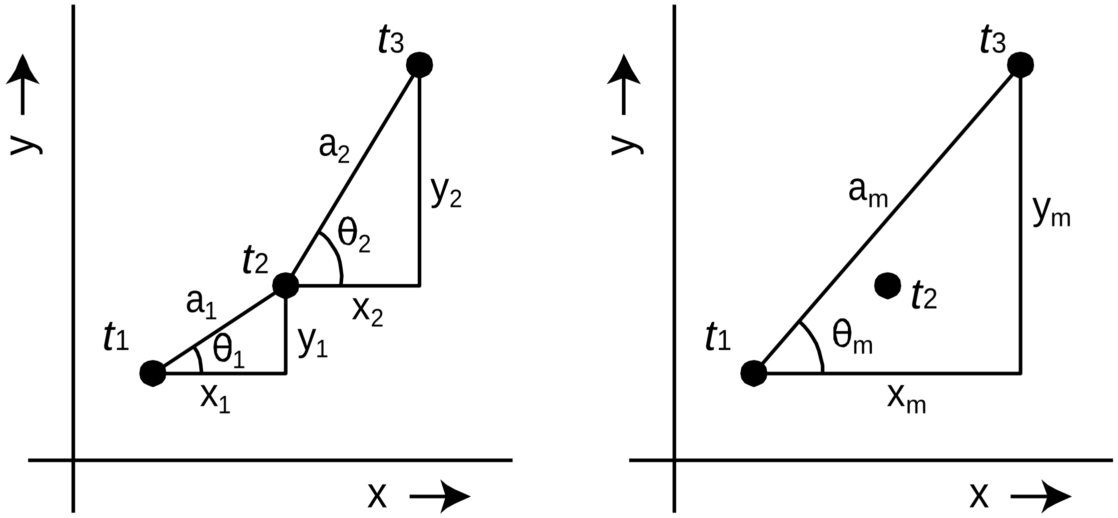


Figure 35: Schematic graph showing three measurement points on the left and the average of those points on the right.

When the total amount and average angle of the required number of detection windows is calculated, these can be compared with an element from the exercise specification playlist. If the amount and angle of the detection and specification match exactly, this is counted as a 100% match, and if the difference is greater than a predefined threshold, the match "quality" is 0%.

There are two methods to calculate the match quality, the absolute difference between the specification and measured value, and the distance to zero. The absolute difference amount is calculated by $|a_m - a_s|$ (see Figure 36). The specified amount is calculated via multiplying the duration of the exercise specification element with the predefined "normal" speed of the user performing the exercise. The absolute difference in angle is calculated by $|\theta_m - \theta_s|$ (keeping into account that 0° = 360°). The specified angle is based on the definition of the specified direction (see Figure 32 and Figure 33).

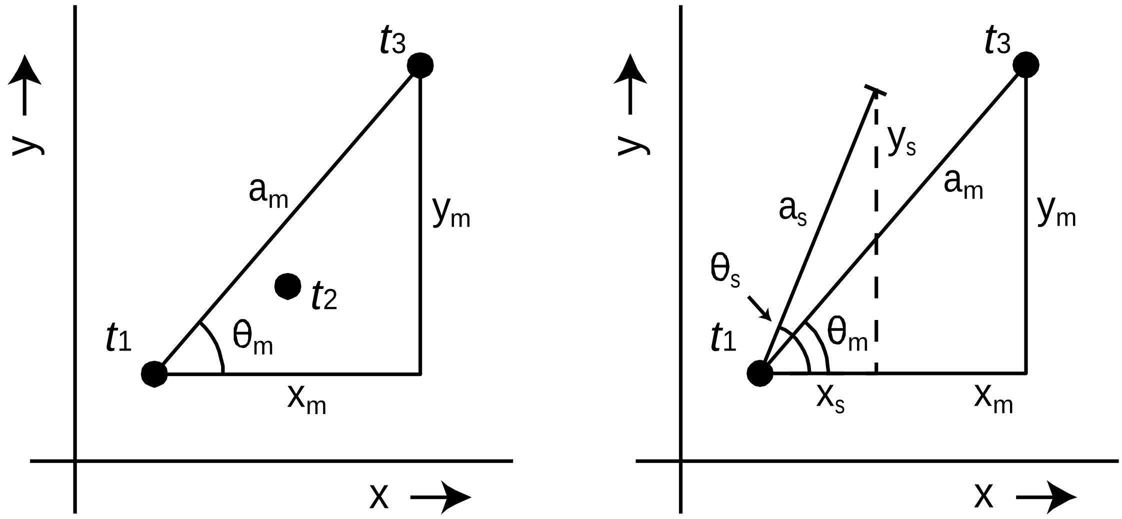


Figure 36: Schematic graph showing the average measurement of three points on the left and the average of those points on the left, and the specified point on the left.

The conversion from an absolute amount difference to the match quality in percentages for horizontal and vertical translations is:

$$match\ quality = \ 100 - \left|100 - a_s / \frac{a_m}{100}\right| \qquad \{1\}$$

this equation always results in a value between 0 and 100%, independent of whether the specification is larger or smaller than the detection result.

A different approach is needed when the specified amount is zero (keep body part static), because it is much easier to remain static than to move in a specific direction with a specific speed. For translations with a specified amount of zero, calculation of the match quality is based on a predefined maximal distance to zero (MDZ). If the measured result is above this maximum distance, the match quality is 0%, if it is actually 0, the match quality is 100%. This results in the following calculation (a_s=0):

$$match\ quality = \ 100 - \frac{MDZ^{-1}}{|a_m|} * 100\% \qquad \{2\}$$

Calculation of the match quality of the difference in angles is based on a predefined maximum difference between the specified and measured angle. If the difference is larger than this threshold (MAD), the match quality is 0%, if the two angles are equal, the match quality is 100%:

$$match\ quality = \ 100 - \frac{MAD^{-1}}{|\theta_m - \theta_s|} * 100\% \qquad \{3\}$$

Calculation of the match quality of the amount of rotation can be done via the equation above {3}. For the rotation, the angle represents the amount of rotation. The direction of the rotation can only be "left" or "right". If the direction of the specified rotation does not match the measured direction, the performance is incorrect, and the match quality is always set to 0%.

The match quality calculation used for the amount of passive translations (equation {2}), can also be used to calculate the match quality of a relative translation when the horizontal direction is "place" or vertical direction is "middle". In this case a_m is substituted by the difference in position of the current and relative body part. For the other directions the difference in angle between the specification and measurement is used to substitute a_m resulting in an equation similar to {3}. The vertical directions "up" and "down" act as a condition, which is either met (the current body part was above / below the relative body part) or not met. Therefore, when the vertical component has the direction "up" o "down" and this condition is not met, the match quality will be 0%. Because of this binary concept, it is not wise to define only a relative translation with only a vertical component. It should always be combined with a horizontal component, or another type of translation.

5.3.5 Determining the position in the exercise specification playlist.

The previous section (5.3.4) explained how the match quality between a single element of the specification, and the detection results for a matching window length can be calculated. Such a calculation only makes sense when the position in the exercise playlist, and thus the progress of the exercise is known. Unfortunately, determining this position is not straight-forward. The two main difficulties are:

A. What to do if elements from the exercise specification playlist are not detected?
B. When can the matching advance to later elements in the exercise specification playlist?

Because the exercise playlist is chronological, the matching will start at the first element of the body part in question. Figure 37 shows the steps in this match process, including the actions taken when difficulty “A” arises.

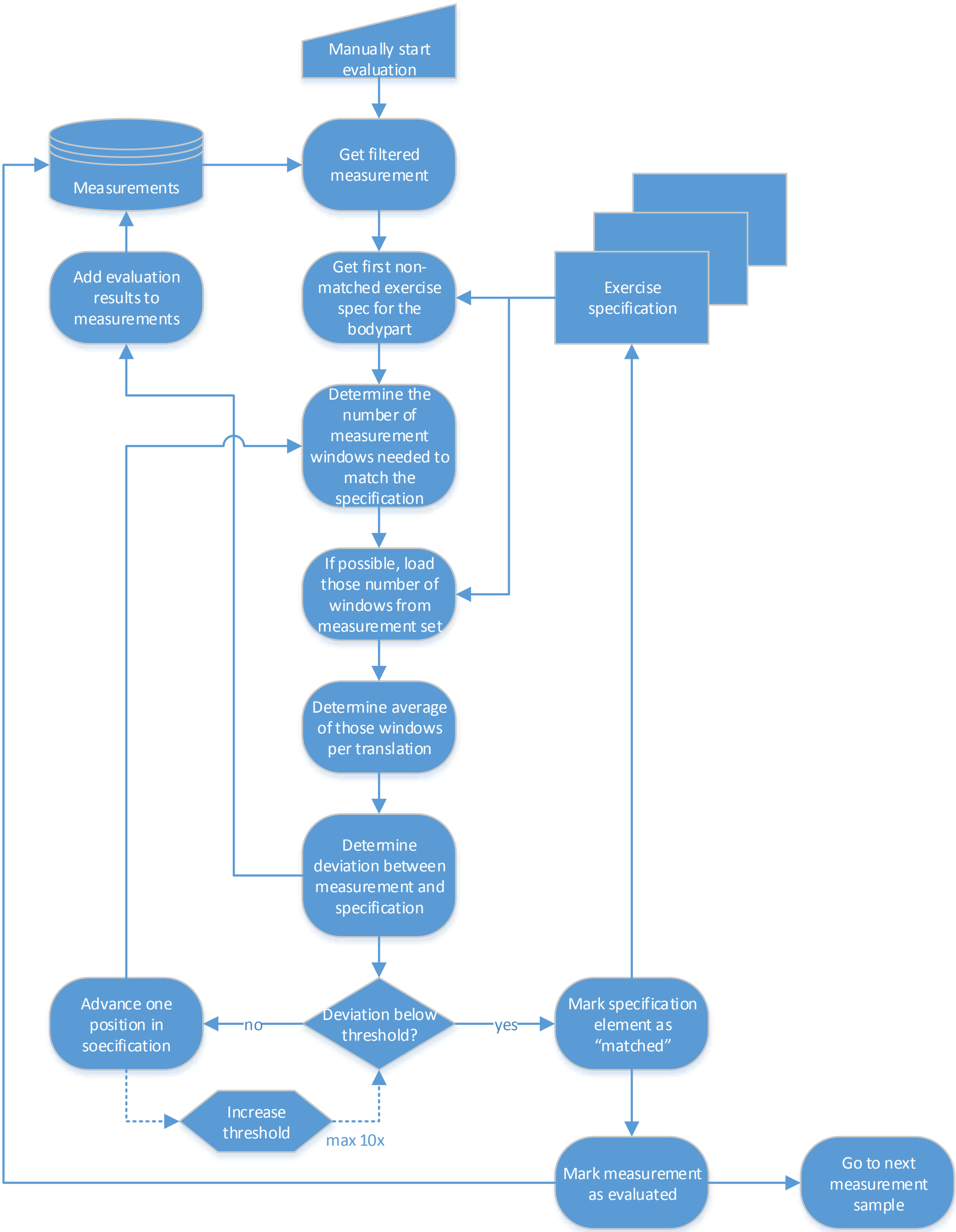


Figure 37: Match process for a single measurement sample.

Although it is possible to start right away, random actions by an unprepared user might accidently match part of the exercise, and cause troubles with the further matching process. Therefore, the first step in Figure 37 is to manually start the matching algorithm. The second step is to load the current measurement sample. This sample contains the filtered translations, and positions for a single body part. Third, the first non-matched specification element for the body part in question is loaded from the exercise specification playlist. It can very well be that this is not possible, because not each relevant body part is active at every moment during the exercise. If this is the case, the matching process is halted, and the system instantly goes to the last step: “Go to next measurement sample”. If the specification element loaded successfully, the fourth step is to compare the duration of the specification element to the window length of the current measurement sample, in order to calculate the number of measurement samples needed. Next, the system will check if this number of measurement samples is available, and if these samples are not already used for a previous successful match. The fifth step is to calculate the amount and direction over the multiple measurement samples, as discussed in paragraph 5.3.3. The sixth step is to compare the amount and direction of the measurement with the specification element, as discussed in paragraph 5.3.4. In the seventh step, the average of the different match qualities for direction and amount of multiple translations is compared to a predefined match quality threshold. If the threshold is exceeded, a “match” is made, and both the specification element and the measurement sample are marked as “matched”. The last step is then to load the next measurement sample and start all over again. This is repeated till the latest measurement samples are processed of all relevant body parts.

If the threshold is not exceeded, difficulty “A” took place: the current specification element could not be detected. This can mean four things:

- the specification element was not yet performed
- the performance of the specification element was not correctly detected (missed by system)
- the performance of the specification element was incorrect (error by user)
- the performance of the specification element was omitted (forgotten by user)

For the first reason, the system should just wait until the specification element is performed. But if the system waits, while the missing element was caused by one of the last three reasons, it will never try to match the rest of the exercise. To prevent stalling in one of these last three situations, the system will continue with the same measurement sample, but now try to match this sample to the **next** element in the exercise specification playlist. Thus it will again perform step three till six, but now with the next specification sample. In step seven, the matching threshold is increased, to make it harder to get a match on the next specification element. This is done to prevent matching on a false part of the exercise, because it is assumed that it is more likely that the user forgets a single step, than multiple steps. After predefinable number of attempts to match the measurement sample, with each time a decreased chance, this process is stopped, and the next measurement sample is loaded. As consequence, the system will fail to find a match if multiple steps were omitted by the user, or when only a few steps were missed, but the steps afterwards were performed mediocre (with a decreased matching chance, a higher performance quality is required before a match can be made).

The result of the matching process described above, is a match quality, a match indication (match or no match) and a position in the exercise specification playlist at which this match was made. If all body parts made a match at the same position in the exercise specification playlist, it’s obvious that for the next measurement samples, also the next position in the specification playlist should be used. But if only a part of the body parts got a match, advancing in the playlist is not obvious. It's evident that all body parts should be at the same position in the exercise playlist, thus some "broker" must decide when to advance the position. The way to handle this problem (Issue "B" on the previous page), is depicted in Figure 38.

Figure 38: Decision tree for advancing in the exercise specification playlist.

The process in Figure 38 starts as soon as a match quality is determined for every relevant body part. The process will load all match qualities, and sums these per position in the exercise playlist (sequence number). This gives a match quality per sequence number, which is compared to the total obtainable match for this sequence number (the number of active body parts is not constant in the exercise playlist, nor is the total obtainable match). This comparison results in a match percentage per sequence number. If none of these match percentages is above a predefined threshold, nothing happens, and the system continues to try to make a match at the same position in the exercise specification playlist as before. If the highest of the matches is higher than the predefined threshold, this is a “global” match. This means that for the next measurement samples, the system will try to match these with the specification elements at this sequence number (position) in the exercise specification playlist. If the match was made for all body parts at the same position in the exercise specification playlist, it was a “perfect” match. Otherwise, the exercise specification elements that were not matched are marked. Because the process will advance to the next position in the playlist, these elements will never be matched. The largest translation present in the measurement samples of the body parts of these missed elements is marked as well. These translations are the most prominent movements the user performed (or the system falsely detected) at the moment of the missed elements. As such, these movements can be considered false, and included in the evaluation of the exercise performance, or in the real-time feedback to the performer.

6 Implementation

6.1 Introduction

In chapter 4 a method is described to parameterize an exercise by first defining the relevant body parts, then divide the whole exercise into segments of a specific duration, and for each segment describe the movements of each relevant body part in terms of horizontal and vertical translations and rotations. Chapter 5 gave a method to convert the parameterized exercise into an exercise playlist (section 5.2). It described how to convert joint positions, measured by the Kinect, into the translations used in the parameterization (horizontal, vertical and rotation) (section 5.3.3). Next a method was given to compare a single measured translation to an arbitrary element from the exercise playlist (section 5.3.4). At the start of the exercise, this comparison can be made between the first measured items, and the first items from the exercise playlist. Section 5.3.5 dealt with the difficult issue on when to advance to later items in the playlist. It's not practical to only advance if all previous items have been matched, because the detection might miss movements, or the user can forget a part of the exercise. Therefore, advancing in the playlist can be done when the larger part of the previous items could be matched, or when the current measured translation form a very good match to a future part of the exercise playlist. The exercise specification elements that were missed, are the movements the user failed to make, and are marked for evaluation.

This chapter describes how the concepts from chapter 4 and 5 are implemented into detection and evaluation framework prototype to test the feasibility of the concept. The detection framework is based on the "Kinect Toolbox" by David Catuhe (Catuhe, 2013). This toolbox is extensively documented in the book "Programming with the Kinect for Windows Software Development Kit". The toolbox shows the RGB image with a skeleton overlay (if available), it has the functionality to detect postures and gestures, and enables recording and replay of Kinect data. In the toolbox, two namespaces are added: `Exercise` and `ExerciseDetection`. The namespace `Exercise` contains the classes that are used to define a static exercise specification (the concepts from chapter 4). These classes are discussed in section 0. The namespace `ExerciseDetection` contains the parts to interpret the exercise specification, record the joint positions, convert joint positions into specific translations and compare (interpret) the translations to the exercise specification (the concepts from chapter5). The implementation of these parts is discussed in section 6.3. Apart from these two namespaces, the only changes to the toolbox are made in the code which controls the main loop of the framework: `MainWindow.xaml.cs`. In this class the initialization takes place, and each time the `SkeletonFrameReady` event is fired, the relevant parts of `ExerciseDetection` are executed.

For development of C# code, Microsoft's Visual Studio 2012 professional 32bit is used. It perfectly integrates with the Microsoft Kinect SDK v1.6, and is one of the most advanced IDEs (Integrated Development Environment) for C++/C# development available for Windows. Other tools of the development environment were: MS Windows SDK for Windows 7 (v 7.1), MS Kinect Runtime v1.6 and MS Kinect Developer Toolkit v1.6.

6.2 Implementation of exercise parameterization framework

In chapter 4 a structure is defined to parameterize an exercise using the Labanotation concept, without requiring extensive knowledge of the Laban movement analysis methods. Section 4.4 proposes a data model to store all parameters which make up the exercise specification (the class diagram is shown in Figure 26 at page 48, the hierarchical data tree is given below, see Table 10). Because this data model is used exclusively to store data, the implementation is straight forward. For each class constructors are made, these constructors force inclusion of all needed elements. For example, to create a `BodyPart` instance, instances of each translation are required, and those must contain an instance of the `Duration` class. A complete specification of a specific exercise is thus one instance of `Exercise`, containing an array of `SegmentSet`, each element of this array contains an array of `Segment`, and each `Segment` contains multiple instances of `BodyPart`.

- **Exercise** | name=(string) | category=(string)
  - **Description**
    - Introduction (String)
    - Performance (String)
    - Attention (String)
  - **Joints** (string) | model=(string)
  - **Personalization** |affected=(string) | side=(string)
  - **Segments**
    - **SegmentSet** | seq=(int) |repeatSet=(int) | nextSegmentSet=(int)
      - **Segment** |id=(int) |repeatSingle=(int)
        - **BodyPart** (String) | side=(string)
          - **HorizontalDirection** (String) |modifier=(string) | accent=(string) | space=(string)
            - **Duration** (int) | constraint=(string)
          - **VerticalDirection** (String) |modifier=(string) | accent=(string) | space=(string)
            - **Duration** (int) | constraint=(string)
          - **HorizontalDirection** (String) |twist=(string) | modifier=(string) | accent=(string) | space=(string)
            - **Duration** (int) | constraint=(string)
          - **Relation** | relative=(string)
            - **HorizontalDirection** (String)
            - **VerticalDirection** (String)
            - **Duration** (int) | constraint=(string)

Table 10: Hierarchical data structure of an exercise specification

Manually creating instances of multiple classes in order to store an exercise specification is only suited for testing of the prototype. Eventually, the system should work together with a site such as CoCo. To communicate the exercise specification, and more importantly, the personalization options, an external data format is better suited. The Extensible Markup Language (XML) is a popular data format to communicate between separate programs. An example XML containing all parameters from the target exercise is made. An extract of this XML is given below:

```xml
<SegmentSet seq="1" repeatSet="0" nextSegmentSet="2">
      &lt;Segment id="0" repeatSingle="0"&gt;
        <BodyPart side="both">
          Support
          <HorizontalDirection modifier="none" accent="normal" space="normal">
            Place
            <Duration constraint="none"> 10 </Duration>
          </HorizontalDirection>
          <VerticalDirection modifier="none" accent="normal" space="normal">
            Middle
```

```xml
      <Duration constraint="none"> 10 </Duration>
    </VerticalDirection>
    <Rotation twist="no" modifier="none" accent="normal" space="normal">
      Neutral
      <Duration constraint="none"> 10 </Duration>
    </Rotation>
    <Relation relative="none">
      <HorizontalDirection> Place </HorizontalDirection>
      <VerticalDirection> Middle </VerticalDirection>
      <Duration constraint="none"> 10 </Duration>
    </Relation>
  </Joint>
```

The XML follows the same hierarchical structure as the data model given in Figure 26 (page 48) and Table 10. An XML reader was not implemented, because this was not deemed essential for this project.

6.3 Implementation of detection and evaluation framework

The implementation of the concepts discussed in chapter 5 can be divided into two stages (see Figure 39). First, the software is initialized (discussed in section 6.3.1). In this stage the exercise parameters are parsed to create the exercise playlist, and suitable parameters such as the sampling window length are defined. The second stage is the "Running" state, which is executed each time new skeleton data from the Kinect is available. The first step of this stage is to check if the skeleton position data needs to be stored, next the joint positions are converted into translations (section 6.3.2), then these translations are compared to the exercise playlist and last, the position in the exercise playlist is advanced if the match between the measured translations and the exercise playlist was good enough (section 6.3.3).

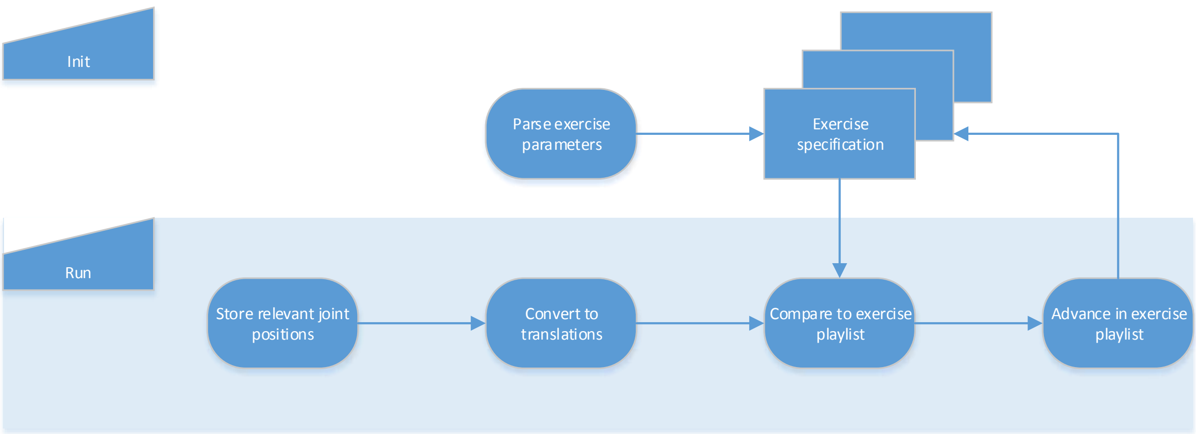


Figure 39: Implementation overview of detection and evaluation framework.

6.3.1 Initialization

The first step of the initialization phase of the detection and evaluation system, is the creation of the `detectionParameters`. This dictionary of `floats` contains all parameters used throughout the system, which need "tuning". The most important one is "Normal speed". This value relates the duration of a translation to the amount, and is essentially the speed at which all movements must be performed. The "Normal speed" is specified as a single value for all horizontal and vertical translations, but can have a different value for rotations. The dictionary also contains a number of thresholds, such as the level below which something is considered zero, the maximum deviation allowed for an amount match etc. All detection parameters can be found in the table in Appendix 10.1, and are also discussed in the description of the class where the parameter is used.

Next, the class `getWindowSizes` looks through the exercise specification, and calculates suitable window sizes to sample the movements. It does so by looking through the duration of each translation type in the exercise specification. The shortest window is the shortest duration divided by the `windowDivider` factor, the longest window is the longest duration divided by the `windowDivider` factor. The middle window is halfway in between the shortest and longest window. Both the medium and longest window are rounded to be a factor of the shortest window.

The third part of the initialization, is the `BodyPartReader`, which looks through the exercise specification to gather which body parts are used in the specification. These body parts are placed in the `RelevantBodyPartList`. Each body part is only placed in this List once. The method takes "affected" and "non-affected" sides into account, and for side "both" it will place both the Left and Right instance of the body part in the list. Furthermore, it substitutes body parts that are not native to the Kinect SDK, with the closest native Kinect SDK body part. For example: “Arms”, “side = both”, becomes: [Wrist Left, Wrist Right] See Table 11 for the non-Kinect joints and their compliant equivalents.

Custom joint	Kinect “compliant” equivalent
Body	Hip center
Upper body	Shoulder center
Lower body	Hip center
Arm	Wrist
Leg	Ankle

Table 11: Reference between custom and Kinect "compliant" joints.

The last part of the initialization is the `exerciseSequenceReader`, a class that converts the exercise specification into an exercise playlist. The main difference between the specification and the playlist is that the latter has a specific incremental sequence number for each element in the list, and that segments which are supposed to be repeated, are placed in the playlist multiple times.

6.3.2 Position and Translation recording

As soon as a person is recognized by the Kinect SDK, it fires `SkeletonReady` events. Such an event indicates that new skeleton data is available. The skeleton data is processed in the Kinect Toolkit method `ProcessSkeletonFrame`. This method is called roughly every 33ms, corresponding to the sample frequency of the Kinect depth camera. The class `WindowTiming` generates a slower clock to reduce the sample frequency of the detection and evaluation system. Each time `WindowTiming` indicates a window length has passed, the following classes are executed sequentially: `WindowSampler, TranslationSampler, ResultWeighter and ResultInterpreter` (see Figure 40). A short description of the first three classes is given in the following paragraphs, together with the `WindowTiming` class, section 6.3.3 will discuss the `ResultInterpreter`.

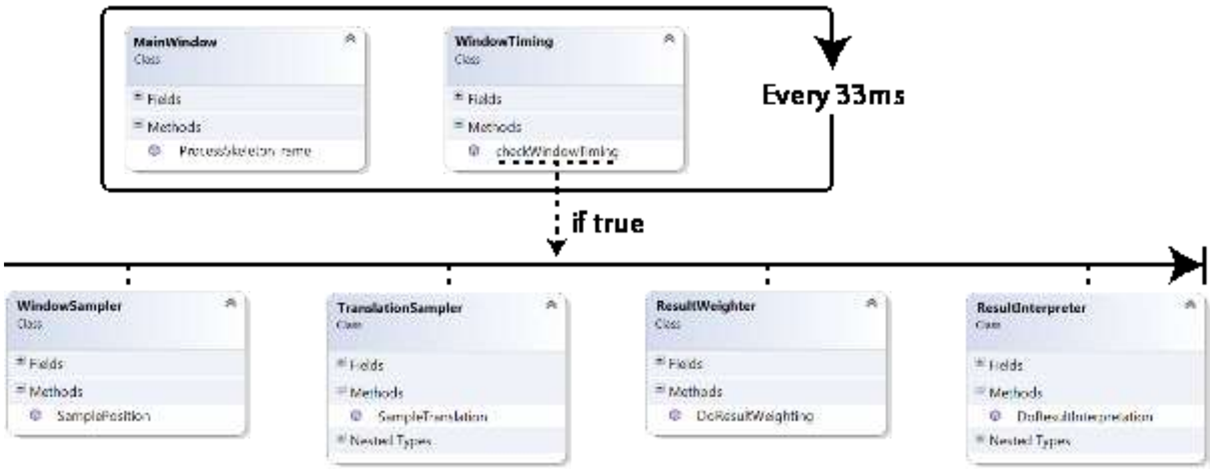


Figure 40: Class overview of the "Run" phase in the MainWindow.xml.cs

Window Timing

The sample rate of the Kinect is 30fps (33ms), and the shortest window length used in the target exercise is 833ms; consequently, only each 25th sample should be taken into account to calculate the translation. Unfortunately, the sample rate of the Kinect is not guaranteed, and will vary due to system load and other external factors. For a non-constant sample rate, a reliable external timer is needed to get a constant interval between two samples that are used to calculate the translation. `CheckWindowTiming` is the method that keeps track of the time. Each time the class is executed, it will check if the current time is a "WindowSize" from the previous time (or from 0). When a "WindowSize" has passed, it will return true, otherwise false. If the time passed, is longer than the "WindowSize", it will substract the time "overdue" from the next "WindowSize", to prevent drift. Only the shortest window has to be taken into account, because the longer windows are a multiple of the shortest window.

Window Sampler

Window sampler is a simple class that stores the current position of a single body part. For each sample, it creates a new instance of `ExerciseEntry`, to store joint position, time, sequence number and body part. It also drops the oldest positions, if the `ExerciseEntries` array grows beyond 1000 items.

Translation Sampler

The Translation Sampler, calculates the difference between two position measurements of a single body part. It automatically starts at the latest sample, and goes back, until all samples (body parts) that belong to the last measurement time are processed. For each sample, checks if the previous sample for the same body part is present. This information is used during the result interpretation, when multiple samples are combined to form a longer window. For each sample, a new instance of `TranslationEntry` is created, and the translations and offset are stored in this entry. It drops the oldest positions, if the `TranslationEntries` array grows beyond 1000 items.

Result Weighter

The `ResultWeighter` class adjusts the weighting for the results from all detectors and all body parts. Currently, two types of weighting are implemented: it increases the weight of constant translations, this means that a translation at a constant speed and constant direction is amplified, and it sets the amount of translations which are close to zero at exactly zero. The `resultWeighter` class is a good place to implement more filtering steps in the future, such as a lowpass filter.

6.3.3 Result interpretation

Result interpretation is the process in which the detected translations are compared to the exercise specification. Figure 41 shows the relevant classes of the `ResultInterpreter` including the two important loops. The outer loop, loops trough all uninterpreted translation samples. The inner loop, loops through the exercise playlist to compare a single measured translation to multiple elements from the exercise playlist.

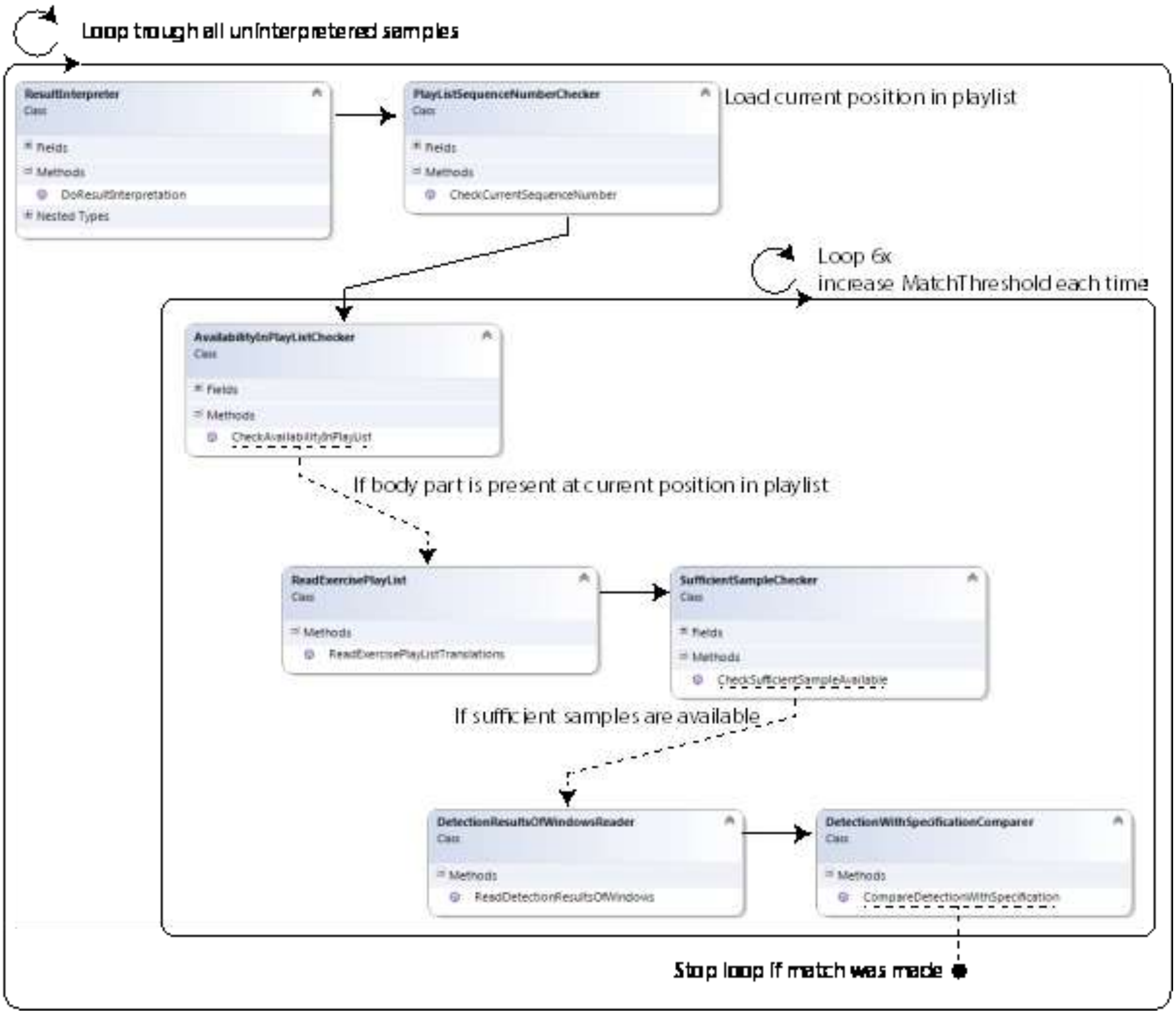


Figure 41: Class overview of the ResultInterpreter

The `ResultInterpreter` starts with the latest item of `EntriesTranslation`. Each item of `EntriesTranslation` contains a single translation of a single body part. The `ResultInterpreter` loops back through the `EntriesTranslation` list, until it reaches samples that have the tag `ResultInterpretation` set to “done”.

The first thing the `ResultInterpreter` does, is to check the current sequence number. The current sequence number indicates at which position in the exercise the patient is. This number is calculated by the `PlayListSequenceNumberChecker` class based on an array containing the match quality at a specific sequence number for a specific body part. If enough body parts have an increased sequence number, the sequence number for all body parts is increased, thus the system advances through the exercise specification. It keeps into account that not all body parts need to play an active role at a certain time in the exercise.

When the current sequence number is known, the `ReadExercisePlayList` will load the specified translations belonging to the body part of the current `EntiresTranslation` sample. These translations, Horizontal, Vertical, Rotation and Relation, all have a direction, angle and duration. The amount is calculated based on "normal speed" of the user.

But not all, or even none, of the translations have to be active (not every relevant body part, performs relevant actions all the time). Therefore, the `AvailabilityInPlayListChecker` checks if the body part is present at the current sequence number, and if at least one of the translations is active. When this is the case, the longest duration of the active translations is divided by the window size, to get the number of windows needed to be able to evaluate a translation over the given duration. If enough uninterpreted samples are available to fulfill the specified duration, the `DetectionResultsOfWindowsReader` will load those samples, and calculate the total amount and average direction (see section 5.3.3). The `DetectionWithSpecifcationComparer` compares this measured total amount and average direction to the specification loaded by the `ReadExercisePlayList` to calculate the match.

The matches for each translation type, are averaged to calculate the total match for the current body part. If this total match is above the match threshold, the loop is stopped, and the `ResultInterpreter` will continue with the next measured translation.

If the total match is below the match threshold, the same measured translation will be compared to the next element in the exercise playlist, and the match threshold is increased. After 6 times, or when the last element of the exercise playlist is reached, the loop is stopped as well.

7 Evaluation of the automated detection and evaluation system

7.1 Introduction

A series of experiments has been carried out to evaluate the performance of the system. The results of these experiments are analyzed in order to give both a quantitative and a qualitative outcome. Because the system is far from complete, it will have (known) gaps in functionality, and sub-optimal performance. In this chapter, an indication of the impact of these gaps is given, but more importantly, the performance of the features that are functional is quantified and discussed. First, the performance indicators are discussed, then the measurement protocol is given, followed by a data analysis of the measurement results. The chapter is concluded by a conclusion and discussion.

7.2 Performance indicators

Quantitative outcome measures require to have tangible performance indicators (PI). These indicators are organized into categories which match the chronological “flow” of the whole process. In the experiment results analysis, these performance indicators will be analyzed for multiple recordings of the same exercise, performed by multiple persons, in multiple ways. The categories of performance indicators are: Exercise, Parameterization, Measurement and Software. The performance indicators are scored per body part per segment of the exercise. All scoring aspects are listed in detail below, and are numbered for reference in the Appendix 10.6. Not all performance indicators can be determined on body part level of detail, therefore some fields in the table are combined. These indicators are marked with “per segment”.

7.2.1 Exercise

For the Exercise category, the number of deviations between the exercise specification and the user performance is scored. These are incorrect movements made by the subject, either intentionally or by accident. In theory, the automated evaluation should come up with the same deviations.

PI 1. Count the number of times a body part is moved such that it would result in a different Labanotation.

7.2.2 Measurement

All errors made by the Kinect depth camera, and accompanying SDK are scored in the Measurement category. Of the aspects measured within classes of the automated detection and evaluation system, those class names are indicated. Nevertheless, the cause of these errors will often lie within the Microsoft Kinect SDK and not in the listed class.

PI 2. Count the number of times the movement of a relevant body part is not correctly calculated by the Kinect SDK skeleton model. A body part is incorrect when its false position would result in a different Labanotation.
PI 3. Count the number of relevant joints that are inferred (this means an uncertain position is given for this joint).
PI 4. Count the total number of joints that are inferred.
PI 5. WindowTiming: number of samples of which the jitter is higher than the window length (per segment).
PI 6. SkeletonTracking: number of times multiple persons are recognized. (per segment)

PI 7. TranslationSampler: number of times the previous sample of same body part cannot be found. This indicates dropped or unmeasured body part samples (per segment).

PI 8. TranslationSampler: the average duration between two consecutive samples of the same body part (in ms) (per segment). (window duration is set to be 833 ms in these experiments)

PI 9. TranslationSampler: the maximum duration between two consecutive samples of the same body part (in ms) (per segment).

PI 10. TranslationSampler: the minimum duration between two consecutive samples of the same body part (in ms) (per segment).

7.2.3 Processing

The last category is processing. Per class of the automated detection and evaluation system, a score of its performance is given, in respect to the intended behavior.

PI 11. ResultInterpreter: number of times there were not enough uninterpreted samples to match the specified duration (per segment).

PI 12. ResultInterpreter: number of times the `maxFutureIterations` is reached (per segment), `maxFutureIterations` is the number of steps of the exercise playlist the `ResultInterpreter` is allowed to look into the future to try to find a match.

PI 13. ResultInterpreter: number of times the `matchThreshold` is increased (per segment), the `matchThreshold` is increased each time the `ResultInterpreter` advances one step into the future of the exercise playlist to try to find a match.

PI 14. ResultInterpreter: number of times the indicated "strongest translation" matched the performed translation when a part of the specification could not be matched. Only indications of an active movement are taken into account. (count negative if a "strongest translation" is given incorrectly. If there were both correct and incorrect indications, only the correct ones are counted.)

PI 15. ResultInterpreter: number of times an indicated match was correct. (count negative if an indicated match was incorrect, if there were both correct and incorrect matches, only the correct ones are counted.)

7.3 Experiment protocol

Before starting the experiments, the tests subjects have to read and agree to the patient information letter, which can be found in Appendix 10.4. Among others, the inclusion criteria are stated in this letter: above 18, proficient in Dutch, and without physical handicaps.

The protocol for the experiment consists of six performances of the target exercise given in section 3.4 (page 35). For the first performance, a very short explanation is given on paper (see Appendix 10.5.1)

> *Start in in normal stance, with both feet next to each other, then swing your right leg along your left leg. Next, move your hips towards the left, while at the same time move the upper body towards the right.*

Before the start of the second performance, the test subject is given the full explanation of the CoCo exercise description, also on paper (see Appendix 10.5.2). After reading this, the example video is shown to them. In this explanation, the references to the supporting chair are removed, because the chair can interfere with the Kinect depth sensing qualities. Furthermore, "affected" side is replaced by "left", and logically "un-affected" by "right".

In the last performance, the test subject is asked to perform the same exercise in the same manner, but move their upper body to the right instead of the left.

Each of these three performances is done twice, the first with instruction to perform the exercise at an easy speed, the second at a bit higher speed. These arbitrary speed instructions will result in exercise performances at different speeds, which can be used to test the algorithms for robustness to change in exercise duration. Because duration is not normalized, this is expected to be of influence to the detection and evaluation quality.

The test subject will perform the exercise while facing the Kinect depth camera. The distance between the camera and the front of the shoes of the test subject is 3 meters. The recording is made via the built-in recording feature of the Kinect Tool Box used throughout this project. This functionality generates data which is proprietary to this program, thus the datasets cannot be interpreted by third party software such as the Kinect toolkits for Matlab. The recording functionality is used despite these limitations, because the recordings can be fed to the implemented automated detection and evaluation system without any effort.

7.4 Analysis

The recordings of each experiment are replayed in the Kinect Toolkit, with all detection and evaluation disabled. Because there is no simple way to convert the Kinect video data including the skeleton overlay to a video file, the screen is captured via a screen recorder (Camtasia Recorder 8 (TechSmith, 2013)). The skeleton overlay generated by the Kinect Toolkit has some issues. The first problem is a lack of synchronization between the RGB video stream and the skeleton model. The skeleton overlay has a delay, which is related to the amount of processing power available. To limit the delay, all processing has been disabled while making the screen recording. The second problem is a scaling offset between the video and skeleton model. Both problems can be seen in Figure 42, while this capture was made, the right hand moved to the left fast, and the skeleton representation is lagging behind. It can also be seen that the shoulders are drawn significantly below where they appear on the RGB image. In general the skeleton model is slightly smaller than the RGB image. The skeleton and RGB image are aligned at the feet, resulting in the largest offset at the shoulders and the head. These two problems are only relevant for the visualization; the position and timestamp of the coordinates of the skeleton model are correct.

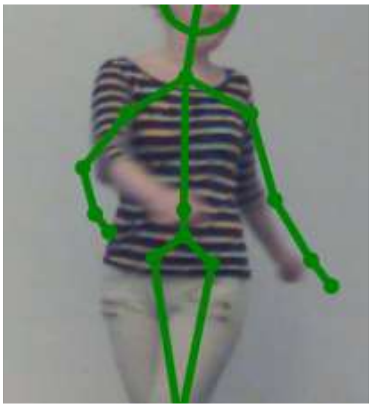


Figure 42: Frame recorded by a screen recording, showing both a scaling offset and a delay between the RGB image and the skeleton overlay (This is a frame from a session with all processing enabled. The screen recordings used for manual annotation do not show this much delay, because most processing steps are disabled).

A logger is implemented to analyze the performance of the detection and evaluation system. This logger writes debug information to a text file with the following structure:

```
time stamp; method name; result 1; .. ; result n;
```

The timestamp is based on the current system time, not the time of the recording. Via annotation of the captured videos, the time of the following moments is determined: start of the recording, start of segment 1, start of segment 2, start of segment 3 and end of segment 3. The start of the recording is also marked in the log file, thus the time stamps can be used to filter the log file for events that happened during a specific segment.

Performance indicators 1 and 2 listed in paragraph 7.4 are determined by manual annotation based on the screen recording. To perform the annotation, the screen recordings with skeleton overlay were analyzed frame-by-frame. The position of the skeleton overlay was compared to the actual position of the body part, but only for the 6 relevant body parts, position errors in other body parts were ignored, because these are not used in the detection and evaluation system.

The rest of the performance indicators is given via filtering of the log files for the specific keywords. Performance indicator 4 uses a free format, in which the incorrect body parts are noted by abbreviations (see Table 12 for the abbreviations used). For performance indicator 14 and 15, the log result is compared to the video via manual annotation based on the screen recording.

Joint	abbr.	Joint	abbr.
Spine	Sp	Hand Right	HaR
Head	He	Hip Left	HL
Shoulder Left	SL	Knee Left	KL
Elbow Left	EL	Foot Left	FL
Hand Left	HaL	Hip Right	HR
Shoulder Right	SR	Knee Right	KR
Elbow Right	ER	Foot Right	FR

Table 12: Abbreviations used to define the non-relevant body parts in the results.

7.5 Results

All 8 subjects successfully completed all 6 performances of the experiment protocol. The subjects did not make the correct movements based on the short description (session 1 & 2). However, three subjects were already familiar with the chosen target exercise, and made no false movements during session 1 & 2. These three subjects are marked in Table 13, together with the ID, length and sex of all subjects.

ID	Length	Sex	Comments
1	160	F	
2	191	M	Was already familiar with the exercise
3	185	M	Was already familiar with the exercise
4	202	M	
5	192	M	
6	173	F	
7	175	F	Was already familiar with the exercise
8	178	F	

Table 13: Information on experiment participants.

7.5.1 Segment duration

As stated in section 7.3, the experiment consisted of three different exercise descriptions: a deliberately vague description, a complete description and a mirrored complete description (see Appendix 10.5.2). Each subject performed each description twice, first at normal speed and second a bit faster. Both normal and faster are subjective instructions, and therefore some deviation in performance speed can be expected.

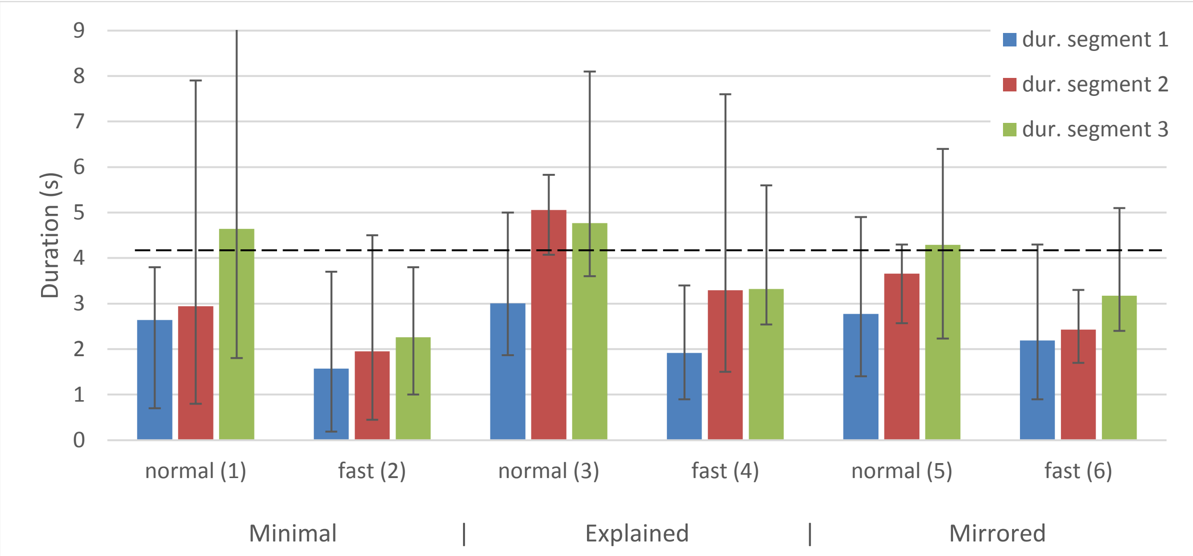


Figure 43: Average segment duration of the six experiments for the three segments. The black error bars indicate the minimum and maximum durations.

Figure 43 shows the average duration of each segment over all 8 subjects, for each experiment session, measured via manual annotation of the video recordings of each experiment. It can be seen that each second performance was performed faster. See Table 14 for the average performance speeds per segment and in total for the "normal" and "fast" performances.

The horizontal line at 5 seconds in Figure 43 is emphasized because the individual segments have a parameterized duration of 5 seconds in the target exercise specification. Unfortunately, the average performance duration of a segment was less than 5 seconds in all but one case. Segment 2 of experiment 3 was the only performance that matched the parameterized duration of 5 seconds with a relatively small margin of error (average 5.06 with a variance of 0.38). In general both the normal and fast performances were too short, and the variance was very high (see Table 14). Because the system doesn't normalize for changes in performance speed, this has a negative effect on the detection performance.

performance speed	**normal**	**fast**
average segment 1	2.8 (1.2)	1.9 (0.9)
average segment 2	3.9 (2.4)	2.6 (2.3)
average segment 3	4.6 (6.5)	2.9 (1.0)
sum	**11.3 (12.7)**	**7.4 (7.1)**

Table 14: average segment durations (variance between brackets) for normal (session 1, 3 & 5) and fast (session 2, 4 & 6) performances.

The short duration of the segments not only disturbs detection of movements, in some cases it even makes evaluation on a segment level impossible. When a segment duration is shorter than the sampling window length, there is a chance that nothing was measured during the segment. Experiment session two by subject 3, 5 and 6, and session 1 by subject 6 have segment durations shorter than 833ms. Due to lack of samples, the result tables in the Appendix show blanks for these sessions.

7.5.2 Performance indicators

In section 7.2, 15 performance indicators are listed. All the results of all 15 performance indicators are listed in the Appendix 10.6 for each subject, each session, each segment and each relevant body part (for some). Table 15 lists a results table for a single session. In the top left the subject is listed ("S1" = Subject 1), the session number is listed below the subject number, next to the body part indicators (B = Body, AL/AR = Arm Left / Right, LL/LR = Leg Left / Right and UB = Upper Body). The 9 rows are repeated six times, for the six sessions. The rows marked 1 till 4 contain the first four performance indicators, which are given per body part. Performance indicators 14 and 15 are marked per body part as well. Performance indicators 5 till 13 are listed in a single row, because these are only given per segment. Each field in the table is colored green or red, green indicates a good result, red indicates a bad result.

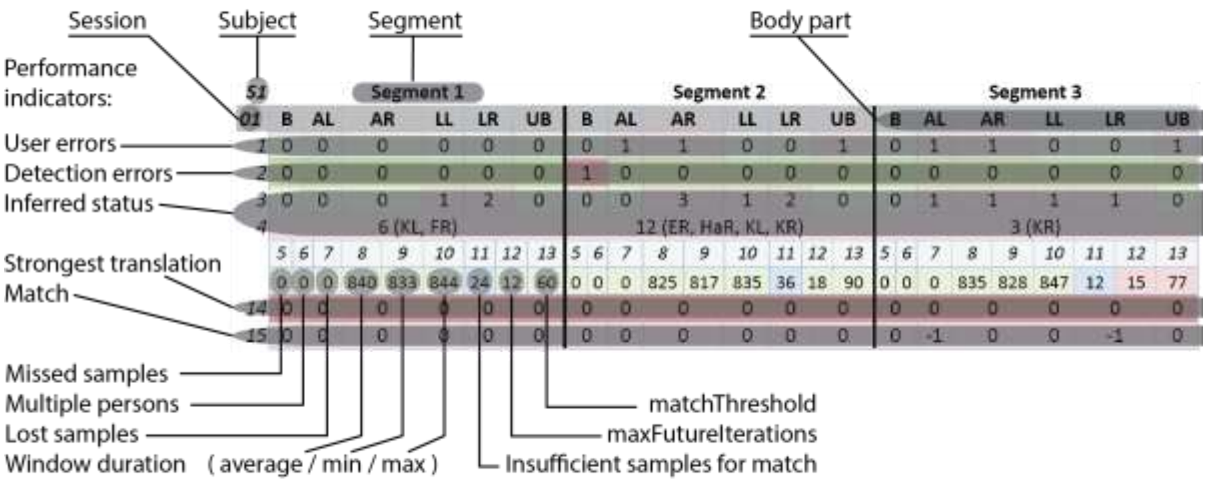


Table 15: Extract from result table, subject 1, session 1.

Performance indicator 1 represents the number of times a body part is moved such that it would result in a different Labanotation. These are errors made by the performer. Obviously, these errors are present in the first two sessions for subjects who were not familiar with the exercise, because of the deliberately vague exercise description. The movement of the upper body also differs from the target exercise in sessions 5 and 6, because the subjects were asked to bend their upper body left instead of right. Notable errors are wrong movement of the right arm of Subject 1, session 4, and of the left arm of Subject 8, session 3, both to regain balance.

Performance indicators 2, 3 and 4 represent errors made by the Kinect SDK. Indicator 2 is determined by subjectively analyzing the videos with the skeleton overlay. Indicator 3 and 4 represent the number of samples that are reported to be inferred by the Kinect SDK.

The results show that the Kinect SDK is better able to track the legs during Segment 1 than during Segment 2 and 3. In other words, the Kinect SDK is better able to follow the legs when these are going from normal to crossed, than when the legs remain crossed, or return to the normal position. In general, Segment 2 has a much higher number of tracking errors than Segment 3, although the movements and durations are comparable (bending of upper body). In Segment 2 the legs remain crossed, which apparently lowers tracking performance for the other body parts as well (upper body, arms).

Even though indicators 2 and 3 both represent position errors of relevant body parts, the different assessment method gives different results. In most cases, the manual annotation and the Kinect SDK agree, but in a few cases they contradict. For example session 3 of Subject 3 had a visually error-free skeleton, but 8 relevant samples were indicated to be "inferred". Session 3 of subject 8 shows the inverse, visually the skeleton did not track the movements correctly, but the Kinect SDK did not report tracking issues. Due to these contradictions, the "inferred" status reported by the Kinect SDK can only be used to predict the chance on tracking errors. This also means the common practice of dropping "inferred" samples is not wise, because these samples can very well be correct.

In general, tracking issues were most apparent for the crossing of the legs, looking at sessions 3 till 6, 11 crossings were correctly detected, and 21 were missed. While bending the upper body, the Kinect SDK confuses the position of the left or right shoulder, but the shoulder center remains fairly accurate. Unfortunately, the Hip Center joint is placed too far towards the bending direction, making it hard to determine the actual angle between the upper and lower body. Indicator 4 shows that the knees and extremities (hands, feet) are often missing. Because this is a common issue of the Kinect SDK, the joints one step more proximal (wrists, ankles) were already chosen to determine the position of the Legs and Arms.

Performance indicator 5 indicates the number of samples of which the jitter is higher than the window length (per segment). These are samples which are measured correctly by the Kinect, but for which the processing took more than a single window length. With a sample interval of 833 ms, none of the measurements show samples of which the jitter is higher than the window length.

Performance indicator 6 indicates the number of times multiple persons are recognized by the Kinect SDK. All performances were carried out with only one the subject in view. The Kinect SDK agrees to this.

Performance indicator 7 indicates the number of times the previous sample of the same body part cannot be found. As stated previously, in order to get some output, the Kinect software was configured such that it did not drop "inferred" samples. Samples with the state "not tracked" would still trigger indicator 7, but none of the measurement showed such samples.

Performance indicators 8, 9 and 10 respectively give the average, minimum and maximum duration between two consecutive samples of the same body part (in ms). The timestamps stored together with the Kinect joint positions are based on the current system time at the moment of processing. If there is insufficient capacity, this processing will take longer, delaying the timestamp given to the Kinect data. For prerecorded data, such a delay causes a shift between the relative time on which the sample was measured, and the time of the time stamp given to the sample. This shift causes an over or under estimation of the duration between two samples, and thus an under or over estimation of the movement speed. Four out of the 48 experiments had *minimal* or *maximal* sample interval durations which deviated more than 5% from the intended sample duration of 833 ms. Subject 7, session 4 was the only performance for which the *average* sample interval deviated more than 5%.

Performance indicator 11 lists the number of times there were not enough "uninterpreted" samples to match the specified duration. When the system finds a match between the measured translations and the specification, these translation samples are marked "interpreted". Thus a lack of "uninterpreted" samples can only happen at the start, or after the system found a match. If the system works correctly, it will find a match at every segment (thus once per 5 seconds) for every body part. A segment lasts 6 window durations (6 x 0.8 = 5s) of which the first 5 would trigger PI11 because there are not enough "uninterpreted" samples yet. Therefore, PI11 would indicate 30 per segment if the system would function perfectly. During the experiments the system did not function perfectly. The results show many cases in which there was a lack of "uninterpreted" samples during the last part (Segment 3), even though not a single match was made. Performance indicator 5 indicates no samples were dropped, so the samples are actually available. The occurrence given for indicator 11 is always a multiple of 6: the number of relevant body parts. Furthermore, if there are no matches, it correlates well with the segment duration (occurrences ≈ (segment duration / window length) x 6). It is apparent that either the system that marks samples as "interpreted" or the system that checks if enough "uninterpreted" samples are available is malfunctioning. More research is needed to indicate the exact source.

Performance indicators 12 and 13 give two numbers which indicate the functioning of the Result Interpreter. If no match can be found between a measured translation and the current specification element, the system will try to match between the same measured translation and the next specification element. But each time it goes to a next element, the chance of a match is lowered. Indicator 12 gives the number of times this chance is lowered. After looking 5 steps into the future, the match is indicated as failed, the number of times this happens is given by indicator 13. Thus, if no match is found, 12 should be five times as large as 13, and if a match is found, this factor should be lower. Table 16 indicates that when matches were found, 21 out 34 times the factor remained 5. This indicates that the system does not stop to look for a match with a future sample, after a match was found. As a result the system will erroneously indicate that the user made both a correct and an incorrect movement at the same time, with the same body part.

	match	no match
factor is 5	21	109
factor is not 5	13	1

Table 16: Four cases for the factor between indicator 12 and 13, a factor of 5 is correct if there was no match, and the factor should be lower when there was a match. The factor is counted for each segment of each session for each subject (total 144).

When the system fails to find a match after looking 5 steps into the future of the exercise playlist, it will indicate the largest translation of the specific body part for evaluation purposes. This indication tells what movement the subject performed instead of the specified movement. Indicator 14 represents the number of times these indicated largest translations correspond with the movement performed by the subject, but only when the subject's movement was not according to the specification. If the movement was according to the specifications, the system should have been able to find a match, and should not indicate the movement under "largest translations". Because most user performances were without errors, most indications are per definition incorrect. The results show negative numbers, because the user performed the movement correctly, and the system counted the movement under "not matched". The largest number of "largest translation" indications is given for the arms, possibly because the wrists (the measurement location for the arms) travel the furthest of all relevant body parts during the exercise.

The last indicator (15) lists the number of correct matches per relevant body part. Just as for indicator 14, the system seems to favor indicating matches for the arms. When a body part is indicated as a match, the system should not look for other matches for the same body part in the same segment. This is in agreement with the results, which never show more than one match per body part per segment. Most matches are made during the third segment, and very few during the first segment. In total, only 4 out of the 288 possible matches were successful in the first segment. As a consequence, the system will never proceed through the exercise playlist in a normal manner; it will fail to match the first item in the playlist, and will match later items with a lower chance. Experiments in which the subject remained motionless a relatively long time before starting the exercise, show more matches in the second segment. This indicates the system needs some time to settle. This settling time can be related to the requirement of at least two weighted translation samples for every body part, before the matching process can start. This delayed start combined with the low sample frequency, means the matching starts 1.6 to 2.4 seconds after start of the recording.

7.5.3 Kinect joint tracking status

Analysis of the log files indicates that a significant number of joint samples are marked “inferred”. Common practice is to drop these samples, because they are unreliable. The implementation of the `WindowSampler` is such that it requests a sample at a specific time, and waits a full window length if no sample was available at the time of request. Furthermore, the `ResultInterpreter` stops looking for other body parts as soon as it cannot find enough samples for a specific body part. These two design choices, make the impact of lost samples very high, because it generates gaps of a full window length in the data. Considering that many performances of experiment segments took place in 2 or 3 window lengths, those segments are missed completely when 33-50% of the data was missing. To limit the impact of the inferred samples, the Kinect Toolkit was configured to never drop samples. This means it can use samples which are known to be unreliable. Further analysis was carried out to find ways to deal with the inferred samples. A logger was made that stores the joint status (Tracked, Inferred, Not-Tracked) for each frame for each sample. The frame rate of the raw joint data is theoretically 30 fps, but the logging showed it was practically 27-28fps. This logger was ran on the recording of experiment session 3 for every subject.

Subject	1	2	3	4	5	6	7	8
HipCenter	0	0	0	0	0	0	0	0
Spine	0	0	0	0	0	0	0	0
ShoulderCenter	0	0	0	0	0	0	0	0
Head	1	0	0	0	0	0	0	0
ShoulderLeft	0	0	0	0	0	0	0	0
ElbowLeft	0	0	0	0	0	0	0	0
WristLeft	0	0	0	0	0	0	0	0
HandLeft	1	4	0	0	0	3	0	0
ShoulderRight	2	1	2	0	2	1	1	0
ElbowRight	0	0	1	0	2	0	0	0
WristRight	1	0	2	0	1	0	1	0
HandRight	0	2	2	3	2	3	3	0
HipLeft	0	0	0	0	0	0	0	0
KneeLeft	3	3	3	6	6	2	4	0
AnkleLeft	1	5	3	3	2	4	7	1
FootLeft	3	3	1	2	1	2	3	0
HipRight	0	0	0	0	0	0	0	0
KneeRight	2	3	2	4	2	2	4	2
AnkleRight	4	3	2	1	2	3	3	2
FootRight	2	1	1	3	0	0	2	1

Table 17: Kinect tracking status during experiment 3 for 8 subjects. The numbers indicate the number of times a block of at least one window length (0.8s) was inferred, the bars indicate the portion of tracked samples.

Table 17 shows the tracking status for each body part for all 8 subjects. None of the samples had the status "Not-Tracked". The bars indicate the percentage of samples that had the status "Tracked", for all subjects the following joints were tracked 100% of the time: Hip Center/Left/Right, Spine, Shoulder Center/Left, Head, ElbowLeft and WristLeft. In other words: only 9 out of 20 joints were correctly tracked, which is not a good score. But a joint which is reliable 90% of the time, can be of use, if the inferred samples are distributed evenly. Figure 44 shows an extract of a set of graphs which show the tracking status over time for the HipRight, KneeRight, AnkleRight and FootRight joints.

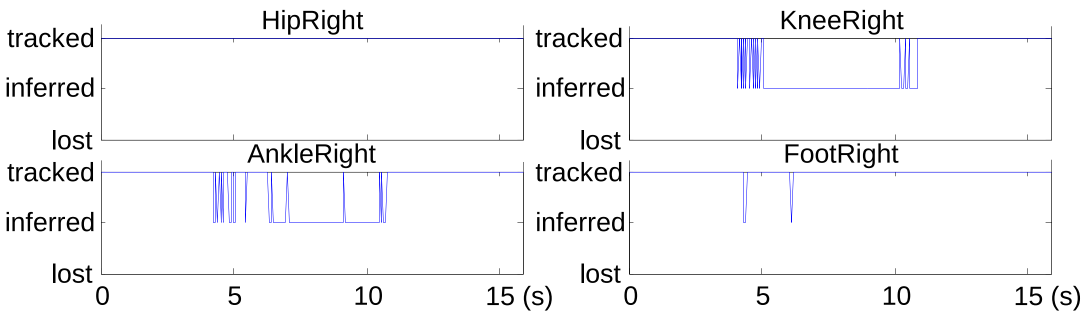


Figure 44: Kinect tracking status of subject 1 experiment 3 for the HipRight, KneeRight, AnkleRight and FootRight joints (x-axis in seconds).

Figure 44 shows that the inferred errors are not distributed homogeneously. Small peaks such as in the Foot Right, could be filtered or ignored via a smart buffering system that always returns the last "tracked" sample. But such a system won't help for the large gaps such as shown in the Knee Right and Ankle Right graphs. Left of the small bars in Table 17 a number is shown, which represents the number of times the tracking status was "inferred" for at least 24 samples. The duration of a single sample is ca 36ms, thus 24 samples equals 870ms, which is more than a single window length (833ms). When 24 samples are missing, this will have a negative impact on the detection algorithms, no matter the filtering or buffering systems used. Table 17 indicates that unrepairable inferred errors (gaps) are present for all subjects, especially for the Ankles, Knees and Hands. Furthermore, the number of gaps does not relate directly to the portion of inferred samples, for example the Knee Left for subject 5 and

6 shows only two gaps, but less than 50% of the samples are tracked. This means the gaps are very long, and will have a large negative impact.

7.6 Summary

The results of the experiments show that the current implementation cannot be used to detect and evaluate exercises. On multiple fields the automated detection and evaluation system is not performing well. The errors are caused unexpected performance speed of the subjects, limited skeleton tracking performance, and errors in the implementation. Despite giving a printed exercise explanation, including an example video, the variance in performance speed was high, and the segment durations were shorter than the desired 5 seconds. The skeleton tracking performance also failed more than expected. Only in 8 out of 46 experiments, the crossing of the legs was detected correctly, and in 10 out of 46 experiments, other relevant body parts were misplaced as well. There was only a single experiment in which the Kinect itself did not report tracking errors, but the crossing of legs was still missed in this experiment, diminishing the value of the “tracked” status reported by the Kinect SDK. Results indicate that the timing and processing speed of the system is adequate, but the system fails to correctly find matches between the measured translations and the exercise playlist. Even when matches are found, in a few cases (21 out of 144) the system continues to look for matches during the same segment. This indicates that the marking of matches is not reliable. On a positive note, the system was able to process all recordings, to read the skeleton data, convert this into translations, compare the translations to the exercise specification, and advance in this exercise specification playlist. Because the duration of the exercise performance differed significantly from the specification and the Kinect SDK skeleton was unreliable, the experiment results do not necessarily reflect the performance of the remaining processing steps.

8 Discussion & Conclusion

8.1 Discussion

As can be read in section 7.5, the results of the experiments show that the implementation all in all does not function well. This section discusses what parts of the design and / or implementation are good, despite the negative results, and where redesign or reimplementation is required.

Figure 45 shows an overview of the most important parts of the detection and evaluation system. The blocks in Figure 45 make it appear as if those blocks are isolated components in the software, and fortunately this is true for the major part. However, due to underestimation of the complexity of the interpretation of the measured translations, parts E and F are intertwined. This underestimation of the complexity is also reflected in the absence of debugging tools, such as tester classes and decent error logging. Such functionality is common in complex software, but often omitted in simple prototypes. A future version of the system definitely should have functionality to isolate each component of the software and test the functionality of the component with known and error-free input data. In the current design, the low quality of the data generated by the Kinect SDK, made it very hard to discern between errors caused by faulty input data and errors caused by bugs in the software. Generating good test data is time consuming, but with hindsight it would have saved a lot of time during debugging of the system. The next paragraphs will discuss issues of each of the components shown in Figure 45, together with possible improvements.

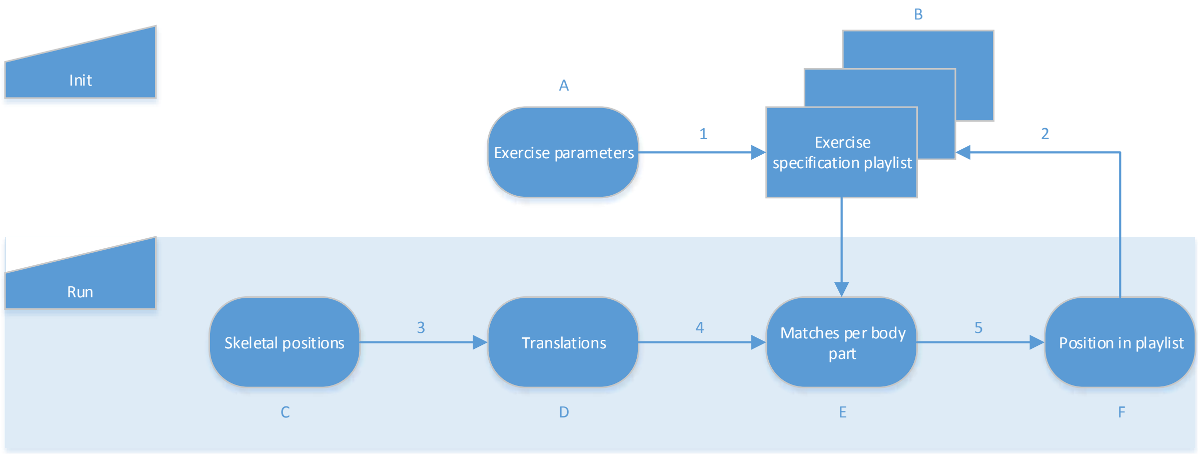


Figure 45: Overview of the most important steps of the detection and evaluation system implementation.

A) Exercise parameters

The exercise parameterization model based on Labanotation provides a good balance between a complete description of each body part's position at every moment in time, and a vague verbal description of the exercise. By omitting irrelevant movements, the model is more robust to detection errors or user variations in body parts which are not relevant. Due to the relatively small set of language components (the different types of translations, body parts, segments etc.) the concept is easy to learn. And despite this small set, the language is not limited to a specific type of exercises or movements. Therefore, it can be stated that the exercise parameterization model based on Labanotation is a satisfactory result to the research initiated by the following research sub-question:

"*Which measurable body movement parameters can be used to evaluate the performance of an exercise that is part of a non-supervised training scheme for rehabilitation patients?*"

Converting an exercise into any structured language does require to be able to recognize and isolate those components in the exercise. For example identifying the efforts that cause a transition from one posture to another can be more complex than just identifying these two postures. A system that can visualize the motion would help significantly in generating correct parameters, because it can provide the physician with direct feedback on the results of the chosen parameters. Two software packages that can visualize Labanotation are: LabanEditor (Kojima, 2002) and LabanDancer (Wang, 2005).

The exercise parameterization model has some support for dynamic personalization, but the current options are too limited to be of use. The "side" of a body part can be personalized ("affected arm"), while the direction does not support personalization. A movement direction "towards affected side" would often be required together with the use of the affected side implementation. A specific requirements analysis is advised to find out what the needs for personalization are for rehabilitation exercises.

B) Exercise specification playlist

The conversion from the exercise parameters into an exercise playlist does not propose problems.

C) Skeletal positions

Without any uncertainty, the Kinect is not the best tool for the automated evaluation of realistic rehabilitation exercises. Analysis showed that out of the 109 realistic exercises, only 11 exercises are suitable without alteration. This analysis was related to the first two research sub-questions: "*What are the pose and movement detection capabilities of the Kinect depth camera when tracking a single person in a non-supervised exercise setting?"* and *"What type of rehabilitation exercises can be evaluated using a Kinect depth camera in a non-supervised setting?*" During the analysis, it became clear that these two questions cannot be discussed separately. Due to the low skeleton tracking performance, only a small subset of the rehabilitation exercises was suitable. Most exercises contained subtle movements, or required contact with a large object (i.e. a bed) that would definitely confuse the detection. Even when the exercise seemed suitable on paper, the skeleton tracking produced too much errors to be of real use. To work around the skeleton tracking issues, almost all rehabilitation exercises will require minor or major adjustments, or specific workarounds have to be found for each exercise. The latter renders the concept of a universal exercise parameterization method pointless, whereas the first is in conflict with the goal of making a universal exercise evaluation tool.

To achieve better tracking performance, a better skeleton tracking technique could be used. At the end of 2013 Microsoft released the successor of the Kinect: the Kinect "One". Microsoft states that this sensor has an improved resolution, and promises that the skeleton tracking will be better. Currently, no release date for the Kinect "One" SDK is given, therefore it remains unknown how much better the Kinect "One" will be. Affordable sensory enhanced clothing recently became available (Athos Works, Inc.) which determines movement of body parts via integrated multichannel EMG. Inevitably, such a system will have both advantages and disadvantages compared to the depth camera based techniques, and thus require new feasibility studies.

Filtering can improve the skeletal input data for the automated detection and evaluation system, without changing the sensor. As discussed in 5.3.2, the skeleton data subsampling system requires low-pass filtering. This filtering step is not implemented in the prototype due to time constraints. The framework in which the filter can be implemented, the `ResultWeighter`, is present, but currently only used to add weight to constant trends. A filter that takes the position of neighboring body parts into account might help to make the position estimations more realistic. In the Kinect Skeleton model the length of bones can change frequently, and there are no limitations on realistic degrees of freedom of a joint. Via patient specific parameters, bone lengths and degrees of freedom of joints can defined, which will help to identify and correct detection errors. Another option that might improve the skeleton tracking performance, is an exchange of Kinect software. In the starting phase of this project a choice between Microsoft's Kinect SDK and PrimeSense's OpenNI/NITE was made. Based on short experiments, and evaluation of the specifications, the skeleton tracking performance of both systems seemed comparable. Because the skeleton tracking performance turned out to be one of the major issues of the system, a more in-depth evaluation on the performance differences between the two skeleton tracking packages is justified. Both filtering and exchanging the Kinect software probably does not improve the tracking performance enough to make the Kinect depth camera suitable for detection of unaltered exercises. All Kinect enabled exercise evaluation applications found in literature, are implemented for a limited set of exercises. These exercises mainly consist of coarse movements, without interaction with foreign objects. If the Kinect would be suitable for a broad range of exercises by using different software and advanced filtering, this would be shown in literature.

Sub-sampling of the skeleton positions for detection and evaluation is done at a relatively low frequency of 0.8 seconds. The average duration of the first segment of all experiments was 2.35s. If this segment starts halfway a sample window, this means that 2 times 17% of the duration of the segment is mixed up with another sample (at beginning and end). It is not acceptable to obscure 1/3th of the segment, therefore a much shorter sample interval should be used. Unfortunately, the experiment results show that processing of the samples can take up to 175ms in the worst case. This indicates that ca 6Hz is the upper max with the present efficiency and processing power (Intel Core i5 2520M). A higher sampling frequency enables a much more flexible windowing system, without the need for overlapping windows. A higher sample rate will also help to shorten the startup delay of the system. Currently, the first output is generated after two window lengths, but not all subjects remain static for so long.

D) Translations

The exercise parameterization model based on Labanotation is not only used to define the exercise, the same set of concepts and language components is also used for the detection and evaluation. Using the same language for both specification and detection makes comparison between the specification and the measurements straightforward. The developed system is able to convert the measured joint positions into translations that are compliant with the Labanotation based model. Every translation belongs to a specific body part, and has a specific duration. The experiment results show that the duration concept is flawed (see the next paragraph), but the concept of multiple translation types per body has much potential. Therefore it forms a successful result for the research initiated by the fourth research sub-question: *"How can the measured body movement parameters be automatically detected from the motion data recorded with a Kinect depth camera?"*

Of the four translation types, currently only the horizontal and vertical translations are fully implemented. To achieve a universal detection and evaluation system, implementation of rotational translations has to be finished. However, for the relative translation, finishing the implementation is not straightforward. The current implementation has a strict division in body parts, but for calculation of the relative translation, information of multiple body parts is needed. Furthermore, the concept of the relative translation is much less straightforward than that of the other translations, and in most cases it gives redundant data in respect to the horizontal and vertical translations. Therefore, a new requirements analysis of the usability of the relative translations is advised before deciding to implement the relative translations.

In section 7.5.1 is shown that there was a large variation in the exercise duration between subjects, but that all performances were shorter than expected. For example, segment 1 lasted only 2.5 seconds, half of the time it was parameterized to last. As a consequence, the movements the subjects performed during the second segment are accounted to belong to segment 1, and are thus mixed up. This happens because the system assumed that the first segment lasted 5 seconds. Obviously, a more flexible segment duration system is needed. If the requirement for a real-time system is dropped, normalization in time can be used. To normalize the time, the start and end of the exercise have to be detected (from static to static posture), and this interval has to be scaled to last one “unit”. The exercise parameters will have to be given in portions of this “unit”. To make the system better suited for real-time applications, correlation can be used to detect the segment durations. Via cross correlation between the exercise specification and the measured translations, the time shift that gives the best match can be found. When the correct time shift is found, the measured translations can be scaled in time (made shorter or longer), to search for the highest correlation with the exercise specification. Such a system would need some settling time, for example by first performing a “trial” exercise. The advantage is that it still enables to give feedback on performance speed, information which would be lost when the duration is normalized to one “unit”.

Translations are calculated based on the change in position between two consecutive samples (see section 5.3.3). For each body part, these translations are calculated in exactly the same way, which can give problems at the extremities. See for an example Figure 46, while this person was bending her upper body, the right hand moved down, and the left arm moved up and towards the left (see also the overlay skeleton on the right of Figure 46). Despite a significant movement of the hands, the user probably did not intent to move her hands. Unintentional movements are obviously only a problem if they occur for relevant body parts. Unfortunately, this was the case for the target exercise, in which the subjects were asked to hold their arms along their body. There are two “solutions” to this problem. The first would be to predict such involuntary movements and explicitly incorporate them in the exercise parameters. The second solution is to calculate the translations based on relative changes of a joint in respect to the parent joint. In the example, the position and orientation of the shoulder joints can first be subtracted from the hand joints, before calculating the movement of the hands.

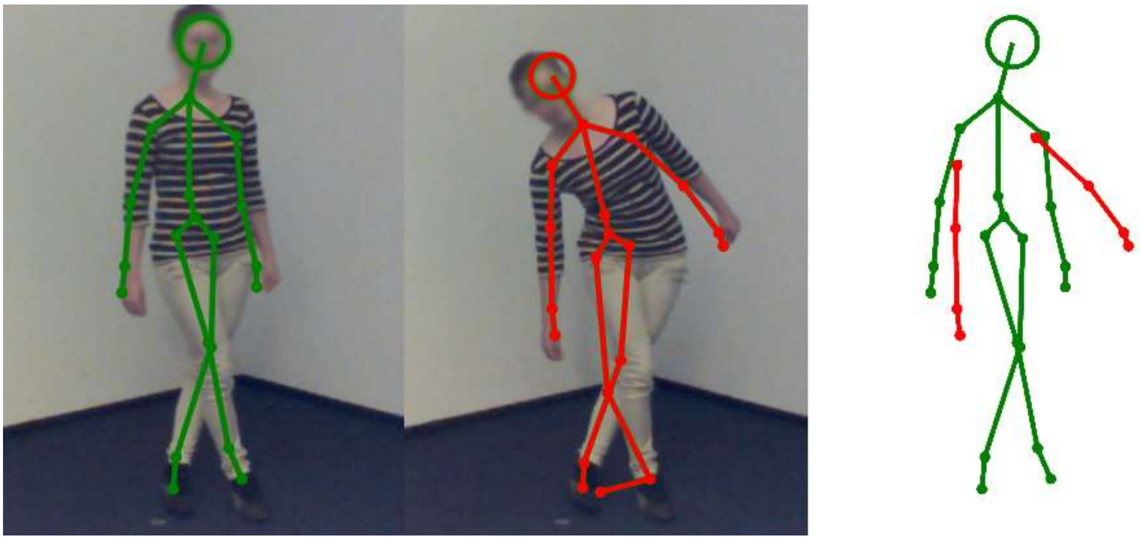


Figure 46: Screen capture of a skeleton overlay of a person standing upright (green), and bend (red). On the right, the full upright skeleton is drawn, and the arms of the bent skeleton.

E) Matches per body part

As stated at the beginning of section 8.1, components "E" (Matches per body part) and "F" (Position in playlist) were designed as a single function. This is also reflected in the last research sub-question: *How can the detected movements be compared to the intended exercise, in order to be able to evaluate performance?*, in which comparison of single movements and evaluation of the whole exercise are proposed in a single question. It soon became clear that comparing a single measurement sample, is not comparable to the evaluation of the exercise in its total. The Labanotation based language provided a very good basis to perform the comparison. Because the same modalities were used for the specification and the storage of the measurement results, it was possible to calculate the "mathematical" difference (the match quality) between a single specification element and a measurement sample. The challenge lies in knowing to what specification element the comparison has to be made. In the start this is simply the first element of the specification playlist, but when to advance to later elements? In the design, the system would advance in the playlist if the average match quality of all relevant body parts was above a predefined threshold. This did not work correctly, mainly because the major part of the exercise specification contained passive translations (do nothing). These passive translations were matched when the subject was standing still, and the system advanced in the playlist before the subject could even start the performance. A division between active and passive translations was made to overcome this problem. Due to limited quality of the skeleton model there was no decent test data to evaluate this concept. Therefore, the system in its current form can only be presented as a system which is able to compare a single measurement sample to a single specification element (component E). The renewed knowledge of the required functionality of the part that controls the position in the playlist (component F) makes a much more reliable design possible, than the current proposed and implemented design.

A full redesign of component E is not required, but there are parts which can be improved. The match between a measured translation and the exercise specification is influenced by the type of specification element. For example, passive elements require a more strict match than active elements. The current implementation to influence the match is a bit of a "hack", because it wasn't part of the design initially, and added later. It would be better to implement a system to add a "quality" indication to each measured translation and for each scoring aspect. This makes it much easier to, for example, give more “weight” to an active translation or to decrease the “weight” of translations that relied on samples with an “inferred” tracking status.

Matches are also influenced by multiple predefined parameters, for example the relation between body part displacement and duration of an action, the minimal match quality for a match to be considered “good”, etc. (see appendix 10.1). Because the system never produced reliable output, it was hard to optimize all these parameters. The current parameters are based on theoretical analysis, or analysis of incomplete results. Therefore, it can be assumed that optimizing these parameters will definitely improve the performance of the system. The most important parameter, “normal speed”, which relates body part displacement to the duration of an action, could even be optimized on the fly, via a correlation system comparable to the segment duration optimization discussed earlier.

F) Position in playlist

The `ResultInterpreter` calculates a match between measured translations of a single body part and the elements of the exercise specification playlist at the position which it receives from the `PercentageChecker`. If this match fails, it will try to match the next element from the exercise specification playlist. If the match was successful, it will mark the measured translations as matched. The `PercentageChecker` does not get feedback on the end result of a match, it just gives a single exercise playlist position based on the results of the previous run. This makes the control of the position in the exercise playlist (the progress of the exercise performance) difficult to follow. Analysis of the log file shows that there are serious bugs in this position control, see for example Table 18, here the Right Ankle triggered a “Failed to match” and “Match” at the same time. This is clearly incorrect, but the exact source of the error is hard to track, due to the lack of a central broker to control the position in the playlist.

Time	Class	Action	Body part	Sequence #	Match quality
25.490	resultInterpreter	Match	AnkleRight	9	46.6
25.491	resultInterpreter	Failed to match	AnkleRight	4	

Table 18: Extract from log file, showing contradictive results.

A “Failed to match” will trigger an indication of the “Largest translation”, to evaluate what the person did instead of the intended movement. If only a single element in the exercise specification playlist could not be matched, it is clear which body part performed a false movement, and how long this movement lasted. But the “Largest translations” are also triggered when the position in the playlist is not known at all. In this case, the “largest translations” can only indicate what the user did, and not what he should have done. Therefore, in the current implementation it is better to mute the “Largest translation” feedback if multiple elements are missing.

8.2 Conclusion

There is a demand for automated rehabilitation exercise detection and evaluation systems which can be used in a non-supervised out-of-clinic setting. In this report, research is done to find out whether the affordable Microsoft Kinect depth camera can be used for such an exercise evaluation system. The Kinect depth camera is developed to control specially designed games via body movements. Due to the limited resolution of the depth camera, the Kinect cannot track subtle movements such as movement of the chest due to respiration, and is easily confused if limbs are crossed. Such limitations hamper the use of the Kinect for automated evaluation of rehabilitation exercises. Analysis of 109 realistic rehabilitation exercises showed that 98 exercises would certainly not be suitable for evaluation without altering the exercise.

The detection and evaluation system is based on a set of parameters of an exercise. Parameterization is chosen over reference recordings, because the latter lacks context information. A reference recording based detection system can indicate that a performance was false, but is not able to tell what exact movement caused the error, and what specific changes are needed to correct the error. Unfortunately, there is no universally accepted method to parameterize an exercise. Therefore, the concepts of a method used to notate dances, Labanotation, are used to create a new parameterization method. This method can be used to specify all movements which are relevant for an exercise. Based on this method, a detection and evaluation system is designed and implemented. This system reads the parameterized exercise to create an exercise playlist containing all translations of each relevant body part in successive order. The system converts the joint positions, measured by the Kinect SDK, into translations of the same type as used in the parameterization. These translations are compared to the exercise playlist, and when a sufficient part is matched, the position in the playlist is advanced.

In order to test the implementation, 8 subjects each performed 6 different versions of the same target exercise. Unfortunately, the results of these experiments were not positive. The main issue lies outside the scope of the implementation: the subpar skeleton tracking performance. Only in 8 out of 46 experiments, the crossing of the legs was detected correctly, and in 10 out of 46 experiments, multiple relevant body parts were misplaced. Solely based on the skeleton tracking performance of the Kinect SDK, it can be stated that the Kinect depth camera is not a suitable tool for evaluation of common rehabilitation exercises. The exercise parameterization method makes it unfeasible to create exercise specific workarounds to cope with the Kinect skeleton tracking limitations. Requiring to adapt the exercises or exercise specifications to the limitations of the Kinect, also defeats the goal of creating a universal exercise detection and evaluation framework.

The high number of errors in the skeletal positions made it hard to determine the performance of the detection and evaluation system. But even in the experiment performances containing few skeleton tracking errors, the number of correct matches made by the system remains too low to advance in the exercise playlist. The high variance in performance duration definitely played a role in the low detection rate, showing that the system requires additional functionality to normalize the exercise duration. Experiment results indicate that the timing and processing speed of the system are adequate, but the system often fails to correctly find matches between the measured translations and the exercise playlist. Even when matches are found, in a few cases (21 out of 144) the system continues to look for matches during the same segment. This indicates that the marking of matches is not reliable.

The main research question was: “How can the Microsoft Kinect camera be used for automated rehabilitation exercise evaluation in a non-supervised setting?” The proposed design and implementation showed that the Microsoft Kinect camera cannot be used for automated rehabilitation exercise evaluation without alteration of the exercises, or exercise specific workarounds. To see if the framework would generate sensible results with a better sensor, an addition is needed which enables testing of the system with known and error free input data. During development, a full redesign of the `ResultInterpreter` was already considered, because identifying the cause of the errors became harder and harder when the complexity grew. Unfortunately, time limitations made this redesign impossible. However, if time is invested in this framework, the parts of the `ResultInterpreter` which control the position in the playlist should definitely be redesigned. The parameterization language based upon Labanotation showed a lot of potential, and should be maintained in the redesign. It caused no problems at all as method to specify the exercise. The interpretation of the duration of a movement turned out to be a hard-to-solve problem. Normalization in time can be used, but is difficult to implement in a real-time system. It is also possible to ignore duration and speed altogether, but this limits the modalities of feedback on the patient performance. Perhaps the best, but also the most complex way, would be to “learn” the performance speed on-the-fly.

8.3 Future vision

When the detection and evaluation system developed in this thesis would be fully functional, it can be a valuable addition to a system such as the CoCo web portal. As stated earlier, the current home rehabilitation systems lack sufficient monitoring tools. They are not able to adapt the exercises to the current progress of the patient, nor are they able to check whether the patient actually performed the prescribed exercises. This section describes two scenarios that give some insight in how a home rehabilitation system equipped with a fully functional detection and evaluation system would offer a more real-time interaction with the patient ("The interactive coach") and how such a system would offer a professional caregiver valuable information on the patient's compliance ("The compliance monitor").

8.3.1 Scenario: The interactive coach

Laura is a 35 year old researcher. Last summer she fell while mountain biking, and seriously injured her legs. Since then she regularly visits the rehabilitation center to regain full function of her legs. But now that she has recovered enough, she is not allowed to go to the rehabilitation center anymore. Instead she has to continue her exercises at home. To increase the efficacy of home training, her rehabilitation doctor prescribes Laura the *Virtual Exercise Coach*. This is a web portal accompanied by a depth camera. The depth camera is installed in Laura's living room on top of her television. To use the *Virtual Exercise Coach* Laura simply browses to the web portal using her Windows PC, and holds her insurance card in front of the depth camera. The *Virtual Exercise Coach* then loads Laura's profile, consisting of a list of suitable exercises for the day. Each exercise has a specific category and difficulty, and the weekly goal consists of a target per category. Laura gets in front of the television, and the *Virtual Exercise Coach* indicates that Laura can choose an exercise and start the workout. By making a fist, and then moving her hands up and down, Laura chooses the exercise "Knee flexion and extension" from the category "leg strength exercises". An animated figure resembling a woman of Laura's age appears on the television screen performing the chosen exercise. Because Laura knows this exercise by heart, she says "skip" out loud to skip the example performance. While performing the exercise, Laura noticed that the pace has gone up compared to last week. The *Virtual Exercise Coach* increased the pace to compensate for Laura's recovery and thus keeping the exercise equally challenging. After 10 minutes Laura starts to get bored due to the short and repetitive exercise. Via visual mood recognition, the *Virtual Exercise Coach* senses this mood change, and suggests to change the exercise to a more exciting one. Laura accepts to start the new exercise, and the *Virtual Exercise Coach* will monitor if her mood indeed improves, to optimize the suggestion system. Laura is not very familiar with the new exercise and makes some errors. After 2 repetitions, the animated figure appears on the television to give an example performance with Laura's errors exaggerated, so she can see what she did wrong. Because Laura is lacking the strength to improve on these errors in this complex exercise, the system presents an alternative exercise that targets the movements she did wrong, but which is easier to perform. Laura accepts this suggestion, and starts performing the exercise. Because the exercise is less demanding, Laura's heart rate drops. The *Virtual Exercise Coach* senses that her heart rate is decreasing via the optical heart rate sensor, and compensates by increasing the pace of the exercise. After another 15 minutes, Laura stops the *Virtual Exercise Coach*, and goes to work on her bicycle. The mobile client of the *Virtual Exercise Coach* installed on her smartphone monitors her trip on the bicycle. When she arrives at her destination, she gets a popup telling her that her physical condition is improving well, and that she was able to travel the distance 2 minutes faster than last week. At the end of the day Laure made a huge detour on her way back to home. Because of this activity, the upper leg strength exercises for the next day are automatically omitted from the program.

8.3.2 Scenario: The compliance monitor

Peter is a 50 year old accountant, who sits at his desk for 8 hours a day. Peter's physical condition is going downhill for quite some years, and is becoming critical recently. On advice of his general practitioner, Peter joined a personal health improvement program of the insurance company. In this program individuals are stimulated to live healthier by following guidelines regarding diet, daily activity and smoking. The adherence to the program is rewarded with a discount on their insurance costs. At the next biannual checkup, Peter assured the GP that he did all the exercises of the program. But the GP had reasons to doubt Peter's honesty, because Peter's weight increased significantly in the last 6 months. The GP wanted Peter to perform his exercises, but knew Peter wouldn't stick to the training scheme on his own. Because Peter's condition is not severe enough to send him to a rehabilitation center, the GP puts Peter on the new *Virtual Exercise Coach* program.

The *Virtual Exercise Coach* is a web portal accompanied by a small PC and a depth camera. The PC was installed in Peter's living room, and connected to his television. The depth camera was placed on top of this television. To use the *Virtual Exercise Coach* Peter simply turns on the small PC, and holds his insurance card in front of the depth camera. The *Virtual Exercise Coach* then loads Peter's profile, consisting of a list of suitable exercises for the day. Each exercise has a specific category and difficulty, and the weekly goal consists of a target per category. Peter gets in front of the television, and the *Virtual Exercise Coach* indicates Peter can choose an exercise and start the workout. By making a fist, and then moving his hands up and down, Peter chooses the exercise "bending the torso" from the category, "sitting movement exercises". Because this exercise has a difficulty of only "3", Peter should perform the exercise 10 times each day to reach the target. But after only 5 times he got bored and stopped with the exercises for the day. In the next weeks Peter exercised less and less, despite messages on his phone reminding him of the exercise goals.

Peter's GP also received a message from the *Virtual Exercise Coach* about Peter's subpar performance, and asked Peter about his performance at the next biannual checkup. As a consequence of his bad performance, Peter is expelled from the personal health improvement program, and loses the discount on his insurance costs. He is allowed to keep the *Virtual Exercise Coach* and will regain the discount when he achieves the exercise target for the next month.

8.3.3 Analysis

The first scenario depicts a system which is interactive and multimodal. Both of which are important aspects to motivate people in performing relatively dull exercises. People are more motivated to do repetitive tasks if their virtual trainer is "reactive" (Bickmore & Cassell, 1989). This reaction (or feedback) should not be limited to technical aspects, psychological and sociological interaction are key aspects in keeping a patient motivated. Unfortunately, interaction on a social level has always turned out to be very difficult to implement into an ICT system. The social aspect of the reaction in the scenario could be facilitated by automated object detection systems, like Fraunhofer's SHORE (Ruf et al., 2011), which can give an indication of the mood of the patient. The technical input is given by measuring key performance indicators such as exercise duration, movement speed, heart rate and movement distance (for example: how far the patient bends his / her upper body). The social aspect can be facilitated by logging activity during the day, and subsequently adapting the exercises based on the activity patterns of the patient. A “vision” of social interaction, not given in this scenario, is a system enhanced with a “virtual community”. The virtual community is a group of patients that all use the same home rehabilitation web portal, for roughly the same goals, and form a community by communicating with each other via the web portal. One of the reasons to build such a community is peer pressure. It is shown that people tend to perform better if the quality of their performance is related to a position in a competition (Torsi & Wright, 2010). Such a ranking can be generated based on the measured exercise performance quality, but normalization is required. Unfortunately, normalization is not trivial because each patient shows his / her own recovery speed, and only a part of this recovery speed is related to the willingness and adherence of the patient. Factors such as the type of trauma play a large role in the recovery speed, but cannot be influenced by the patient, and therefore such factors should not play a role in the competition.

9 Bibliography

Asus. (2012). Xtion PRO LIVE. Retrieved February 18, 2013, from http://www.asus.com/Multimedia/Xtion_PRO_LIVE/#specifications

Bickmore, T., & Cassell, J. (1989). Social Dialogue with Embodied Conversational Agents. In *Natural, Intelligent and Effective Interaction with Multimodal Dialogue Systems.*

Brucker, F. (2012). *Analysis of methods for segmentation and representation of time series for the recognition of motion pattern*. Institut fur Parallele und Verteilte Systeme. Retrieved from http://elib.uni-stuttgart.de/opus/volltexte/2012/7497/

Camporesi, C., Kallmann, M., & Han, J. J. (2010). VR Solutions for Improving Physical Therapy, (128), 915665.

Catuhe, D. (2013). Kinect Toolbox. Retrieved October 07, 2013, from http://kinecttoolbox.codeplex.com/

Chang, C.-Y., Lange, B., Zhang, M., Koenig, S., Requejo, P., Somboon, N., … Rizzo, A. (2012). Towards Pervasive Physical Rehabilitation Using Microsoft Kinect. *Proceedings of the 6th International Conference on Pervasive Computing Technologies for Healthcare*, 2–5. doi:10.4108/icst.pervasivehealth.2012.248714

Dutta, T. (2012). Evaluation of the Kinect™ sensor for 3-D kinematic measurement in the workplace. *Applied ergonomics*, *43*(4), 645–9. doi:10.1016/j.apergo.2011.09.011

Elteren, T. van, & Zant, T. van der. (2012). Real-Time Human Pose and Gesture Recognition for Autonomous Robots Using a Single Structured Light 3D-Scanner. *Workshop Proceedings of the 8th …*, 213–220. doi:10.3233/978-1-61499-080-2-213

Foroud, A., & Whishaw, I. Q. (2006). Changes in the kinematic structure and non-kinematic features of movements during skilled reaching after stroke: a Laban Movement Analysis in two case studies. *Journal of neuroscience methods*, *158*(1), 137–49. doi:10.1016/j.jneumeth.2006.05.007

Fujimura, W., Kosaka, T., & Robert, S. (2012). AccuMotion : for the post-PC era. In *VRIC '12 Proceedings of the 2012 Virtual Reality International Conference*. doi:10.1145/2331714.2331732

Griesbeck, C. (1996). Introduction to Labanotation. Retrieved from http://user.uni-frankfurt.de/~griesbec/LABANE.HTML

Guest, A. H. (1977). *Labanotation: Or, Kinetography Laban : the System of Analyzing and Recording Movement*. Theatre Arts Books. Retrieved from http://books.google.nl/books?id=Tq1YRDuJnvYC

IFixit. (2011). Microsoft Kinect Teardown. Retrieved February 18, 2013, from http://www.ifixit.com/Teardown/Microsoft+Kinect+Teardown/4066/1 iPiSoft. (2013). ASUS Xtion vs MS Kinect Comparison. *Wiki*. Retrieved February 18, 2013, from http://wiki.ipisoft.com/ASUS_Xtion_vs_MS_Kinect_Comparison

Jack, D., Boian, R., & Merians, A. (2000). A virtual reality-based exercise program for stroke rehabilitation. *Proceedings of the …*, 56–63. Retrieved from http://dl.acm.org/citation.cfm?id=354340

Jolly, K., Taylor, R., Lip, G. Y. H., Greenfield, S., Raftery, J., Mant, J., … Lee, K. W. (2007). The Birmingham Rehabilitation Uptake Maximisation Study (BRUM). Home-based compared with hospital-based cardiac rehabilitation in a multi-ethnic population: cost-effectiveness and patient adherence. *Health Technol Assess.*, *11*(35).

Kahol, K., & Tripathi, P. (2006). Motion Sequences with a Personalized Annotation System. *MultiMedia, IEEE (Volume:13 , Issue: 1)*, 37–45.

Khoshelham, K., & Elberink, S. O. (2012). Accuracy and resolution of Kinect depth data for indoor mapping applications. *Sensors (Basel, Switzerland)*, *12*(2), 1437–54. doi:10.3390/s120201437

Kojima, K. (2002). Labaneditor: Graphical editor for dance notation. *Robot and Human …*, 59–64. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1045598

Livingston, M. a., Sebastian, J., Ai, Z., & Decker, J. W. (2012). Performance measurements for the Microsoft Kinect skeleton. *2012 IEEE Virtual Reality (VR)*, 119–120. doi:10.1109/VR.2012.6180911

Lu, Y., & Jiang, H. (2013). Human movement summarization and depiction from videos. In *Multimedia and Expo (ICME), 2013 IEEE International Conference on*.

Microsoft. (2012). Kinect for Windows Sensor Components and Specifications. *Micosoft Developer Network*. Retrieved February 18, 2013, from http://msdn.microsoft.com/en-us/library/jj131033.aspx

Microsoft. (2013a). Kinect for Windows SDK - MSDN Docs. Retrieved from http://msdn.microsoft.com/en-us/library/hh855347.aspx

Microsoft. (2013b). Kinect for Windows SDK: Tracking Modes (Seated and Default). Retrieved from http://msdn.microsoft.com/en-us/library/hh973077.aspx

OECD Health Data 2010. (2010). *Eco-Santé 2011*.

Park, J.-H., Shin, Y.-D., Bae, J.-H., & Baeg, M.-H. (2012). Spatial uncertainty model for visual features using a Kinect™ sensor. *Sensors (Basel, Switzerland)*, *12*(7), 8640–62. doi:10.3390/s120708640

Pastor, I., Hayes, H. a, & Bamberg, S. J. M. (2012). A feasibility study of an upper limb rehabilitation system using kinect and computer games. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference* (Vol. 2012, pp. 1286–9). doi:10.1109/EMBC.2012.6346173

PrimeSense. (2011). The PrimeSense ™ 3D Awareness Sensor.

PrimeSense. (2012). Developers > Get your Sensor. Retrieved February 18, 2013, from http://www.primesense.com/developers/get-your-sensor/

Ruf, T., Ernst, A., & Küblbeck, C. (2011). Face detection with the sophisticated high-speed object recognition engine (SHORE). *Microelectronic Systems*, 237–246. doi:10.1007/978-3-642-23070-7

Tabak, M., Brusse-Keizer, M., Ommeren, C., Kotte, H., Weltevreden, P., Hermens, H., & Vollenbroek-Hutten, M. (2013). A telecare programme for self-management of COPD exacerbations and promotion of an active lifestyle. In *European Respiratory Society Annual Congress*. Retrieved from http://www.rrd.nl/projects/CoCo.html

TechSmith. (2013). Camtasia Studio. Retrieved from http://www.techsmith.com/download/camtasia/default.asp

Torsi, S., & Wright, P. (2010). The self-management of chronic illnesses: Theories and technologies. *… Technologies for …*. doi:10.4108/ICST.PERVASIVEHEALTH2010.8914

Ukita, N., Kaulen, D., & Röcker, C. (2014). Towards an Automatic Motion Coaching System. In *International Conference on Physiological Computing Systems* (pp. 167–172).

Wang, W. (2005). The IS Framework for Preserving Dance Heritage. *Goizueta Business School Paper Series*.

Welbergen, H. Van, & Ruttkay, Z. (2008). Elbows Higher ! Performing , Observing and Correcting Exercises by a, 409–416.

10 Appendixes

10.1 Software Parameters

Parameter Name	Value	Explanation
ZeroThreshold	0.005	All translation amounts below threshold are set to zero in the "Weighted" translation list.
RepetitionBonus	1.3	When a translation is happening in the same direction, and with the same amount, for multiple windows, it is increased with this bonus factor, to make it more important.
RepetitionThresholdPercent	85	Threshold in percentage, sequential samples may differ this much to be considered within trend.
WindowDivider	4	The shortest window is "WindowDivider" times shorter than the shortest element from the exercise specification.
NormalSpeed	4	This is the essential value, relates a "normal" movement of a certain duration into an absolute displacement. Used for Horizontal and Vertical translations.
NormalSpeedRotation	4	This is the essential value, relates a "normal" movement of a certain duration into an absolute displacement. Used for Rotational translations.
FutureTimePenalty	0.95	The penalty for shifting forward in de spec.
MatchThresholdPercent	75	If deviation in percentages between the specification and measurement is above this number, that body part is matched.
MatchThresholdAllBodyParts	75	The percentage of the relevant body parts that have to be matched, in order to skip to the next element in the spec.
ZeroThresholdMatching	0.05	The minimal value before match can be made.
MaxDistanceToZero	0.2	If specified value is 0, this is the max distance to this value, all above gives 0% match, all below "ZeroThreshold" gives 100% match.
VerticalMiddleThreshold	0.15	If measured vertical translation is between + and - VerticalMiddleThreshold it is seen as a "middle" direction.
HorizontalPlaceThreshold	0.10	If measured horizontal translation is between + and - HorizontalPlaceThreshold it is seen as a "place" direction.
MaxAngleDeviationHorizontal	180	The maximum difference between measured and specified angle, before the match is considered 0%.
MaxAngleDeviationVertical	360	The maximum difference between measured and specified angle, before the match is considered 0%.
MaxFutureIterations	9	If measurement cannot be matched with current specification, an attempt is made to match with the next step in the specification. This is the limit on how often this is done.
VerticalAngleFactor	25	The verticalAngleFactor is a number between one and 100, if it is 25, this means the angle accounts for 1/4th of the match and the amount for 3/4th.
ZeroThreshold	0.005	All translation amounts below threshold are set to zero in the "Weighted" translation list.

10.3 CoCo Exercise evaluation

Number	Name	Incomplete model	Fine movements	Contact objects	Occlusion problems	Viewpoint problems	Orientation information	Detection possible	Evaluation possible	Comments
COPD	Ademhalingstechnieken									
C01	PLB (Pursed lips breathing)	▼	▼	▲	▲	▲	▲	✖	✖	
C02	Ademhaling voelen - Ruglig	▼	▼	▼	▼	▼	–	✖	✖	
C03	Ademhaling voelen - Zit	▼	▼	–	▼	▲	▲	✖	✖	
C04	Ademhaling voelen - Stand	▼	▼	–	▼	▲	▲	○	✖	Standing can be detected
C05	Borst/flank/buikademhaling	▼	▼	▲	▼	▲	▲	○	✖	Standing can be detected
C06	Middenrif functie optimaliseren	▼	▼	▼	▲	▲	▲	✖	✖	
C07	Krachttraining hulpademhalingsspieren m.b.v. Threshold	▼	▼	–	▲	▲	▲	✖	✖	
COPD	Sputummobilisatie									
C08	Huffen	▼	▼	–	▲	▲	▲	✖	✖	
C09	Hoesten	▼	▼	–	▲	▲	▲	✖	✖	Coughing noise can be detected??
C10	ACBT (Actieve Cyclus van ademhalingstechnieken)	▼	▼	–	▲	▲	▲	✖	✖	"Hands in sides" can be detected
COPD	Thoraxmobilisatie									
C13	Strekken van romp	–	–	▲	▲	▲	▲	✔	✔	Respiration cannot be detected
C14	Draaien van de romp	▲	▲	▲	▲	▼	▲	✔	○	Respiration cannot be detected
C15	Zijwaarts bewegen van romp	▲	▼	▲	–	–	▲	✔	○	Pulling at neck cannot be detected
COPD	Ontspanning									
C16	Ontspanning schouderspieren	▲	–	–	▲	▲	▲	✔	○	
C17	Algehele ontspanning	▼	▼	▼	▼	▼	▲	✖	✖	
COPD	Functionele oefeningen									
C11	Krachtoefening rechte buikspieren	▲	▲	▲	–	–	–	✔	✔	
C12	Krachtoefening - schuine buikspieren	▲	▲	▲	▲	–	–	✔	✔	
C18	Krachtoefening buigspier arm	▲	▲	–	–	▲	▼	✔	○	
C19	Krachtoefening strekspier arm	▲	▲	▼	▲	▲	▲	○	○	Interaction with chair rests causes problems
C20	Uitduwoefening vanaf muur	▲	▲	▼	–	–	▼	○	✖	Interaction with wall causes problems
C21	Krachtoefening bovenbeenspier - gaan zitten	▲	▼	–	▲	▲	–	✔	○	Detection of hovering above chair is difficult
COPD	Conditie									
C22	Looptraining	▲	▲	▲	▲	▼	▲	○	○	Patient will walk out of range
C23	Steppen op een trap	▲	▲	▼	–	▼	▲	○	✖	Strairs are probably not in range of camera
C24	Fietsen op de hometrainer	▲	–	▼	–	▼	–	✖	✖	Interaction with home training will cause problems
Heup Ac.	Transfers									
HA01	Lig-zit	▲	▲	▼	▼	▼	▲	✖	✖	
HA02	Lig - zit (met gebruik hoofdsteun)	▲	▲	▼	▼	▼	▲	✖	✖	
HA03	Bed - stoel (met looprek)	▲	▲	▼	▼	▼	▲	✖	✖	
HA04	Bed - stoel (met rollator)	▲	▲	▼	▼	▼	▲	✖	✖	
HA05	Bed - stoel (met elleboogkrukken)	▲	▲	▼	▼	▼	▲	✖	✖	
HA06	Bed - stoel (met begeleiding één persoon)	▲	▲	▼	▼	▼	▲	✖	✖	
HA07	Gaan staan/zitten (met looprek)	▲	▲	▼	–	▲	▲	○	✖	Walker blocks frame
HA08	Gaan staan/zitten (met rollator)	▲	▲	▼	–	▲	▲	○	✖	Rollator blocks frame
HA09	Gaan staan/zitten (met elleboogkrukken)	▲	▲	▼	–	▲	▲	○	✖	Sticks will confuse detection
HA10	In/uit auto (met plastic zak)	▲	▲	▼	▼	▼	–	✖	✖	
HAB03	Zit - lig	▲	▲	▼	▼	▼	–	✖	✖	
HAB12	Op- en afstappen van de fiets	▲	▲	▼	▼	▼	–	✖	✖	
Heup Ac.	Oefeningen in lig									
HA12	Aanspannen bovenbeenspieren	▲	▼	▼	▼	▼	–	✖	✖	
HA13	Flexie poetsen	▲	▲	▼	–	▼	▲	✖	✖	
HA14	Been gestrekt heffen	▲	–	▼	▼	▼	▲	✖	✖	
HA15	Hakken/tenen optrekken	–	▼	▼	–	▼	▲	✖	✖	
HA16	Beenheffing in zijlig	▲	▲	▼	▼	▼	▲	✖	✖	
HAB13	Buiklig	▲	–	▼	▼	▼	–	✖	✖	
HAB14	Voeten ronddraaien	▼	▼	▼	▼	▼	▼	✖	✖	
HAB17	Eindstrekking knie	–	–	▼	▼	▼	–	✖	✖	
	HC06 - HC63: contact object problems							✖	✖	
Heup Ac.	Oefeningen in zit									
HA17	Hakken/tenen optrekken	▼	▼	–	▲	▲	–	✖	✖	
HA18	Knieën strekken	–	–	–	▲	▲	▲	✔	✖	Movement of foot cannot be evaluated
HA19	Knieën heffen in zit	▲	–	–	▲	▲	▲	✔	○	Chair can disturb detection
HA20	Spreid/sluit benen	▲	–	–	▲	▲	▲	✔	○	Chair can disturb detection
HA21	Voor/achter in de stoel	▲	–	–	▲	▲	▲	○	✖	Chair can disturb detection
HA22	Vorlage	▲	–	–	▲	▲	▲	○	✖	Chair can disturb detection
HAB22	Strekking knie	▲	▲	–	▲	▲	▲	○	✖	Stool can disturb detection
HAB26	Bovenbeen op zitting duwen	▲	–	–	–	▲	▲	○	✖	Arms probably too close to body

Heup Ac.	Oefeningen in stand									
HA23	Gewicht verplaatsen	▲	▬	▬	▲	▲	▲	!	!	Chair can disturb detection
HA24	Lopen op de plaats	▲	▲	▬	▬	▲	▲	!	!	Chair can disturb detection
HA25	3D aantippen	▬	▬	▬	▬	▲	▲	!	!	Chair can disturb detection
HA26	Tenenstand	▬	▬	▬	▲	▲	▲	!	!	Chair can disturb detection
HA27	Been zijwaarts heffen in stand	▲	▲	▬	▲	▲	▲	✔	!	Chair can disturb detection
HA28	Balans in stand	▲	▲	▬	▬	▲	▲	✔	!	Chair can disturb detection
HAB34	Stand op één been, knie buigen	▲	▬	▬	▬	▲	▲	!	✖	Chair can disturb detection
HAB37	Been achterwaarts heffen in stand	▲	▲	▬	▬	▲	▲	!	!	Chair can disturb detection
HAB38	Op- en afstappen	▲	▲	▬	▲	▬	▲	!	!	Steps will disturb detection
HAB39	Wall slides	▲	▬	▼	▬	▲	▲	✖	✖	
HAB40	Squat	▲	▲	▬	▬	▬	▲	!	!	Chair can disturb detection
Heup Ac.	Lopen									
HA29	Lopen (met looprek)	▲	▲	▬	▲	▬	▲	!	!	Walker can interfere, range is limited
HA30	Lopen (met rollator)	▲	▲	▼	▲	▬	▲	!	✖	Roller can interfere, range is limited
HA31	Lopen (met elleboogkrukken)	▲	▲	▬	▲	▬	▲	!	!	Crutches can interfere, range is limited
HA32	Lopen (met één elleboogkruk)	▲	▲	▬	▲	▬	▲	!	!	Crutches can interfere, range is limited
HA33	Zijwaarts lopen	▲	▲	▲	▲	▬	▲	✔	✔	range is limited
HA34	Trap oplopen (met aansluitpas)	▲	▲	▬	▬	▬	▲	!	✖	Strairs will interfere with detection
HA36	Trap aflopen (met aansluitpas)	▲	▲	▬	▬	▬	▲	!	✖	Strairs will interfere with detection
HA38	Drempel nemen (met looprek)	▲	▲	▬	▬	▬	▲	!	✖	Walker can interfere
HA39	Drempel nemen (met rollator)	▲	▲	▼	▬	▬	▲	✖	✖	
HA40	Trottoir op (met rollator)	▲	▲	▼	▬	▬	▲	✖	✖	
HA41	Trottoir af (met rollator)	▲	▲	▼	▬	▬	▲	✖	✖	
HAB45	Lopen (met één begeleider)	▲	▲	▼	▬	▼	▲	✖	✖	
Heup C.	Oefeningen in zit									
HC10	Zit, buigen heup	▲	▲	▲	▬	▲	▲	✔	✔	
HC11	Zit, draaien heup naar buiten	▲	▬	▲	▲	▲	▬	✔	!	When legs are appart stool will interfere
HC17	Zit, buigen en strekken van knie	▲	▬	▬	▲	▲	▲	✔	!	
HC18	Zit, knie door strekken met weerstand	▲	▲	▼	▲	▬	▲	!	✖	
HC34	Zit, knie strekken met voet naar buiten gedraaid	▬	▲	▲	▲	▲	▼	!	✖	Rotation of foot is hard to detect
HC35	Zit, knie strekken met weerstan	▲	▲	▬	▲	▲	▬	!	✖	Depends on the type of "Weight"
HC62	Zit, draaien heup naar buiten (zware variant)	▲	▲	▲	▼	▲	▬	!	✖	
Heup C.	Oefeningen in stand									
HC01	Rekken spieren aan binnenzijde bovenbeen	▲	▬	▲	▲	▲	▲	✔	!	Streching the muscle is not detectable
HC02	Schutterstand, strekken heup	▲	▲	▬	▬	▲	▲	✔	!	
HC03	Rekken spieren achterzijde bovenbeen	▲	▬	▬	▬	▲	▲	✔	!	Streching the muscle is not detectable
HC04	Rekken van spier aan voorzijde bovenbeen	▲	▲	▲	▼	▲	▲	!	✖	Lower leg will be occluded by upper leg
HC05	Rekken van spier buitenzijde bovenbeen	▲	▲	▲	▬	▲	▲	✔	✔	
HC12	Stand, buigen heup	▲	▲	▬	▬	▲	▲	✔	!	Chair can interfere, contact lower leg - hands confuses
HC13	Stand, strekken heup	▲	▲	▬	▲	▲	▲	✔	!	Chair can interfere
HC14	Stand, been zijwaarts bewegen	▲	▲	▬	▲	▲	▲	✔	✔	Chair can interfere
HC20	Stand, kuitspier rekken	▲	▬	▲	▲	▲	▲	✔	!	Streching the muscle is not detectable
HC36	Uitvalspas	▲	▲	▲	▲	▲	▲	✔	✔	
HC37	Lunge	▲	▲	▲	▬	▲	▲	✔	!	Lower leg will be occluded by upper leg
HC39	Squat	▲	▬	▬	▲	▲	▲	✔	!	Chair can interfere
HC40	Stand op traptrede, been grond aantikken	▲	▬	▬	▲	▬	▲	✔	!	Steps can interfere
HC41	Step up op traptrede	▲	▲	▬	▲	▬	▲	✔	!	Steps can interfere
HC45	Stand op één been	▲	▬	▲	▲	▲	▲	✔	!	Streching the muscle is not detectable
HC46	Stand op één been, bewegen met armen	▲	▲	▲	▲	▲	▲	✔	✔	
HC47	Stand op één been, bewegen met het andere been	▲	▲	▲	▲	▲	▲	✔	✔	
HC53	Stiff leg good morning	▲	▲	▲	▬	▬	▲	✔	!	Detection of angle of upper body is unreliable
HC54	Deadlift	▲	▲	▬	▲	▲	▲	✔	!	Depends on "weight" used
HC56	Kleine sprong op de plaats	▲	▬	▲	▲	▲	▲	✔	!	Equal power is hard to detect
HC57	Voorwaarts/achterwaarts springen	▲	▬	▲	▲	▲	▲	✔	!	Equal power is hard to detect
HC58	Op één been springen	▲	▬	▲	▲	▲	▲	✔	!	Bending of knee is too subtle
HC59	Vanuit middenpositie zijwaarts springen	▲	▬	▲	▲	▲	▲	✔	!	Bending of knee is too subtle
HC60	Schaatssprongen	▲	▬	▲	▲	▲	▲	✔	!	Bending of knee is too subtle
HC61	Huppelen	▲	▲	▲	▲	▲	▲	✔	✔	
HC64	Stand, bilspier achterzijde heup aanspannen	▲	▬	▼	▲	▲	▲	!	✖	Streching the muscle is not detectable
HC70	Stand, voet op traptrede	▲	▲	▬	▬	▲	▲	!	!	Steps can interfere, lower leg can occlude upper leg
Heup C.	Looptraining									
HC48	Lopen buiten	▲	▲	▲	▲	▬	▲	!	✖	Range is too limited

10.4 Patient information letter

Titel studie

Ontwikkeling van een herkennings- en evaluatie platform gebaseerd op de Kinect diepte camera.

Inleiding

U hebt aangegeven interesse te hebben in deelname aan het hierboven genoemde onderzoek met. Uw toestemming moet u kunnen baseren op goede voorlichting onzerzijds. Daarom ontvangt u deze schriftelijke informatie, die u rustig kunt (her)lezen en in eigen kring bespreken. Ook daarna kunt u altijd nog vragen voorleggen aan de onderzoeker die aan het einde van deze informatie genoemd wordt.

Doel en achtergrond van het onderzoek

Als uitbreiding op de reeds beschikbare thuis trainingssystemen, is een meer interactieve invulling gewenst. Enerzijds om meer informatie te krijgen over de therapietrouw van de patiënt, en anderzijds om de patiënt sneller te voorzien van feedback om zo de training nuttiger te maken.

Wanneer komt u in aanmerking voor het onderzoek?

De studie bestaat uit een experimenten. U komt in aanmerking voor deelname aan experiment als u in staat bent tot het lezen en schrijven van de Nederlandse taal en minimaal 18 jaar oud bent. U kunt helaas niet meedoen aan de experimenten wanneer u lijdt aan een lichamelijke beperking, bijvoorbeeld wanneer u niet normaal kunt lopen, of balans problemen heeft.

Wat houdt het onderzoek voor u in?

Het experiment bestaat uit het 6 keer uitvoeren van de zelfde, korte, oefening voor een Kinect diepte camera. Bij elke uitvoering zullen andere instructies gegeven worden. Alle uitvoeringen worden opgenomen, waarbij u herkenbaar in beeld bent. Deze beelden worden alleen voor analyse gebruikt. Beelden die in media gebruikt worden, zullen eerst gecensureerd worden.

Risico's

Er zijn geen risico’s verbonden aan het experiment.

Mogelijke voordelen

Dit onderzoek draagt voornamelijk bij aan de ontwikkeling van nieuwe technieken en methoden waar mogelijk in de toekomst andere personen voordeel van kunnen hebben.

Vertrouwelijkheid

De gegevens die gedurende het onderzoek over u verzameld worden zullen vertrouwelijk behandeld worden. De gegevens zullen zodanig gecodeerd worden dat ze niet tot u te herleiden zijn. De codering is dan ook niet gebaseerd op bijvoorbeeld geboortedatum, initialen en geslacht.

Vrijwillige deelname

U bent vrij deelname aan dit onderzoek toe te staan of te weigeren. Ook indien u nu toestemming geeft, kunt u te allen tijde zonder opgave van redenen weer intrekken.

Voor nadere informatie

Indien u nog vragen heeft, kunt u die voorleggen aan de verantwoordelijke onderzoeker

Frodo Muijzer, BSc.

Roessingh Research and Development

Roessinghsbleekweg 33b

7522 AH Enschede

Telefoon: 0621711087

Mail: f.muijzer@rrd.nl

Hierbij verklaart u de Proefpersoneninformatie gelezen te hebben, en akkoord te zijn met de bepalingen die hierin gesteld zijn:

Proefpersoon: ______________

Handtekening: ______________

Datum: ______________

Plaats: ______________

10.5 Experiment explanation for test subject

10.5.1 First experiment

Oefening Rekken van spier buitenzijde bovenbeen

Start in normale stand houding, met beide voeten naast elkaar. Zwaai het rechter been voor het linker been. Beweeg dan de heupen naar links, en tegelijkertijd het bovenlichaam naar rechts.

10.5.2 Second experiment

Bekijk oefening Rekken van spier buitenzijde bovenbeen (hc05)

Inleiding

Deze oefening heeft als doel het verlengen van de spier aan de buitenzijde van het bovenbeen. Het behoud van de lengte van deze spier is essentieel voor de beweging die u tijdens het looppatroon moet maken.

Uitvoering

Uitgangshouding:

In stand rechtop, beide voeten plat op de grond.

Kruis uw rechter been voor uw linker been langs en zet de voeten naast elkaar neer. Houdt hierbij de knieën gestrekt. Beweeg nu met uw heupen zijwaarts richting uw linker been waarbij u uw bovenlichaam in tegengestelde richting beweegt. Als het goed is voelt u nu de rek aan de buitenzijde van uw bovenbeen. Houdt deze rek enkele tellen vast. Beweeg vervolgens weer rustig terug naar de beginpositie.

Let op

Compensatiebewegingen

Voldoende rek aan buitenzijde van het bovenbeen

Voor meer rek kan u met de arm van de linker zijde boven uw hoofd mee bewegen naar de rechter zijde

Blijf recht naar voren kijken

Overig

Bij deze omschrijving hoort een video, die u dient te bekijken.

10.6 Experiment results

	Segment 1						Segment 2						Segment 3					
01	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	1	1	0	0	1	0	1	1	0	0	1
2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	2	0	0	0	3	1	2	0	0	1	1	1	1	0
4	6 (KL, FR)						12 (ER, HaR, KL, KR)						3 (KR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	840	833	844	24	12	60	0	0	0	825	817	835	36	18	90	0	0	0	835	828	847	12	15	77

14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	-1	0

02	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	1	1	0	0	1	0	1	1	0	0	1
2	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
3	2	1	0	0	0	0	4	1	4	0	0	0	1	0	0	0	0	0
4	2 (KL, KR)						15 (SR, ER, HR, KL, KR, FR)						1 (KR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	836	829	843	24	12	60	0	0	0	836	809	852	36	24	120	0	0	0	829	820	837	12	14	76

14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0

03	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
3	0	0	0	0	0	0	4	0	0	0	0	0	2	0	0	0	0	0
4	3 (KL, KR, FL)						13 (SR, FL, KR)						2 (KR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	823	813	834	24	12	60	0	0	0	836	803	857	36	29	145	0	0	0	827	820	832	24	24	120

14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0

04	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0
2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0
3	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
4	2 (KR)						5 (SR, KL, KR)						2 (KR, FR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	834	825	843	30	15	75	0	0	0	836	825	848	12	12	60	0	0	0	837	816	867	0	18	90

14	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

05	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
3	2	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
4	2 (FL, KR)						9 (SL, KL, FL, KR, FR)						2 (KL, FR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	823	818	827	24	12	60	0	0	0	835	825	841	24	18	90	0	0	0	833	816	848	12	17	85

14	0	0	0	0	0	0	0	-1	-1	0	0	0	0	-1	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

06	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0
3	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
4	1 (KR)						9 (SL, HaL, KL, FL, FR)						0 ()					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	822	817	827	0	0	0	0	0	0	836	821	857	36	18	90	0	0	0	830	816	852	24	16	85

14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	1	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

S2	Segment 1						Segment 2						Segment 3					
01	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
3	1	0	0	0	0	0	3	0	0	0	0	0	2	0	0	0	0	0
4	3 (KR)						9 (HaR, KR)						2 (HaR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	827	817	833	18	22	110	0	0	0	836	802	861	42	53	265	0	0	0	833	821	857	30	30	150

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	-1	-1	0	0	0	0	-4	-3	0	0	0	0	-5	0	0	0	0
15	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
02	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	0
3	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
4	4 (HaL, KR)						8 (HaL, HaR, KR)						3 (HaL, KL, KR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	838	730	946	24	12	60	0	0	0	830	807	859	0	18	90	0	0	0	833	809	855	12	23	115

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	0	0	0	0	0	0	-3	-3	0	0	0	0	-1	-3	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
03	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
3	0	1	0	0	0	0	5	1	0	0	0	0	0	3	0	0	0	0
4	0 ( )						13 (HaL, HaR, KL, KR)						10 (HaL, HaR, KL, KR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	840	822	860	48	24	120	0	0	0	830	803	845	42	41	205	0	0	0	832	805	853	24	35	175

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	0	0	0	0	0	0	0	-5	0	0	0	0	-1	-4	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
04	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
3	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0
4	0 ( )						5 (KL, KR, FR)						4 (KL, KR, FR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	846	838	854	24	12	60	0	0	0	829	798	843	0	30	150	0	0	0	834	808	850	0	18	90

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	0	0	0	0	0	0	-5	-5	0	0	0	0	-3	-3	-1	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
05	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	1	0	0
3	1	1	0	0	0	0	1	1	0	1	0	0	0	4	0	2	0	0
4	0 ( )						7 (SL, HaL, KL)						12 (SL, HaL, KL, FL)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	824	798	862	48	24	120	0	0	0	837	817	855	30	29	145	0	0	0	833	815	860	12	42	210

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	-3	0	0	0
15	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
06	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**	**B**	**AL**	**AR**	**LL**	**LR**	**UB**
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0
3	0	0	0	0	0	0	1	0	0	0	0	0	2	0	0	0	0	0
4	0 ( )						3 (SL, HaL, KL)						7 (HaL, KL)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	838	835	840	0	0	0	0	0	0	836	824	854	48	24	120	0	0	0	829	813	845	12	24	120

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	0	0	0	0	0	0	0	0	0	0	0	0	-3	-3	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

S3	Segment 1						Segment 2						Segment 3					
01	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
4	0 ( )						1 (FL)						1 (KR)					
14	0	0	0	0	0	0	0	0	0	0	0	0	-1	-1	0	0	0	-1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

01	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	830	812	848	12	6	30	0	0	0	837	821	853	24	11	55	0	0	0	833	826	838	24	17	91

02	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
3																		
4																		
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

02	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
																			0	0	0	832	824	839	0	0	0

03	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
4	0 ( )						0 ( )						0 ( )					
14	0	0	0	0	0	0	0	-5	0	0	0	0	0	-5	-4	0	0	0
15	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

03	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	831	813	844	36	18	90	0	0	0	835	826	850	36	35	175	0	0	0	833	790	868	6	30	150

04	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0	0	0	0	0	2	1	2	0	0	0
4	0 ( )						7 (KL, FL, KR, FR)						9 (SR, HaR, FL, FR)					
14	0	0	0	0	0	0	0	0	0	0	0	0	-2	-2	-1	-2	0	-2
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

04	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	833	804	870	12	6	30	0	0	0	825	809	849	48	24	120	0	0	0	838	825	859	12	21	111

05	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	1	0	0	0	0	2	1	0	2	0	0	0	1	0	0	0	0
4	5 (KL, FL, FR)						13 (HaL, FL, KR, FR)						0 ( )					
14	0	0	0	0	0	0	-2	-2	-2	-3	-2	-2	-3	0	0	-3	0	1
15	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0

05	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	831	821	846	36	18	90	0	0	0	829	813	856	0	28	143	0	0	0	836	827	848	18	18	90

06	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	1	0	0	0	0	3	1	0	1	0	0	2	0	0	0	0	0
4	1 (FR)						7 (HaL, KL, FR)						3 (HaL, KL)					
14	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	-1	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

06	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	827	794	848	24	12	60	0	0	0	837	558	1107	36	18	90	0	0	0	832	809	853	6	16	80

S4	Segment 1									Segment 2									Segment 3								
01	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**
1	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0
2	0		0	0		1	1		0	0		0	0		1	1		0	0		0	0		1	1		0
3	0		1	0		0	0		0	0		1	0		0	0		0	1		1	1		0	0		0
4	1 (KR)									5 (HaR, KL, KR)									4 (HaR, KL, KR)								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	824	798	848	36	18	90	0	0	0	831	814	852	24	24	120	0	0	0	839	818	855	0	23	115
14	0		0	0		0	0		0	0		-2	-2		0	0		0	0		-3	-3		0	0		0
15	0		0	0		0	0		0	0		0	0		0	0		0	0		1	0		0	0		0
02	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**
1	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0
2	0		0	0		0	0		0	0		0	0		1	1		0	0		0	0		1	1		0
3	0		0	0		0	0		0	0		0	0		0	0		0	0		1	0		0	0		0
4	0 ( )									0 ( )									3 (HaR, KL, KR)								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	850	839	860	24	12	60	0	0	0	821	805	837	24	12	60	0	0	0	830	800	856	12	18	90
14	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0
15	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0
03	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**
1	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0
2	0		0	0		0	0		0	0		0	0		1	1		0	0		0	0		1	1		0
3	0		2	0		0	0		0	0		2	0		0	0		0	0		1	0		0	0		0
4	2 (FR)									6 (HaR, KL, FR)									2 (KL)								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	837	811	851	36	29	145	0	0	0	832	819	843	18	42	210	0	0	0	833	820	854	30	35	175
14	0		0	0		0	0		0	0		0	0		0	0		0	0		-2	-2		0	0		0
15	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0
04	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**
1	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0
2	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		1	1		0
3	2		0	0		0	0		0	2		0	0		0	0		0	1		0	0		0	0		0
4	1 (KR)									7 (HaR, KL, KR)									0 ( )								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	831	800	863	24	12	60	0	0	0	835	817	860	24	21	110	0	0	0	821	804	831	0	19	91
14	0		0	0		0	0		0	0		-2	-2		0	0		0	-2		0	0		-3	-3		-2
15	0		0	0		0	0		0	0		0	0		0	0		0	0		1	0		0	0		0
05	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**
1	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		1
2	0		0	0		0	0		0	0		0	0		1	1		0	0		0	0		1	0		0
3	1		1	0		0	0		0	0		1	0		0	0		0	0		0	0		0	0		0
4	2 (KL)									4 (KL, KR)									2 (KR)								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	835	827	844	48	24	120	0	0	0	829	809	854	12	24	120	0	0	0	833	808	877	6	35	175
14	0		0	0		0	0		0	0		-3	-3		0	0		0	0		-4	-5		0	0		0
15	0		0	0		0	0		0	0		0	0		0	0		0	0		1	0		0	0		0
06	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**	**B**		**AL**	**AR**		**LL**	**LR**		**UB**
1	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		0	0		1
2	0		0	0		0	0		0	0		0	0		1	1		0	0		0	0		1	1		0
3	1		2	0		0	0		0	0		2	0		0	0		0	0		0	0		0	0		0
4	1 (KL)									5 (SL, KL)									1 (KL)								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	840	826	854	24	12	60	0	0	0	827	695	986	24	18	90	0	0	0	830	812	849	12	22	115
14	0		0	0		0	0		0	0		-1	-1		0	0		0	-2		0	0		-2	-1		-1
15	0		0	0		0	0		0	0		0	0		0	0		0	0		1	0		0	1		0

<table>
<tr><th>S5</th><th colspan="9">Segment 1</th><th colspan="9">Segment 2</th><th colspan="9">Segment 3</th></tr>
<tr><th>01</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th></tr>
<tr><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>0</td></tr>
<tr><td>2</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>0</td></tr>
<tr><td>3</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>4</td><td colspan="9">2 (KL, KR)</td><td colspan="9">2 (FL, KR)</td><td colspan="9">0 ( )</td></tr>
<tr><td></td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td></tr>
<tr><td></td><td>0</td><td>0</td><td>0</td><td>839</td><td>808</td><td>871</td><td>24</td><td>12</td><td>60</td><td>0</td><td>0</td><td>0</td><td>827</td><td>817</td><td>839</td><td>12</td><td>18</td><td>90</td><td>0</td><td>0</td><td>0</td><td>835</td><td>811</td><td>861</td><td>12</td><td>23</td><td>115</td></tr>
<tr><td>14</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>-2</td><td colspan="2">-2</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>-1</td><td colspan="2">-3</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>15</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><th>02</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th></tr>
<tr><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>1</td></tr>
<tr><td>2</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>3</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>4</td><td colspan="9">0 ( )</td><td colspan="9">3 (HaL, FL, KR)</td><td colspan="9">0 ( )</td></tr>
<tr><td></td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td></tr>
<tr><td></td><td>0</td><td>0</td><td>0</td><td>827</td><td>827</td><td>827</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>830</td><td>830</td><td>831</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>825</td><td>818</td><td>831</td><td>24</td><td>12</td><td>60</td></tr>
<tr><td>14</td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td></tr>
<tr><td>15</td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td><td colspan="2"></td><td></td></tr>
<tr><th>03</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th></tr>
<tr><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>2</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>1</td><td colspan="2">1</td><td>0</td></tr>
<tr><td>3</td><td colspan="2">1</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">4</td><td>0</td><td colspan="2">3</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>4</td><td colspan="9">2 (KL, KR)</td><td colspan="9">16 (SR, ER, HaR, KL, KR)</td><td colspan="9">3 (KL, KR)</td></tr>
<tr><td></td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td></tr>
<tr><td></td><td>0</td><td>0</td><td>0</td><td>821</td><td>797</td><td>846</td><td>12</td><td>6</td><td>30</td><td>0</td><td>0</td><td>0</td><td>836</td><td>816</td><td>867</td><td>48</td><td>42</td><td>210</td><td>0</td><td>0</td><td>0</td><td>829</td><td>815</td><td>840</td><td>0</td><td>30</td><td>150</td></tr>
<tr><td>14</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>-3</td><td colspan="2">-3</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>-5</td><td colspan="2">-5</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>15</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><th>04</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th></tr>
<tr><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>2</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>1</td><td colspan="2">1</td><td>0</td></tr>
<tr><td>3</td><td colspan="2">0</td><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>1</td><td colspan="2">2</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>4</td><td colspan="9">3 (KL, FL, KR)</td><td colspan="9">4 (ER, HaR, KR)</td><td colspan="9">1 (KR)</td></tr>
<tr><td></td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td></tr>
<tr><td></td><td>0</td><td>0</td><td>0</td><td>829</td><td>827</td><td>831</td><td>12</td><td>6</td><td>30</td><td>0</td><td>0</td><td>0</td><td>830</td><td>809</td><td>852</td><td>24</td><td>12</td><td>60</td><td>0</td><td>0</td><td>0</td><td>835</td><td>819</td><td>848</td><td>30</td><td>21</td><td>111</td></tr>
<tr><td>14</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">-1</td><td>-1</td><td colspan="2">0</td><td>-1</td><td colspan="2">0</td><td>-1</td></tr>
<tr><td>15</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>0</td><td colspan="2">1</td><td>0</td></tr>
<tr><th>05</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th></tr>
<tr><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>2</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>0</td><td colspan="2">1</td><td>0</td><td colspan="2">0</td><td>1</td><td colspan="2">1</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>1</td><td colspan="2">1</td><td>0</td></tr>
<tr><td>3</td><td colspan="2">0</td><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">2</td><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>4</td><td colspan="9">1 (KR)</td><td colspan="9">8 (SL, KL, FL, KR, FR)</td><td colspan="9">2 (KR)</td></tr>
<tr><td></td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td></tr>
<tr><td></td><td>0</td><td>0</td><td>0</td><td>818</td><td>771</td><td>846</td><td>12</td><td>6</td><td>30</td><td>0</td><td>0</td><td>0</td><td>831</td><td>816</td><td>846</td><td>48</td><td>30</td><td>150</td><td>0</td><td>0</td><td>0</td><td>833</td><td>819</td><td>864</td><td>0</td><td>30</td><td>150</td></tr>
<tr><td>14</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>-1</td><td colspan="2">-1</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>-5</td><td colspan="2">-5</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>15</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><th>06</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th><th colspan="2">B</th><th>AL</th><th colspan="2">AR</th><th>LL</th><th colspan="2">LR</th><th>UB</th></tr>
<tr><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>1</td></tr>
<tr><td>2</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">1</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>3</td><td colspan="2">1</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>1</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>4</td><td colspan="9">2 (KR, FR)</td><td colspan="9">1 (SL)</td><td colspan="9">2 (SL, KR)</td></tr>
<tr><td></td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td></tr>
<tr><td></td><td>0</td><td>0</td><td>0</td><td>834</td><td>822</td><td>845</td><td>12</td><td>6</td><td>30</td><td>0</td><td>0</td><td>0</td><td>839</td><td>836</td><td>843</td><td>24</td><td>12</td><td>60</td><td>0</td><td>0</td><td>0</td><td>831</td><td>822</td><td>844</td><td>24</td><td>24</td><td>120</td></tr>
<tr><td>14</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>-2</td><td colspan="2">-2</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
<tr><td>15</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td><td colspan="2">0</td><td>0</td></tr>
</table>

S6	Segment 1						Segment 2						Segment 3					
01	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	1	1	0	1	0	0	1	1	0	1	1	0	1	1	0	1	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1 (KR)						1 (FL)						0 ( )					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	832	832	832	0	0	0	0	0	0	835	835	835	12	6	30	0	0	0	836	821	850	24	12	60

14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
02	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	1	1	0	1	0	0	1	1	0	1	1	0	1	1	0	1	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0 ( )						0 ( )						1 (HaR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13

14																		
15																		
03	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	0	0	0	0	2	1	0	0	0	0	2	1	0	0	0	0
4	5 (KL, FL, KR)						18 (HaL, SR, HaR, KL, KR)						2 (KR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	843	839	849	12	6	30	0	0	0	829	816	843	48	36	180	0	0	0	830	822	841	0	24	120

14	0	0	0	0	0	0	0	-2	-2	0	0	0	0	-4	-4	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
04	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
3	1	0	0	0	0	0	1	0	0	2	0	0	1	0	0	0	0	0
4	1 (KR)						13 (HaL, SR, HaR, KR)						1 (KR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	832	832	833	0	0	0	0	0	0	833	812	854	36	18	90	0	0	0	833	825	840	24	15	84

14	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
05	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	1	1	0
3	2	0	0	0	0	0	5	0	0	0	0	0	1	0	0	0	0	0
4	1 (KL)						16 (SL, HaL, HaR, KL, KR)						2 (KR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	833	823	843	0	0	0	0	0	0	835	804	855	48	24	120	0	0	0	834	818	854	48	12	118

14	0	0	0	0	0	0	0	0	0	0	0	0	-3	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	-1
06	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	0
3	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0
4	6 (SL, HaL, HaR, KL, KR)						9 (HaL, HaR, KL)						5 (HaL, HaR, KL, FL)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	848	825	856	0	0	0	0	0	0	827	818	837	36	18	90	0	0	0	830	804	868	36	26	134

14	0	0	0	0	0	0	0	0	0	0	0	0	-2	0	0	-1	-1	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	-1

S7	Segment 1									Segment 2									Segment 3								
01	**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**			
1	0	0	0	0	0	0				0	0	0	0	0	0				0	0	0	0	0	0			
2	0	0	0	1	1	0				0	0	0	1	1	0				0	0	0	1	1	0			
3	1	0	0	0	0	0				4	0	0	0	0	0				5	3	0	0	0	0			
4	2 (KR)									3 (SR, KR)									4 (KL, KR)								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	833	816	856	36	24	120	0	0	0	831	816	848	0	24	120	0	0	0	836	792	849	30	28	140
14	0	-1	-1	0	0	0				0	-4	-4	0	0	0				0	-1	-1	0	0	0			
15	0	0	0	0	0	0				0	0	0	0	0	0				0	1	1	0	0	0			
02	**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**			
1	0	0	0	0	0	0				0	0	0	0	0	0				0	0	0	0	0	0			
2	0	0	0	1	1	0				0	0	0	1	1	0				0	0	0	1	1	0			
3	0	1	0	0	0	0				1	1	0	0	0	0				2	0	0	0	0	0			
4	3 (KL, FL, KR)									2 (KL, KR)									2 (KR)								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	824	796	859	36	18	90	0	0	0	828	816	839	12	8	41	0	0	0	839	818	861	6	15	74
14	0	0	0	0	0	0				0	0	0	0	0	0				0	-1	-2	0	0	0			
15	0	0	0	0	0	0				0	0	0	0	0	0				0	1	0	0	0	0			
03	**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**			
1	0	0	0	0	0	0				0	0	0	0	0	0				0	0	0	0	0	0			
2	0	0	0	0	0	0				1	0	0	1	1	0				1	0	0	0	1	0			
3	0	0	0	0	0	0				2	0	0	0	0	0				6	2	0	0	0	0			
4	4 (KL, KR)									7 (KL, KR)									11 (SR, HaR, KL, FL, KR)								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	838	819	873	54	35	175	0	0	0	824	797	841	24	30	150	0	0	0	836	801	855	0	60	300
14	0	0	-1	0	0	0				0	-1	-4	0	0	0				0	-9	-9	0	0	0			
15	0	1	0	0	0	0				0	0	0	0	0	0				0	0	0	0	0	0			
04	**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**			
1	0	0	0	0	0	0				0	0	0	0	0	0				0	0	0	0	0	0			
2	0	0	0	0	0	0				0	0	0	1	1	0				0	0	0	0	0	0			
3	2	0	0	0	0	0				1	0	0	0	0	0				1	1	0	0	0	0			
4	4 (KL, KR)									4 (SR, KR)									3 (KL, KR)								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	868	822	955	36	23	115	0	0	0	789	687	844	12	14	75	0	0	0	829	813	844	6	38	190
14	0	-1	0	0	0	0				0	-2	0	0	0	0				-5	-1	0	-6	-3	-5			
15	0	0	1	0	0	0				0	0	0	0	0	0				0	0	0	0	1	0			
05	**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**			
1	0	0	0	0	0	0				0	0	0	0	0	0				0	0	0	0	0	1			
2	0	0	0	1	1	0				0	0	0	1	1	0				0	0	0	1	1	0			
3	2	0	0	0	0	0				0	0	0	0	0	0				0	0	0	0	0	0			
4	3 (KL, KR)									2 (KR)									5 (KL, KR)								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	834	821	849	60	36	180	0	0	0	830	812	855	0	24	120	0	0	0	834	800	867	0	42	210
14	0	-1	-1	0	0	0				0	-4	-4	0	0	0				0	-7	-7	0	0	0			
15	0	0	0	0	0	0				0	0	0	0	0	0				0	0	0	0	0	0			
06	**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**				**B**	**AL**	**AR**	**LL**	**LR**	**UB**			
1	0	0	0	0	0	0				0	0	0	0	0	0				0	0	0	0	0	1			
2	0	0	0	1	1	0				0	0	0	1	1	0				0	0	0	1	1	0			
3	0	0	0	0	0	0				0	0	0	0	0	0				0	0	0	0	0	0			
4	2 (KL)									3 (SL, KL)									6 (EL, KL, FL, KR)								
	5	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*
	0	0	0	830	812	844	48	24	120	0	0	0	834	818	850	12	12	60	0	0	0	834	820	852	12	35	175
14	0	0	0	0	0	0				0	-1	-1	0	0	0				0	-3	-5	0	0	0			
15	0	0	0	0	0	0				0	0	0	0	0	0				0	1	0	0	0	0			

S8	Segment 1						Segment 2						Segment 3					
01	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0
3	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
4	3 (KL, KR)						1 (KL)						6 (KL, KR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	838	813	857	36	18	90	0	0	0	826	816	842	12	16	84	0	0	0	835	813	870	228	103	515

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	0	0	0	0	0	0	0	-1	1	0	0	0	0	0	-3	-3	0
15	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0

02	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
3	2	1	0	0	0	0	2	1	0	0	0	0	3	0	0	0	0	0
4	4 (KL, KR)						5 (KR)						2 (KR)					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	835	825	855	60	30	150	0	0	0	830	814	844	6	29	145	0	0	0	834	818	849	24	30	150

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	0	0	0	0	0	0	0	0	-2	-1	0	0	0	0	-2	-2	0
15	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

03	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	1	1	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0 ( )						0 ( )						0 ( )					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	835	794	897	24	13	69	0	0	0	831	817	848	36	39	192	0	0	0	833	820	850	36	28	140

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	0	0	0	0	0	1	-1	0	0	-2	-2	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0

04	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	0
3	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
4	0 ( )						2 (FL, FR)						0 ( )					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	845	845	845	0	0	0	0	0	0	833	806	855	60	48	240	0	0	0	832	815	865	24	28	140

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	0	0	0	0	0	0	0	0	0	0	0	0	-2	-2	0	0	0
15	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	1	0	0	0

05	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	1	0	0	0	0	0	1	0	0	0	0	0	2	0	0	0	0
4	1 (FR)						8 (SL, HaL, HaR, FR)						0 ( )					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	827	826	827	24	12	60	0	0	0	834	812	858	12	30	150	0	0	0	833	823	852	0	30	150

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	0	0	0	0	0	0	0	0	-3	-3	0	0	0	0	-3	-3	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

06	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
4	1 (FR)						0 ( )						0 ( )					

	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13	5	6	7	8	9	10	11	12	13
	0	0	0	855	855	855	12	6	30	0	0	0	824	811	836	24	12	60	0	0	0	837	834	844	24	18	90

	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB	B	AL	AR	LL	LR	UB
14	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	-1	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0