Enabling Real-Time Automatic Assessment of Patient Exercises for Technology-Assisted Physical Rehabilitation Interventions

JOE SARSFIELD

School of Science and Technology

A thesis submitted in partial fulfilment of the requirements of

Nottingham Trent University for the degree of

Doctor of Philosophy

November 2018

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisors, Prof. David Brown, Prof. Nasser Sherkat and Mr James Lewis. Even during their busy schedules they have always been accessible for support and feedback during my four years of study. I would also like to extend my sincere gratitude to Dr Caroline Langensiepen for her invaluable guidance and support during my research. Her guidance was thorough and continued into her retirement, for which I am truly grateful.

I would like to give a special thanks to Mrs Louise Selwood for kindly giving up her time to help with patient recruitment, data collection and providing insight into how stroke care is delivered within the NHS.

Finally, I would like to thank the many other researchers and practitioners who have helped contribute to my research.

Abstract

Technology-assisted physical rehabilitation interventions (TAPRI) have the potential to offer patients a safe, motivating and always accessible platform for undergoing rehabilitation. The emergence of compact and affordable depth sensors provide an opportunity to realise such interventions in a home environment. These types of depth sensors can run pose estimation algorithms that track full-body human joint positions in real-time. TAPRIs that provide real-time patient performance assessment and feedback require sufficiently accurate algorithms to ensure a correct assessment.

The research presented in this thesis aims to overcome some of the algorithmic challenges in enabling real-time patient performance assessment and feedback. This research focuses on two algorithms: real-time tracking of human joint positions and real-time segmentation of exercise repetitions. This research targets stroke rehabilitation as a challenging use case for achieving real-time patient exercise assessment as stroke patients often have varying levels of mobility.

Research contributions. The first contribution of this thesis is a quantitative and clinical evaluation of a state-of-the-art pose estimation algorithm (human joint tracking) to determine if the joint position estimations are sufficiently accurate for correctly assessing stroke rehabilitation exercises. This evaluation also determines what the limitations are and propose recommendations for future pose estimation algorithms intended for clinical applications. The second contribution is an evaluation of the inter-rater reliability of clinicians assessing the suitability of the pose estimation algorithm, to quantitatively determine where the clinicians are in agreement and propose more robust criteria for the assessment of new clinical technologies. The final contribution is the proposal of a real-time segmentation algorithm that requires only a single exemplar repetition of an exercise to segment repetitions from other subjects, including those with impaired mobility.

Main research findings and results. The accuracy of current state-of-the-art pose estimation algorithms are insufficient for correctly assessing patient performance. There was a low inter-rater agreement between clinicians evaluating the accuracy of the individual joints of a state-of-the-art pose estimation algorithm,

however overall the accuracy was found to be insufficient. Our proposed segmentation algorithm correctly segments ≈90% of stroke patient exercise repetitions from our own rehabilitation exercise dataset and is capable of segmenting a 20 second window at 30Hz in real-time on a desktop computer.

Publications

The following research papers have been produced as a result of the research presented in this thesis:

Refereed Journal Papers

J. Sarsfield, D. Brown, N. Sherkat, C. Langensiepen, J. Lewis, M. Taheri, C. McCollin, C. Barnett, L. Selwood, P. Standen, P. Logan, C. Simcox, C. Killick and E. Hughes, "Clinical Analysis of Depth-Sensor Based Pose Estimation Algorithms for Technology Enhanced Rehabilitation Applications," International Journal of Medical Informatics. DOI https://doi.org/10.1016/j.ijmedinf.2018.11.001, [1] (published).

J. Sarsfield, D. Brown, N. Sherkat, C. Langensiepen, J. Lewis, M. Taheri, L. Selwood, P. Standen and P. Logan, "Segmentation of Exercise Repetitions Enabling Real-Time Patient Analysis and Feedback Using a Single Exemplar," IEEE Transactions on Neural Systems and Rehabilitation Engineering, [2] (published).

Refereed Conference Papers

J. Sarsfield, D. Brown, N. Sherkat, C. Langensiepen, J. Lewis and P. Standen, "Reducing Clinical Subjective Discrepancies in Evaluation of Clinical Technology using Objective Measures," 12th International Conference on Disability, Virtual Reality and Associated Technologies, [3] (published).

Contents

Nomenclature

AAT	Atomic Action Template
ANN	Artificial Neural Network
ARAT	Action Research Arm Test
AT	Acceptable Tracking
BBT	Box and Block Test
C	Candidate
CAD	Cornell Activity Dataset
CMU	Carnegie Mellon University motion capture database
DB	Distance Base
DF	DTW Motion Feature
DM	Distance Multiplier
DTW	Dynamic Time Warping
F	Motion Feature
FMA	Fugl-Meyer Assessment
FR	Feature Rank
GT	Ground Truth
HMM	Hidden Markov Model
IMU	Inertial Measurement Unit
k-NN	k-Nearest-Neighbour
M	Mean
MAE	Mean Average Error
MHAD	Berkeley Multimodal Human Action Database
MSR	Microsoft Research
MSRC	Microsoft Research Cambridge Dataset
MT	Moderately Acceptable Tracking
PCA	Principle Component Analysis
PDF	Probability Density Function
PEA	Pose Estimation Algorithm
PRMD	University of Idaho - Physical Rehabilitation Movements Dataset
Q	Query
SD	Standard Deviation
ST	Segment Threshold
STREAM	Stroke Rehabilitation Assessment of Movement
SVM	Support Vector Machine
TAPRI	Technology-Assisted Physical Rehabilitation Intervention
TCT	Total Cost Threshold
TE	Time Error
TSRP	Toronto Rehabilitation Stroke Pose Dataset
UT	Unacceptable Tracking
ZVC	Zero Velocity Crossings

List of Figures

List of Tables

1 Introduction

Physical rehabilitation is the action of restoring functional movement and mobility lost due to disability, disease or injury. Contact time with physiotherapists can often be limited with patients expected to continue their exercise program daily at home unsupervised. In order to solve the problem of unsupervision during exercising this thesis investigates the challenges in enabling technology-supervised rehabilitation. Solutions to algorithmic and design challenges are required to enable practical real-time automatic assessment of patient exercises for technology-assisted physical rehabilitation interventions (TAPRI). This research concentrates on upper-body stroke rehabilitation as a challenging use case for TAPRI, as stroke patients can have a low range of motion, diverse levels of impairment and communication problems (aphasia). Upper-body rehabilitation is targeted because upper limb weakness is the most common impairment after stroke [5]. This chapter discusses the background and impact of stroke, and provide a research overview including the aims, objectives and contributions of the thesis.

1.1 Background and societal impact of stroke

Stroke was the third most common cause of disability worldwide in 2010 [6], ranked by disability-adjusted life years. Only 31% of stroke patients adhere to their exercise program [7] even though evidence suggests that they could recover significant motor control through high-intensity and repetitive task-specific practice [8], [9]. Depression, boredom and perceived lack of benefit [10] have been reported by patients as reasons for low adherence of exercise programs. Although exercises need to be performed on a daily basis, contact time with physiotherapists is often limited with NHS care usually ceasing within 6 months [11]; even though gains in motor function can occur years after stroke, given an appropriately structured rehabilitation program [12]. During rehabilitation it is important to exercise in a correct manner to encourage positive motor recovery - incorrect rehabilitation can lead to reduced range of motion and pain [13]. Many of these incorrect movements are known as compensatory movements whereby the correct muscle groups are not being activated due to abnormal movement patterns. Consistant and effective stroke rehabilitation interventions would improve recovery and increase the likelyhood of regaining independence; thus reduce the cost of care (£23,315 per stroke patient [14]) and improve quality of life.

TAPRIs have the potential to offer patients an engaging platform for safely performing their rehabilitation exercise programs at home or in the community. The following list presents the key aspects that a TAPRI could provide:

- A dynamic exercise program that adapts the challenge to the patients performance and functional movement requirements.

- Real-time automatic assessment of patient exercises to enable timely feedback for encouraging correct functional movements.
- An interface that engages the patient in a manner that improves adherence and responsiveness to feedback.
- Load-balancing of physiotherapists' time to target patients with low adherence and/or recovery.

1.2 Research overview

Several technological and algorithmic advances have led to a promising avenue for realising technology-supervised rehabilitation. Specifically the introduction of commercial depth sensors, such as Microsoft Kinect, Intel RealSense and Occipital Structure, that offer an affordable, practical and compact sensor for generating depth maps in real-time. These depth maps can be used as input into a pose estimation algorithm to predict human joint positions. This was first shown in Jamie Shotton's seminal paper [15] that proposes a real-time pose estimation algorithm requiring only a single depth map generated from a depth-sensor. This approach provides a promising avenue for practical clinical applications that require human joint position tracking, but it is important that the accuracy of the joint positions are clinically validated. TAPRIs that aim to provide real-time automatic assessment of patient exercises and feedback require accurate joint position estimations and segmentation of exercise repetitions. The research presented in this thesis has concentrated on evaluating and resolving some of these algorithmic challenges for enabling real-time automatic assessment of patient exercises. The research questions addressed in this thesis are:

- Are depth-sensor-based state-of-the-art pose estimation algorithms accurate enough for assessing the quality of an exercise and what are the limitations?

- What is the inter-rater agreement of physiotherapists assessing the suitability of a pose estimation algorithm for clinical use?
- Can exercise repetitions from subjects with impaired mobility be accurately segmented in real-time?

1.3 Research aims and objectives

The following research aims and objectives relate to the algorithmic requirements for enabling accurate real-time automatic assessment of patient exercises for TAPRIs. This research aims to:

- Discover the accuracy and limitations of current state-of-the-art pose estimation algorithms (PEA) for the clinical assessment of exercises.
- Determine the inter-rater agreement among clinicians evaluating the accuracy of a PEA in terms of performing an accurate clinical assessment.
- Propose a real-time segmentation algorithm capable of segmenting exercise repetitions from subjects with diverse levels of impairment.

The following objectives were identified to accomplish these aims:

- Collect a dataset containing joint position data of stroke patients performing rehabilitation exercises captured from a consumer depth sensor.
- Evaluate the results of a clinical evaluation (from several physiotherapists) on the current state-of-the-art PEA to determine whether the accuracy is suitable for correctly assessing the quality of stroke rehabilitation exercises and determine what the limitations are.
- Evaluate the physiotherapists' analyses of the PEA's joint tracking accuracy to determine the inter-rater agreement and find areas of agreement and disagreement.

- Design, develop and evaluate a real-time segmentation algorithm on real stroke patient exercise data and a public dataset.
- Evaluate the performance of the segmentation algorithm against other proposed algorithms. Determine the strengths and weaknesses of the proposed approach.

1.4 Research contribution

The main research contributions are:

- A clinical and quantitative analysis of a depth-sensor based state-of-the-art PEA to determine whether the joint position estimations are accurate enough to correctly assess an exercise and what the limitations are, published in [1].
- An analysis of the inter-rater agreement among physiotherapists rating the accuracy of joint position estimations in the context of assessment of exercises, published in [3].
- A real-time segmentation algorithm that requires only a single exemplar repetition to segment repetitions from other subjects including those with impaired mobility, published in [2].

1.5 Thesis outline

The following chapters are summarised as follows:

- Chapter 2 Background and Related Work: This chapter reviews: different compact and cost-effective depth sensors; current and proposed TAPRIs; different data representations/descriptors of human motion; analyses of PEAs; inter-rater agreement of clinicians; algorithms for segmenting human motion;

and publicly available action/exercise datasets.

- Chapter 3 New Ideas - This chapter discusses in more detail the research undertaken in this thesis.
- Chapter 4 Clinical Evaluation of Kinect's PEA: This chapter describes: the experimental set-up of the evaluation; the methodology of the clinical evaluation; the methodology of the quantitative analysis; describes the methodology for evaluating the inter-rater agreement among the physiotherapists clinically evaluating the joint position estimations; presents and evaluates the results of the clinical and quantitative analysis; and evalutes the inter-rater agreement among the physiotherapists clinically evaluating the joint position estimations.
- Chapter 5 Segmentation Algorithm: This chapter describes the methodology for collecting patient exercise data, including the experimental set-up, justification for the selected exercises and the type of data collected; proposes a segmentation algorithm; describes the motion features; pre-processing steps for the exermplar; feature extraction of the salient points; segmentation of a repetition; and lists the main parameters of the algorithm; evaluates the algorithm against a public dataset; evaluates the execution performance including the time complexity; performs a detailed evaluation on our stroke rehabilitation exercise dataset; and performs a parameter evaluation to show how the algorithm changes with respect to different parameter combinations.
- Chapter 6 Conclusion and Future Work: This chapter reiterates the findings and contributions; and discusses future work, including a clinical pose estimation algorithm in development intended to overcome the limitations highlighted in this thesis.

2 Related work and background

This chapter explores different types of depth-sensors; reviews existing and proposed technology-assisted physical rehabilitation interventions (TAPRI), and discusses the current limitations; explores different data representations/descriptors of human motion; reviews literature analysing the accuracy of state-of-the-art pose estimation algorithms (PEA), with emphasis on papers targeting a clinical evaluation; reviews and discusses algorithms for the segmentation of human motion; and discuss different public datasets of human exercises.

2.1 Related work

This literature review focuses on papers that meet the overall aim of the research, that is a practical TAPRI for stroke patients that can be used in a home setting. It

Table 2.1: Summary of compact and cost-effective depth sensors.

Depth sensor	**Depth resoltuion**	**RGB resolution**	**Has microphone?**	**Type**
Microsoft Kinect v2	512×424	1920×1080	Yes	Time-of-Flight
Microsoft Kinect 1	320×240	640×480	Yes	Structured Light
Occipital Structure	640×480	N/A	No	Structured Light
Asus Xtion PRO	640×480	1280×1024	Yes	Structured Light
iPhone X	1280×720	3840×2160	Yes	Structured Light
Intel D435	1280×720	1920×1080	No	Stereo Vision
ZED Mini	3840×1080	3840×1080	No	Stereo Vision

is assumed exercises are presented to the patient, given a patient's exercise program, therefore the literature review targets approaches to segmentation where the exercise is known. Given that rehabilitation exercises often have repetitive motions, papers on activity segmentation, whereby the activity can be performed in many different ways, is of less relevance but still considered.

2.1.1 Depth sensors

Compact and cost-effective depth-sensors have made an appearance in the last decade, providing a promising avenue for home-based rehabilitation. The depth data from these sensors can be used as input data to a pose estimation algorithm to determine human joint positions or other human landmarks in real-time; for example full body joint tracking [16], hand joint tracking [17], tracking of facial landmarks [18]. Table 2.1 details the specification of popular depth sensors. These devices mainly use three methods for calculating depth; Time-of-Flight, structured light and stereo vision, described in this list:

- Time-of-Flight [19]: Time-of-Flight depth-sensors produce infrared light pulses that are detected by a sensor, by calculating the time that the pulse takes to return to the sensor, the depth of the scene can be determined.

- Structured Light [20]: This method projects an infrared pattern into the scene, this pattern is deformed based on the surfaces it reflects off and a sensor detects the deformed pattern allowing the depth of the scene to be calculated.
- Stereo Vision [21]: By using two RGB cameras with a known distance apart, the pixel data from both images can be evaluated to calculate the depth of a scene.

2.1.2 Technology-assisted physical rehabilitation interventions

Numerous TAPRIs have been proposed in literature and/or developed for healthcare providers [22–33]. This section reviews different technological approaches to delivering a rehabilitation intervention.

Non-wearable interventions

This section reviews TAPRIs that require no wearable sensors to detect human motion such as depth-sensors.

A game-based rehabilitation system was proposed by Chang, et al. [23]. This approach used a Kinect depth sensor to track the patient's joint positions and provided auditory and visual feedback to the patient. The intervention was evaluated on two patients with motor impairments. During rehabilitation a baseline session was conducted whereby the patient would perform their exercises without assistance, then a second session was performed with the proposed intervention. During both sessions the number of correct movements were logged. It was found that the intervention led to a statistically significant increase in the number of correct movements performed during rehabilitation when compared against the baseline.

MindMotion GO [34] provides patients with a virtual environment for performing their exercise program, the system uses a depth sensor to track the patient and provides feedback on compensatory movements. However, there is no public information on the accuracy of the joint position estimations.

Kim, et al. [27] developed and evaluated a Kinect 1 game-based rehabilitation intervention, similar to other approaches the patient would sit in front of a monitor displaying a game based activity to deliver the exercise program. They evaluated the system against a control group whereby each group performed their rehabilitation program for 10 consecutive days for 30 minutes per session. The outcomes were assessed using the Fugl-Meyer Assessment (FMA). However, they found that their Kinect intervention was not more efficacious than the control group.

A balance rehabilitation intervention for people with cerebral palsy was proposed by Jaume-I-Capó, et al. [31]. They reported a significant improvement in patient scores from their initial score, but the study did not have a control group and therefore can not be compared against conventional therapy.

A virtual game that uses Kinect's pose estimation algorithm to detect incorrect body postures for the elderly was proposed by Saenz-De-Urturi, et al. [32]. The game encourages correct posture during use. They report a detection of incorrect posture to 95.20% accuracy. Although they do not mention to what level of accuracy the system detected the incorrect posture.

Another Kinect game based rehabilitation system was proposed by Roy, et al. [33], the system included a website for doctors to view their patients' performances. However, sufficient evaluation of the system and intervention was not performed.

These interventions usually consist of a depth-sensor that observes the patient, with the exercise program delivered via a monitor. The advantages to this approach are that the set-up only requires the patient to be within the range of the sensor, it is affordable and can be used in a home environment. However, these methods for tracking human motion can be susceptible to occlusion, fail to track joints when objects are used for exercises and may struggle to track fine motor skills.

Wearable interventions

This section reviews TAPRIs that use wearable-sensors to detect human motion such as gloves with infrared emittters or Inertial Measurement Units (IMU).

Standen, et al. proposed a game-based rehabilitation intervention that uses a virtual glove that tracks the position of the patient's fingers. This approach can be used to rehabilitate fine and gross hand movements. However it requires the patient to wear a glove and does not include information on other joints that are required to assess compensatory movements.

Panagiotis, et al. [35] and Borghetti, et al. [36] have proposed gloves that can measure the pose of the hand and fingers. Panagiotis's glove includes actuators for applying force to guide a patient's movements and for reinforcement feedback.

A wearable soft sensing garment was proposed and developed by Menguc, et al. [37] for human gait measurement. The garment consists of pressure sensors embedded in the fabric to track leg motion.

With these interventions the patient may struggle to put on the required wearables in a home environment and unless sufficient wearables are placed on relevant areas of the body, an analysis of the human motion may be rudimentary when compared against a depth-sensor PEA approach which detects many joint positions. Although the accuracy of a wearable sensor may be more accurate in tracking human motion than non-wearable interventions.

Robot-assisted interventions

This section reviews TAPRIs that use robots to deliver the intervention such as a robotic arm that is grasped by the patient or a socially-assitive robot.

Chang, et al. [26] reviewed evaluations of upper limb robot-assisted interventions to determine if they improve upper limb motor function. This approach typically involves a robotic arm that is grasped by the patient, a monitor displays an interactive game requiring the patient to perform upper body movements to achieve success. They conclude that the present evidence supports these types of interventions for improving motor function.

Lotfi, et al. [29] proposed a socially assistive robot for delivering an exercise program for the elderly population. The Double robot has a tablet for demonstrating the exercise and for visual feedback to the patient, speakers provide audio feedback for encouragement. Another socially assitive robot was developed by Yinbei, et al. [38], this humanoid robot performs the exercise in front of the patients.

Currently robot-assisted interventions are impractical for home-based rehabilitation given the size and cost and therefore they will likely be used for group-based rehabilitation sessions. Socially assistive robots offer patients an engaging platform for rehabilitation. Where a robotic arm is grasped by the patient, this can provide the patient with force feedback.

2.1.3 Human motion descriptors

This section reviews different data representations/descriptors of human motion. Human motion can be captured from various modalities such as depth-sensors, IMUs and RGB cameras. This review concentrates on descriptors that use joint position estimations for the task of segmentation and/or exercise/action recognition. For these tasks the requirements for a human motion descriptor are typically invariance to scale, rotation, translation and deviation while remaining selective to the exercise or action. The processing time for producing the descriptor from raw joint position data may need to be considered for real-time applications.

Hussein, et al. [39] use a sequence of the covariance of 3D joint positions to represent human motion, the joint coordinates were normalised to a range from 0 to 1 to achieve scale invariance before computing the covariance matrix. The human motion representation was combined with a linear Support Vector Machine (SVM) and evaluated on three public datasets on the task of human action recognition. The approach outperformed state-of-the-art methods in multiple datasets.

Ofli, et al. [40] proposed a representation called Sequence of the Most Informative Joints (SMIJ), this approach ranks the joints by the most informative i.e. the metric with the highest value e.g. variance of the joint. This metric is calculated and ranked for each joint over N time segments. This shows the evolution of each joint, ranked by a metric, over time. This human motion representation was evaluated on the task of human activity recognition. This approach requires a time segment parameter to be set.

Vemulapalli, et al. [41] have proposed an approach that models the 3D spatial relationships between groups of joints within a Lie group. Using this descriptor, actions are modelled as curves. This requires manually choosing sets of joints to represent body parts.

An approach by Wang, et al. [42] uses the pairwise relative position of joints, this produces a set of direction vectors between each joint. A Fourier Temporal Pyramid (FTP) is used to capture the temporal dynamics of each joint. This approach was extended by Hu, et al. [43] to include the FTP of the gradient of the joint direction vectors. This extension encodes the velocity of the groups of joints into the descriptor.

An approach that uses histograms of 3D joint positions was proposed by Xia, et al. [44]. A spherical coordinate system is used with the hip centre as the origin. The 3D joint positions are partitioned into bins. These approaches require manually selecting informative joints.

2.1.4 Evaluation of pose estimation algorithms

A number of studies have been undertaken on Kinect 1 and Kinect 2, though within different scenarios. Generally, these studies used marker based motion capture systems to establish a ground truth. Fernández-Baena et al. [45] examined Primesense's NITE pose estimation algorithm using depth data from Kinect 1. They claimed that joint accuracy could be improved by imposing a fixed length on the bones, and indicated that "Kinect can be a very useful technology in present rehabilitation treatments", though they performed no clinical evaluation.

Obdrzalek et al. [46] examined Kinect 1 for elderly coaching exercises and concluded that measurements "could be used to assess general trends in the movement", though they made no clinical claims. Kurillo et al. [47] found that Kinect 1's pose estimations were sufficiently accurate for reachable workspace analysis.

Xu et al. [48] compared the quality of Kinect 1 and 2 for poses within activities of daily living, and interestingly found that Kinect 1 produces lower errors. In contrast, Wang et al. [49] considered that Kinect 2 was superior over a range of 12 exercises particularly when occlusion and body rotation occurred.

In terms of clinical assessment, Yeung et al. [50] found that Kinect 1 could achieve acceptable accuracy for total body centre of mass movements, but performed better for medial and lateral movements than anterior and posterior movements. In terms of stroke rehabilitation, Webster et al. [51] evaluated the joint accuracy of Kinect 1 on 13 gross movements. They found that Kinect 1 accuracy is sufficient for gross movement-based rehabilitation systems for clinical and in-home use. However, there was no assessment within standard rehabilitation exercises or clinical evaluation for the possibility of detecting compensatory movement.

Mobini et al. [52] evaluated the accuracy of the flexible action and articulated skeleton toolkit [53] using Kinect 1 for upper body stroke rehabilitation applications.

They found that lateral variations in position did not significantly impact joint accuracy, though horizontal distance had some effect.

These previous studies concentrated on absolute joint accuracy as compared with a ground truth provided by motion capture systems. The comparison with a clinical study, where expert clinicians provide analysis on significant aspects of the poses calculated by the equipment, was not performed. During rehabilitation, clinicians stress the importance of ensuring that the patients avoid compensatory movements, and so the evaluation and assessment of the pose algorithms needs to emphasise this aspect.

2.1.5 Inter-rater agreement of clinicians

This section provides a brief overview of literature evaluating the inter-rater agreement among clinicians assessing patient performance.

Barth et al. [54] reviewed studies on inter-rater agreement in evaluation of disability assessed by medical experts. They found that there was an indication of high variation in judgement. They highlighted a need for the development and testing of instruments and structured approaches to improve the reliability in expert evaluation of disability. Our results support this conclusion and highlight a need for more reliable evaluation methodologies for technology intended for clinical use.

Ageberg et al. [55] measured the inter-rater reliability of the clinical assessment of single limb mini squats. They found the medio-lateral motion of the knee can be reliably assessed. However only two examiners were compared and both examiners discussed the scoring of the knee position before deciding whether there was agreement. The examiners did not assess the exercises in isolation. They state that the examiners received explicit guidelines and thorough training prior to study start, likely contributing to the achieved high reliability.

Blackburn et al. [56] evaluated the inter-rater reliability of scores from two physiotherapists using the Modified Ashworth Scale to assess patient muscle tone during stroke rehabilitation. They found the inter-rater reliability of the physiotherapists to be poor. Chmielewski et al. [57] investigated the inter-rater reliability of two methods used for the evaluation of movement quality during rehabilitation. Three clinicians assessed twenty-five healthy subjects performing exercises. They found that agreement was better than chance but neither method used for clinical assessment produced high agreement. They conclude that the results indicate a need to develop more explicit criterion for rating movement deviation severity.

A common theme within the literature is that inter-rater agreement is often low and that there is a need for a more objective criterion to support raters, including the use of instruments for testing.

2.1.6 Segmentation of human motion

This section introduces published research relating to the temporal segmentation of actions and exercises.

Gong, et al. [58] have proposed an approach that segments multivariate time series data of temporal joint positions (or angles). The algorithm was trained and tested on a subset of data of 15 people performing 10 actions once. However there was no evaluation using clinical data of patients with physical disability that commonly show low range of motion and instability.

Kohlmorgen, et al [59] proposed a Hidden Markov Model (HMM) approach that calculates multiple probability density functions (PDF), which act as the segmentation features, over a moving window over the observation data. These PDFs are used to train the model, generating the state transitions from the PDFs which act as the segment points. Lin and Kulic [60] proposed using zero-velocity crossings

(ZVC) as segment candidates with a HMM based template matching method to segment points, these HMM approaches require multiple samples for training.

A multimodal approach was proposed by Wu, et al. [61] that segments gestures from joint positions, depth and RGB data. This approach requires training data for different modalities. Chaun-Jun, et al [62] have proposed a Dynamic Time Warping (DTW) approach for segmenting rehabilitation exercises. The system was tested on three shoulder exercises performed by four healthy people. They use DTW to align the joint data but no dimensionality reduction methods are used.

De Souza Vicente, et al. [63] have proposed using latent-dynamic conditional random fields. A filtering technique based on key poses is used to reduce the number of frames prior to segmentation. They tested their algorithm on Taekwondo moves which are relatively fast compared to general actions. This approach was tested on data from athletes where the authors state that there is little variation of each Taekwondo move and therefore the approach may be insufficient for segmenting exercises from subjects with impaired mobility.

Krüger, et al. [64] proposed a kd-tree-based nearest-neighbor-search that is said to be a fast alternative to subsequence DTW alignment. This approach was used by Baumann, et al. [65] and extended to enable its use for action recognition. This approach was not tested on clinical subjects with limited mobility.

Krüger, et al. [66] have proposed an unsupervised approach that identifies action repetitions and further decomposes the actions into atomic motion primitives. However, the segmentation algorithm was tested on non-clinical data from "fairly constrained settings".

Wang, et al. [67] have proposed an unsupervised approach to segmentation. Joint trajectories are converted to a kinematic model using an unscented Kalman filter and the most representative kinematic parameters for the segmentation of an action repetition are selected. ZVC are detected to produce a list of segmentation candidates. Finally, k-means clustering is used to determine the boundaries of each repetition. The algorithm was tested on joint data from healthy subjects performing non-clinical actions.

Lin, et al. [68] have proposed a two-class classifier to classify each data point as either a segment or non-segment point. Dimensionality reduction is performed prior to data point classification. However, the classification stage requires training data. The top performing classifiers used Principal Component Analysis (PCA) prior to classification. Support Vector Machine (SVM), Artificial Neural Network (ANN) and k-Nearest Neighbour (k-NN) provided the highest accuracy in segmentation. They mention high processing costs of k-NN making it unsuitable for real-time applications.

Devanne, et al. [69] have proposed a segmentation algorithm that jointly analyses the shape of the human pose and motion in a Riemannian manifold. The approach was tested on temporal skeleton data of healthy participants performing actions.

Shan, et al. [70] identify key poses by analysing the joint data for minimal changes in kinetic energy, a parameter that must be tuned. Then atomic action templates (AAT) are produced from the key poses and temporal midway points. Multiple AATs can form an action template. Finally, a classifier is used to classify the AATs and determine a label for the action. Four classification models were tested; HMM, k-NN, SVM and Random Forest. The classifiers obtain similar recognition results suggesting the feature representations of an action are sufficiently discriminative, although this was tested on healthy subjects performing non-clinical actions. This approach also requires training data.

Many of these approaches are limited in that they either require multiple samples for training and/or were evaluated using healthy subjects, or are too computationally expensive to run in real-time.

2.1.7 Public exercise datasets

This section describes different human motion datasets consisting of joint positions of subjects performing activities, actions and exercises captured from marker-based motion capture systems or depth-sensors.

A stroke rehabilitation dataset (TRSP) was published by Dolatabadi, et al. [71]. This contains two stroke rehabilitation exercises (Reach Side-to-Side and Reach Forward-Backward) from 9 stroke survivors and 10 healthy subjects captured by a Kinect depth-sensor. Another recently published rehabilitation dataset, UI-PRMD [72], contains physical rehabilitation exercises from 10 healthy subjects captured from a marker-based motion capture system.

Carnegie Mellon University Motion Capture Database (CMU) [73] consists of various activities, actions and exercises performed by 144 different subjects. HDM05 dataset [74] contains about 70 different motion classes of exercises and actions such as squats, sitting, walking. Berkeley's MHAD dataset [75] consists of 11 actions/exercises such as jumping jacks, waving and punching. The actions were performed by 7 males and 5 female subjects. These datasets were captured by marker-based motion capture systems.

The Microsoft Research (MSR) Cambridge-12 dataset(MSRC-12) [76] contains 12 actions performed by 30 subjects. The dataset include 6,244 gesture instances with 20 joints tracked. The MSR Daily Activity [77] dataset consists of activities such as drink, eat and use laptop performed by 10 subjects. The MSR Action3D dataset [78] contains 20 actions from 10 subjects. The CAD-60 and CAD-120 activity datasets [79] contain 12 and 10 activities respectively from 4 subjects. The UTK Action dataset [80] contain 10 actions such as walk, throw, wave from 10 subjects. The UCFK Dataset [81] contains 16 actions such as kick, duck, run, which were performed by 13 males and 3 females each 5 times. These datasets were captured from a Kinect depth-sensor.

Except for [71], which was published after we had collected our own data and only contain two exercsies, the publicly available datasets lacked real-world patient data of rehabilitation exercises captured on a compact depth-sensor, for this reason we collected our own stroke rehabilitation dataset described in Chapter 5.

2.2 Background

This section reviews existing clinical assessments of stroke and describes deterministic and probabilistic/stochastic techniques for human action/exercise segmentation.

2.2.1 Clinical assessments of stroke

The Fugl-Meyer Assessment (FMA) [82] is a common assessment criteria specifically designed for evaluating stroke survivors. FMA assesses motor function, balance, sensation and joint function. Each dimension of the assessment is scored with points and the FMA total motor score provides a classification of impairment, ranging from severe to slight. The assessment is carried out by a physiotherapist observing the patient.

The Chedoke-McMaster [83] stroke assessment consists of a two-part measure using a physical impairment inventory and disability inventory. This contains a six dimensions each measure on a 7-point scale relating to stages of motor recovery. Each dimension measures shoulder pain, postural control, arm, hand, leg and foot movement.

The Stroke Rehabilitation Assessment of Movement (STREAM) [84] consists of 30 test movements which fall into three dimensions of upper-limb, lower-limb and basic mobility movements. 3 and 4 point scales are used to measure each dimension. The assessment is scored by direct observation from a physiotherapist.

Two upper-arm assessments, Action Research Arm Test (ARAT) [85] and the Box and Block Test (BBT) [86], provide a measure of arm mobility. ARAT measures a larger variety of movements whereas BBT measures the number of blocks that are moved between two compartments within 1 minute.

All these assessments have a degree of subjectivity due to measurements being performed by a direct observation from physiotherapists and are not designed with quantifiable performance measures carried out by technology.

2.2.2 Approaches to segmentation of human motion

Dynamic time warping

The Dynamic Time Warping (DTW) algorithm [87] is a template matching technique that aligns two signals that minimises some distance metric. This approach normalises the alignment in the time dimension, providing a time invariant alignment. This is especially useful in action segmentation as repetitions can be performed at different speeds. DTW has been applied to a variety of applications: aligning gene expression data [88–90]; voice recognition [91–93]; recognising hand written words [94–96]; audio matching for music retrieval [97]; classifying disturbance in electric power systems [98]; protein fold recognition [99]; and action recognition and segmentation [62, 100, 101].

With classical DTW [87] the problem formulation is as follows, given two time-series signals, a query sequence Q and a candidate sequence C, find the an alignment called the warping path W that minimises the total distance.

A distance measure needs to be defined, the type of distance measure to use depends on the type of data. It is important a suitable distance measure is used for a

Table 2.2: Definitions of common distance measures. Where k is an index of W of length K, W_k contains a pair of indices (i, j) indexing an element in Q and C respectively. $Dist(W)$ is the total distance/cost of the warping path W. p is a the order parameter for Minkowski distance, where $p \geq 1$.

Measure	Equation
Euclidean	$Dist(W) = \sqrt{\sum_{k=1}^{K}(W_{ki} - W_{kj})^2}$
Manhattan	$Dist(W) = \sum_{k=1}^{K} \mid W_{ki} - W_{kj} \mid$
Minkowski	$Dist(W) = (\sum_{k=1}^{K} \mid W_{ki} - W_{kj} \mid^p)^{1/p}$
Chebyshev	$Dist(W) = \max(\mid W_{ki} - W_{kj} \mid)$

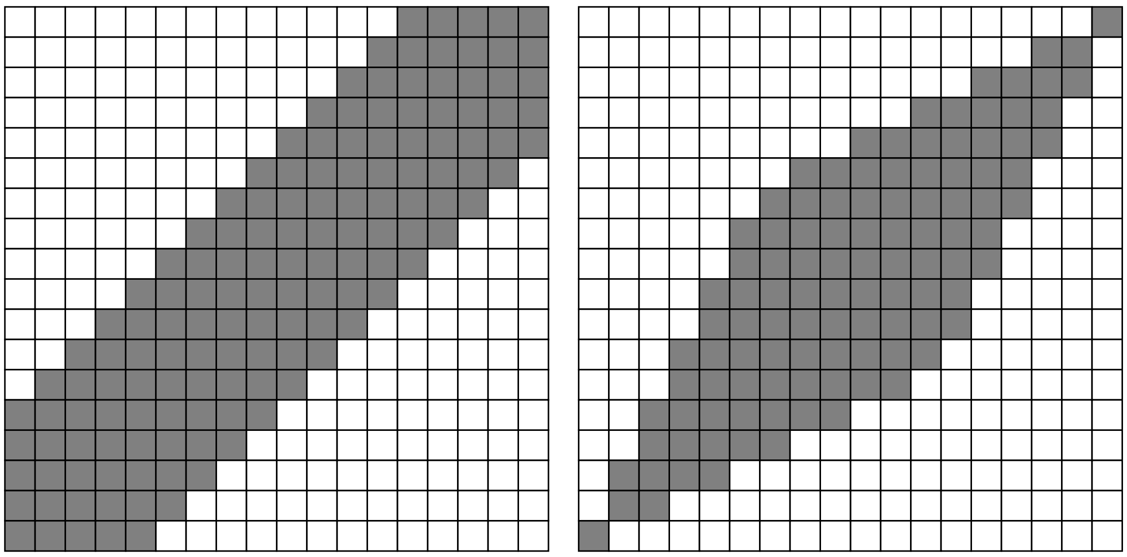


Figure 2.1: Grey squares indicate pairs of indices of the sequences that can exist in the warping path, white squares are not evaluated. Sakoe-Chiba window (left) and Itakura Parallelogram window (right).

given problem as DTW finds the optimal warp path that minimises this distance measure. Here we describe several common distance measures and present the equations in Table 2.2. The Euclidean distance calculates the distance between two points as a straight line. Manhattan/taxicab distance [102] calculates the absolute difference between two values. Minkowski distance is a generalisation of Euclidean and Manhattan distance in a normed vector space, e.g. a p-norm of 1 equals the Manhattan norm and 2 equals the Euclidean norm. Chebyshev distance is the maximum value between two vector spaces in any given dimension/coordinate.

The original DTW implementation came with conditions that the warping path must meet to be eligible, such as the boundary condition whereby the first pair in the warping path must be first elements in Q and C, likewise the last pair in the warping path must be the last elements. The monotonic condition ensures the warping path cannot go back on itself. The step size condition ensures no indices in Q and C are skipped in the warp path. Window constraints can also be used to limit the number of possible warping paths and improve performance, but it should be noted that in doing so finding the optimal warping path is not guaranteed. Two common window constraints are Sakoe-Chiba and Itakura Parallelogram, shown in Figure 2.1.

A variant to classical DTW is subsequence DTW where a query sequence is matched to a subsequence within the candidate. This requires several subsequences of the candidate to be evaluated. This approach is especially useful for action segmentation from a continuous input stream, as the candidate action sequence will likely contain human motion outside of the segmentation envelope of the action.

Hidden markov models

A Hidden Markov Model (HMM) aims to model a system that is a Markov process with hidden states. The process learns state transition and emission probabilities that best move between states to map a given set of observations to a desired output.

In terms of human motion segmentation, various HMM approaches have been used to model segmentation points [70, 103, 104]. On-line HMM variants exist that processes only the latest data points given the previous state.

This approach consists of a training stage whereby the exemplar exercise repetitions are used to create HMM motion templates. A general HMM consists of a sequence of unobservable states. A state transition probability matrix is used to determine the probability a state will transition to another state. An observation probability matrix represents, for example in human segmentation, human motion from the observation, usually consisting of informative motion such as zero velocity crossings or other key poses. An initial state distribution representing the probability of an observation beginning in a state, for left-right HMMs (described in this section) it can be assumed the initial observation begins in the first state.

The number of states differ depending on the exercise, but there must be enough states to accurately model the motion, as too few states may lead to under fitting. If too many states are used the model may over fit and struggle to segment similar motions. The more states there are the higher the computational costs.

Within fully connected HMMs, the states can transition to any other state, while with left-right HMMs, the state transition can only transition to its current state or advance to the next state. The latter approach is usually used for segmentation algorithms as it is expected that the motion states of the query template are sequential.

This approach requires tuning the transition probabilities to ensure examples that deviate from a ground truth example are correctly segmented.

Support vector machines

Support Vector Machines (SVM) attempt to find the maximum-margin hyperplane that maximally separates a set of data points into their classes, this hyperplane acts as the decision boundary between two classes. Kernels can be used to map the data points to a subspace where a suitable linear maximum-margin hyperplane can be found which can result in a non-linear decision boundary when mapped back to original space.

SVM have been used in human action segmentation to classify segment and non-segment points [105]. An SVM is a supervised learning model that trains a two-class classifier. A full explanation of SVM is presented [106].

An SVM can be supplied with a kernel function for measuring the distance between all data points. The standard SVM implementation uses a linear function that attempts to find the linear separation, called the hyperplane, that maximises the margin between the two classes. Non-linear kernels map the data points to a high dimensional space where a suitable linear hyperplane may be found, this decision boundary becomes non-linear when mapped back to the original feature space.

K-nearest-neighbour-search

A k-d tree nearest neighbour search was proposed by [64] for similarity searches, this approach can be used for the segmentation of human actions. Like DTW this approach uses a distance metric for the local distance measure between two sequences.

Here we describe the four main stages of Krüger's nearest-neighbour-search approach [64] for the segmentation of human motion. During the pre-processing stage, a k-d tree [107] is built from a dataset of motion features. Other types of trees can be used such as R-trees, as shown by Keogh, et al [108]. The following stage is performed for each query sequence. A search stage finds the nearest neighbours, defined by the distance measure, using k-nearest-neighbour-search resulting in sets of similar poses. A parameter must be defined to limit the number of neighbours evaluated. A graph is constructed given the sets of poses following similar rules for traversing DTW cost matrices whereby steps between neighbours can advance by at most one step, the resulting graph is a directed acyclic graph. Finally path search can be performed by following the nodes with the shortest path, giving a global accumulated distance between the exemplar repetition and the observation data. This approach finds the optimal alignment, if all the ground truth frame alignments are in the neighbourhood of the query motion.

3 Research proposal

From the review of literature on proposed TAPRIs, proposed systems usually have a component for automatic assessment and feedback of exercises. Such components could measure the patient's exercise performance such as range of motion, stability in movements and compensatory movements. Proposed components and the underlying algorithms that enable the assessment of exercises need to be clinically evaluated to determine the accuracy of the assessment and the limitations. Evaluating the state-of-the-art algorithms that underlie these proposed components and proposing improvements form the basis of the research and contributions in this thesis. This chapter proposes the research undertaken in this thesis and details the rationale for why this work is needed.

3.1 Clinical evaluation of pose estimation algorithm

Based on the literature review, all the proposed TAPRIs use out-of-the-box algorithms, such as Kinect's pose estimation algorithm. Therefore, the first contribution of this thesis is a clinical evaluation of Kinect V2's state-of-the-art pose estimation algorithm that has been used by several proposed TAPRIs such as [34], [27], [31], [32] and [33]. Current published work on the evaluation of PEAs are either not targeted towards clinical applications, have not been evaluated by clinicians or lack a rigorous evaluation on the accuracy of the individual joint positions of stroke rehabilitation exercises, including repetitions performed with common compensatory movements. The overall findings from this research found Kinect's PEA to be insufficiently accurate for correctly assessing most of the exercises and compensatory movements which differs from the findings in several research papers. This research has been published in [1].

Furthermore, we evaluated the inter-rater agreement between the physiotherapists' evaluations to determine the level of agreement on the accuracy of a new technology for clinical use. Although there has been research into the inter-rater agreement of clinicians performing a clinical assessment, to the best of our knowledge there has not been an evaluation on the inter-rater agreement of clinicians evaluating whether a new clinical technology is accurate enough for clinical use. This research has been published in [3].

3.2 Segmentation algorithm

Accurate joint position estimations are required for accurately assessing patient exercises, but the segmentation of exercise repetitions, i.e. finding the start and end of a repetition, is also required to ensure each repetition is assessed correctly. Many proposed segmentation algorithms have not been designed or evaluated on stroke patient exercise data, which is important for ensuring the approach is robust to subjects with impaired mobility. As it is intended that TAPRIs present the patient with an exercise to perform, the segmentation algorithm can exploit this knowledge by evaluating against a single exemplar repetition. For this reason approaches that combine action segmentation and recognition, which are usually machine learning approaches with large training datasets, are not necessarily required.

Thus, the final contribution of this thesis is a segmentation algorithm that requires only a single exemplar repetition from a healthy subject to subsequently segment repetitions from other subjects including those with impaired mobility. This approach uses Dynamic Time Warping (DTW) to measure the distance between an exemplar repetition and a patient's movements, which achieves accurate segmentation results when combined with a robust human motion descriptor. DTW has been used in segmentation algorithms before but there are many different modifications to DTW and DTW forms one component of our proposed approach. Many proposed DTW segmentation approaches also lack evaluation against stroke subjects. The methodology and results of the algorithm are described in chapter 5. This research has been published in [2].

4 Clinical evaluation of kinect's pose estimation algorithm

This chapter describes the methodology for clinically and quantitatively evaluating Kinect V2's pose estimation algorithm (PEA) and presents the results. Specifically it describes: the experimental set-up of the evaluations; the methodology and results of the clinical evaluation; the methodology and results of the quantitative evaluation; the methodology and results of the inter-rater agreement between the clinicians evaluating Kinect's PEA; and finally summarises the findings.

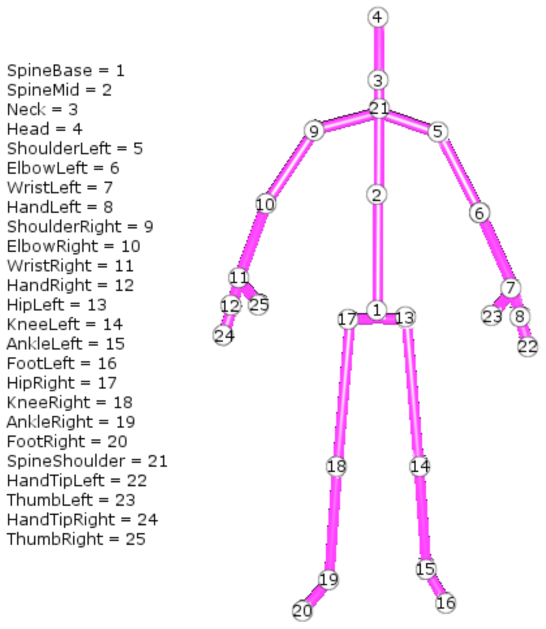


Figure 4.1: The human joint positions tracked by Kinect V2's pose estimation algorithm with the joint labels annotated.

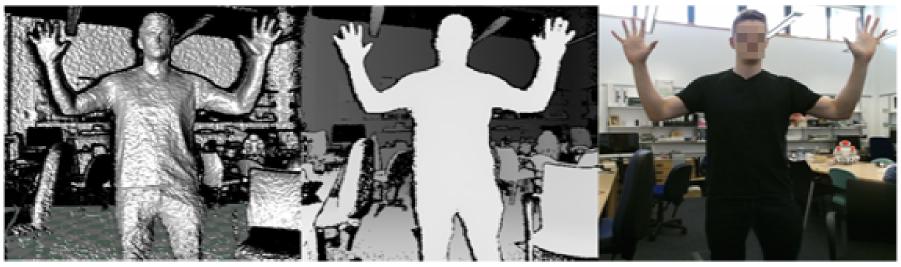


Figure 4.2: A 3D representation of the depth data (Left). A 2D greyscale representation of the depth data (Centre). The RGB colour image (Right). These images are all captured from a single time step from the Xbox One Kinect.

4.1 Experimental set-up

Our evaluation is based upon version 2.0.1410.19000 of the pose estimation algorithm of the Kinect for Windows SDK for the Kinect V2 (Xbox One Kinect). This provides pose estimations for 25 joints at 30Hz, the joint positions that are tracked are shown in Figure 4.1. Joint locations were recorded while seated. Kinect produces a depth image with a resolution of 512 × 424 pixels [109], as shown in Figure 4.2. This depth image is then used as input to the Kinect Software Development Kit's (SDK) pose estimation algorithm, which is based on the approach presented by Shotton

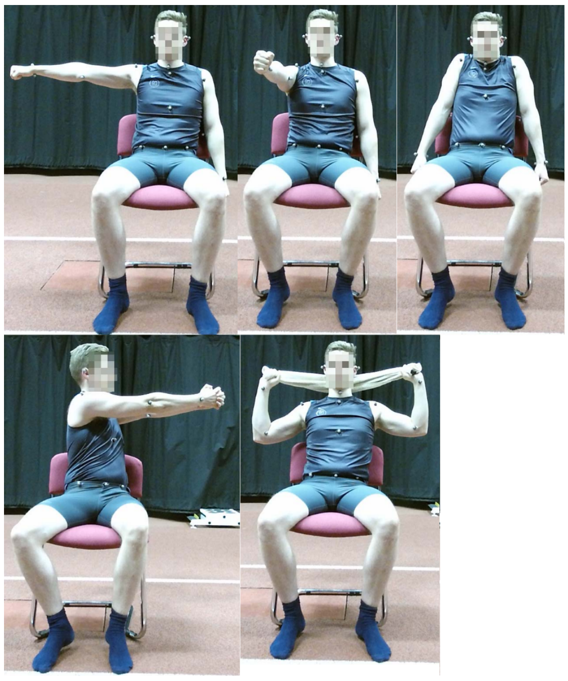


Figure 4.3: The five exercises selected from the GRASP manual [4] for evaluation. Arm to Side (Top Left): Shoulder joint is abducted to 90 degrees and then adducted back to 0 degrees. Arm to Front (Top Middle): Shoulder joint is flexed to 90 degrees and then extended back to 0 degrees. Shoulder Shrug (Top Right): Shoulder joints are elevated and then depressed. Twist (Bottom Left): Shoulders are flexed to 90 degrees and hands are clasped, thorax is rotated towards 90 degrees in one direction then returned to starting position and rotated towards 90 degrees in the opposite direction. Drying off (Bottom Right): Towel is grasped and placed behind the neck, arm extension and flexion is performed along the frontal plane.

et al. [15] to infer the joint positions. The PEA maps a depth image consiting of 217088 pixel values per frame to a set of 25 joint positions consiting of only 75 position values per frame. Given that Kinect's pose estimation algorithm runs in under 5ms on an Xbox 360 graphical processing unit (GPU) [15], further in-depth analysis of human motion can more efficiently take place on this reduced feature space. It should be noted that when Kinect is tracking a body, joints are classified as either tracked or inferred. A joint is classed as tracked when confidence in the data is high i.e. there is little or no occlusion of the point cloud data surrounding the joint. If there is full or significant occlusion of the point cloud data surrounding the joint, its coordinates are classed as inferred.

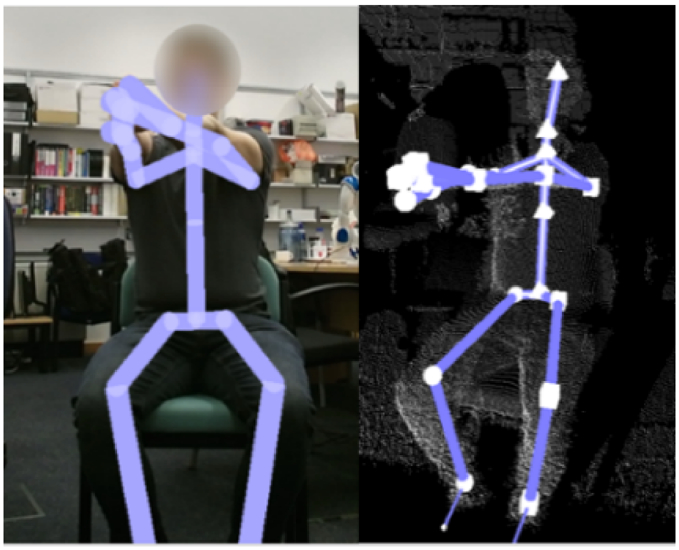


Figure 4.4: The video and depth data available to the physiotherapists when performing an evaluation of the estimated joint positions.

Table 4.1: List of exercises the physiotherapists observed including the associated common compensatory movements.

Exercise	**Associated Common Compensatory Movements**
Arm to side	Trunk lateral flexion, Shoulder elevation, Thorax rotation, Arm flexion
Arm to front	Trunk backward flexion, Shoulder elevation, Thorax rotation, Arm flexion
Shoulder shrug	Head side flexion, Shoulder abduction
Twist	Trunk lateral flexion, Arm flexion
Drying off	Trunk lateral flexion, Shoulder elevation, Dipped arm, Head side flexion

4.2 Clinical/qualitative analysis

4.2.1 Qualitative analysis methodology

The following gross upper-body exercises selected for analysis were taken from GRASP [4], a stroke rehabilitation exercise manual: Arm to Side, Arm to Front, Shoulder Shrug, Twist and Drying Off (Figure 4.3). They range from relatively simple exercises, e.g. Arm to Side, to more difficult exercises, e.g. Drying Off, which requires motion from multiple limbs and a towel to be grasped. All exercises were recorded from a frontal view as this is the expected view for observing patients. As we are investigating the pose estimation accuracy in the context of upper-body stroke rehabilitation applications, pose positions below the hips and on the hands were

Table 4.2: Evaluation criterion used by the physiotherapists for the evaluation of the joint position estimations in Table 4.3

Evaluation criterion	
Acceptable Tracking (AT)	A joint's estimated positions' result in an acceptable difference from the true position. The error in the position does not lead to misclassification in the assessment, e.g. a limb is showing no flexion when no flexion is occurring.
Moderately Acceptable Tracking (MT)	A joint's estimated positions' result in a moderately acceptable difference from the true position. The error in the position leads to a minor misclassification in the assessment, e.g. a limb is showing minor flexion when no flexion is occurring.
Unacceptable Tracking (UT)	A joint's estimated positions' result in an unacceptable difference from the true position. This change in position leads to a significant misclassification in the assessment, e.g. a limb is showing severe flexion when no flexion is occurring.

not considered. To perform a clinical evaluation of the pose estimations, recordings of a participant performing the GRASP exercises were captured in Kinect Studio. The physiotherapists viewed video and depth recordings of the exercises with the estimated joint positions overlaid, as shown in Figure 4.4. Each exercise was first performed correctly, and then repeated with each of the common compensatory movements, as listed in Table 4.1. For example, the Drying Off exercise was performed 5 times; correctly and then 4 separate versions with each compensatory movement. The physiotherapists made observations and were asked to give their expert opinion on the accuracy of the pose estimations using the evaluation criterion in Table 4.2, noting the accuracy of the joint position estimations for assessing the performance of the participant. This evaluation criterion differs from clinical assessment criteria as the evaluation is performed on the accuracy of a technology for clinical use rather than the evaluation of a stroke patient's performance. To guide the rater, the categories of tracking accuracy are delineated by the severity of misclassification of compensatory movements.

Table 4.3: Physiotherapists' evaluation of Kinect's pose estimation for each GRASP exercise. See definitions of AT, MT and UT in Table 4.2.

Description	***P1***	***P2***	***P3***	***P4***	***M***	***SD***	***Comments***
ElbowRight joint 'Arm to Side' All versions	MT	AT	AT	AT	AT	0.43	P1: Jitter occurs along the axis of the bone, resulting in variable limb lengths.
WristRight joint 'Arm to Side' All versions	MT	AT	AT	AT	AT	0.43	P1: Jitter occurs along the axis of the bone, resulting in variable limb lengths.
SpineMid joint 'Arm to Side' Trunk lateral flexion	AT	MT	AT	AT	AT	0.43	P2: Angle around SpineMid joint is represented as a straight line when trunk flexion is occurring.
Hip joints 'Arm to Side' Trunk lateral flexion	UT	AT	AT	MT	MT	0.83	P1: Hip joints give the impression that one hip is being lifted from the seat. P4: Hip joints showing exaggerated movements than is true.
Shoulder joints 'Arm to Side' Shoulder elevation	UT	MT	AT	MT	MT	0.71	P1: Roughly a quarter of the vertical movement is reported in the joint. P2: Shoulder position not accurately portraying severity of shoulder elevation. P4: Shoulder elevation is visible but not to the extent that is true.
ShoulderRight joint 'Arm to Front' All versions except trunk backward flexion and shoulder elevation	MT	AT	AT	AT	AT	0.43	P1: As the arm reaches 90 degrees the shoulder joint drops to the axilla this gives the impression the arm is at a higher angle than is true.
ElbowRight joint 'Arm to Front' All versions except elbow flexion	MT	AT	UT	MT	MT	0.71	P1: Jitter occurs when joint is occluded resulting in elbow flexion when the arm is straight. Joint also reports different limb lengths. P3: Incorrectly displaying elbow flexion when the arm is raised. P4: Jitter can cause confusion with knowing whether the patient kept their arm straight during the exercise.

WristRight joint ‘Arm to Front’ All versions	MT	AT	UT	MT	MT	0.71	P1: Jitter occurs when joint is occluded resulting in elbow flexion when the arm is straight. Joint also reports different limb lengths. P3: Incorrectly displaying elbow flexion when the arm is raised. Incorrectly showing flexion extension in the wrist. P4: Jitter can cause confusion with knowing whether the patient kept their arm straight during the exercise.
Shoulder joints ‘Arm to Front’ Trunk backward flexion	UT	UT	UT	UT	UT	0	P1: The shoulders track inwards severely when this is not the case, thus falsely reporting elbow flexion. P3: Visually looks like the shoulder joints move towards the torso as trunk backward flexion occurs.
SpineMid joint ‘Arm to Front’ Trunk backward flexion	AT	MT	AT	AT	AT	0.43	P2: Angle around SpineMid joint is represented as a straight line when trunk flexion is occurring.
Shoulder joints ‘Arm to Front’ Shoulder elevation	UT	UT	UT	UT	UT	0	P1: UT occurs when the arms occlude the shoulder. P2: Shoulder dips down as the arms occlude the shoulder. P3: Initially elevates but when the shoulder is occluded by the arm, the shoulder joint depresses. P4: Not clear shoulder elevation is occurring.
Elbow joints ‘Arm to Front’ Elbow flexion	UT	AT	AT	MT	MT	0.83	P1: When the elbow is flexed, jitter occurs even when the joint is not occluded. P4: Jitter can cause confusion with knowing whether the patient kept their arm straight during the exercise.

Shoulder joints 'Shoulder Shrug' All versions	UT	UT	MT	MT	MT/UT	0.5	P1: Only a minor vertical movement when elevating the shoulders. P2: Minor shoulder elevation tracked when significant shoulder elevation occurring. P3: Not showing elevation to the degree the shoulders are. P4: Can see some elevation but not showing the range.
Wrist joints 'Shoulder Shrug' All versions	MT	AT	AT	AT	AT	0.43	P1: Jitter occurs when joint becomes occluded by the legs.
Hips and SpineBase joints 'Shoulder Shrug' All versions	AT	AT	AT	MT	AT	0.43	P1: Acceptable jitter can be seen, they also slightly elevate as the shoulders are lifted even though the true joint positions remain still. P4: Hips elevate with the shoulders.
Neck joint 'Shoulder Shrug' Head Flexion	MT	MT	UT	AT	MT	0.71	P1: Reporting only minor head lateral flexion when severe. P2: Not correctly showing the severity of head flexion. P3: Not able to interpret the head and neck markers as flexion.
Head joint 'Shoulder Shrug' Head Flexion	MT	MT	UT	MT	MT	0.43	P1: When severe head lateral flexion occurs, the joint has MT, resulting in reporting a minor head lateral flexion. P2: Not correctly showing the severity of head flexion. P3: Not able to interpret the head and neck markers as flexion.
SpineMid joint 'Shoulder Shrug' Shoulder abduction	MT	MT	UT	MT	MT	0.43	P1: Falsely reporting minor trunk lateral flexion, when no trunk lateral flexion occurring.
Shoulder joints 'Twist' All versions	UT	AT	MT	UT	MT	0.83	P1: Joints track around the axilla as the arms are raised to 90 degrees. P4: When arms raised shoulders become depressed down to the rib cage.

Elbow joints ‘Twist’ All versions	MT	AT	MT	AT	AT/MT	0.5	P1: Joint showing jitter and variable limb lengths during the exercise.
Wrist joints ‘Twist’ All versions	MT	AT	UT	AT	MT	0.83	P1: Joint showing jitter and variable limb lengths during the exercise.
SpineShoulder, Head and Neck joints ‘Twist’ All versions	UT	AT	MT	UT	MT	0.83	P1: Joints incorrectly track vertically as the joints are occluded by the arms. P3: Rotation is inferred by arm joint positions. P4: Joints elevated when occlusion occurs.
SpineMid joint ‘Twist’ Trunk lateral flexion	AT	UT	AT	AT	AT/MT	0.87	P2: Angle around SpineMid joint is represented as a straight line when trunk flexion is occurring.
Elbow joint ‘Twist’ Trunk lateral flexion	UT	MT	MT	MT	MT	0.43	P1:Unacceptable jitter. P2: Jitter occurring. P4: Shows more flexion than is occurring during some of the exercise.
Wrist joint ‘Twist’ Trunk lateral flexion	UT	MT	MT	AT	MT	0.71	P1: Unacceptable jitter. P2: Jitter occurring.
Shoulder joints ‘Drying Off’ All versions	UT	UT	MT	UT	UT	0.43	P1: Unacceptable jitter and tracking on the towel. P2: Joint incorrectly tracks on towel. P3: Joint positions briefly glitch onto the towel. P4: Unacceptable because the joint occasionally tracks on the towel.

Elbow joints ‘Drying Off’ All versions	UT	UT	MT	MT	MT/UT	0.5	P1: Unacceptable jitter and tracking on the towel. P2: Joint incorrectly tracks on towel. P3: Joint positions briefly glitch onto the towel. P4: Unacceptable because the joint occasionally tracks on the towel.
Wrist joints ‘Drying Off’ All versions	UT	AT	AT	MT	MT	0.83	P1: Unacceptable jitter and tracking on the towel.
Hip and SpineBase joints ‘Drying Off’ All versions	UT	AT	MT	UT	MT	0.83	P1: The hips and SpineBase joints show UT in the vertical axis. P4: Joints move around during the exercise.
SpineMid joint ‘Drying Off’ Trunk lateral flexion	UT	UT	UT	UT	UT	0	P2: Angle around SpineMid joint is represented as a straight line when trunk flexion is occurring. P4: SpineMid does not move. Not showing any side flexion.

4.2.2 Qualitative analysis results

Four practising physiotherapists analysed the accuracy of the joint position estimations for each GRASP exercise. Their assessments are presented in Table 4.3. For video samples from the clinical evaluation sessions see [110]. Physiotherapists were free to comment on any joint, using their clinical judgement to decide what was worthy of comment. Because of time constraints, only P1 watched all of the variants for all of the five exercises, which consisted of a total of 16 repetitions containing correct movements and compensatory movements. Where P1 made no comment on a repetition, they confirmed that it was because they considered the pose tracking to be acceptable for assessing the exercise with regards to the compensatory movements. However they did explicitly comment on some of the repetitions that were wholly acceptably tracked, and these are shown in the table. The repetitions where P1 had made some comment about the tracking quality, these were presented individually to the other physiotherapists (without any indication of each other's views) to determine whether they also considered the tracking to have some problems. To calculate the mean (M) and standard deviation (SD), the categories AT, MT and UT from Table 4.2 were given a value of 1, 2 and 3 respectively.

As can be seen from Table 4.3, each of the exercises resulted in some undesirable aspects in the tracking. Even the more straightforward exercises such as Arm to Side, which would have little or no occlusion, caused some issues. Problems occurred with jitter at the elbow and wrist joints, which could give rise to variable bone lengths. Fernández-Baena et al. [45] commented that fixed bone lengths might improve the joint accuracy. Trunk flexion caused problems throughout, partly due to occlusion. It was noted by several physiotherapists that to perform a correct analysis of the exercise, joint rotational information is required. This was noted when assessing trunk flexion during the twist exercise. Of more clinical interest

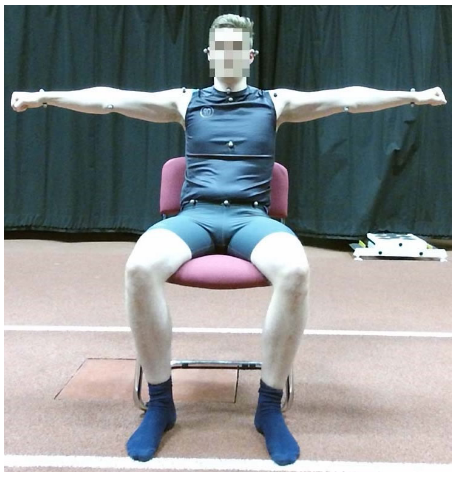


Figure 4.5: Seated T-Pose posture used as Kinect's ground truth for determining the SD and mean error of joint positions over an exercise.

is the variation between the opinions of the physiotherapists. This may partly be due to familiarity, as P1 spent much longer analysing the results. The highest variation came from the SpineMid joint for the twist exercise while trunk lateral flexion occurred, where P2 rated the joint unacceptable, noting that the spine was not showing flexion, while the others rated it acceptable. The mean categorisation shows that exercises with objects or substantial occlusion leads to unacceptable or moderately acceptable tracking and therefore can be difficult to correctly assess. On occasions joint position estimations would result in an anatomically impossible pose, for example shoulder joints tracking inwards towards the spine as trunk backwards flexion occurs.

4.3 Quantitative analysis

4.3.1 Quantitative analysis methodology

For the quantitative assessment [1], the ground truth (GT) was provided by a passive retro-reflective marker based motion capture system [111]. This system captured

[1]See appendices for code snippets of the quantitative analysis program.

the exercises at 240Hz simultaneously with Kinect. The participants sat in an armless chair ≈2 metres from the Kinect. Previous work has indicated that at this distance, Kinect has an average depth accuracy error of less than 2mm [112]. 14mm passive reflective markers were placed on the centre of the anatomical joints to be tracked. Where Kinect's counterpart anatomical joint is unclear, e.g. SpineMid, the markers were placed over the top of the Kinect joint while the user was in a seated T-pose posture, as shown in Figure 4.5. Multiple markers were used on certain joints to determine the location of the centre of the joint. For example, two markers were placed on the front and back of each shoulder and the position between the two markers were calculated to get the joint centre. The Kabsch algorithm [113] was used for rotational alignment of the datasets in the X and Y axes. This approach minimises the root mean square deviation between the HipLeft, HipRight and SpineShoulder joints for both datasets at the T-pose posture frames. As the markers are visually placed on top or near their counterpart Kinect joints, rotational alignment is accurate for the X and Y positions. To find a good rotational alignment for the datasets in the Z position, a reference frame was defined when the user's arms were by their side, and the Qualisys dataset rotated around the X axis by 0.5 degrees to find the minimum difference in WristLeft Z position between the Kinect dataset and Qualisys dataset on the reference frame and T-pose frame. After the alignment of the two datasets, to accurately calculate the standard deviation (SD) and mean error of Kinect's joint positions, a seated T-pose posture was selected as the GT frame of Kinect's joints (see Figure 4.5); this posture presents Kinect's pose estimation with little difficulty. The joint SDs were modelled as ellipsoids to enable visualisation of the variance/jitter of each joint in all axes [49]. The exercises were performed by 5 volunteers, and the calculated results averaged.

Table 4.4: Table showing the error and SD for each joint position estimation averaged over all repetitions. The right arm was used for the arm to side and arm to front exercises.

Joint	**Arm to Side**		**Arm to Front**		**Shoulder**		**Twist**		**Drying Off**	
	Error	**SD**	**Error**	**SD**	**Error**	**SD**	**Error**	**SD**	**Error**	**SD**
Spine base	1.14	0.34	1.18	0.34	2.42	1.19	6.80	3.51	3.09	0.98
Spine mid	1.13	0.33	1.78	0.37	2.73	1.00	8.39	5.15	4.62	1.60
Neck	0.77	0.24	1.12	0.27	1.27	0.45	6.53	3.75	2.78	1.17
Head	0.61	0.15	0.57	0.21	1.36	0.66	4.80	2.55	3.71	1.72
Shoulder left	1.47	0.23	1.56	0.38	2.71	0.97	8.21	5.48	4.90	1.91
Elbow left	4.58	0.45	3.69	0.24	3.89	1.11	15.44	5.41	5.78	1.99
Wrist left	6.77	0.98	5.61	0.28	6.21	1.54	18.11	5.63	10.80	3.39
Shoulder right	1.18	0.47	2.37	1.02	3.06	1.47	10.91	6.13	4.99	2.48
Elbow right	2.35	1.15	11.48	6.19	3.75	1.12	16.84	6.58	5.99	2.67
Wrist right	3.27	1.57	15.45	6.69	5.60	1.53	20.47	7.09	14.84	5.20
Hip left	1.36	0.39	1.51	0.28	2.81	1.04	7.90	4.10	3.58	0.98
Hip right	1.31	0.52	1.99	0.38	2.83	0.83	7.90	4.00	3.46	1.01
Spine shoulder	1.08	0.28	1.79	0.33	2.30	1.01	6.48	3.51	3.35	1.45

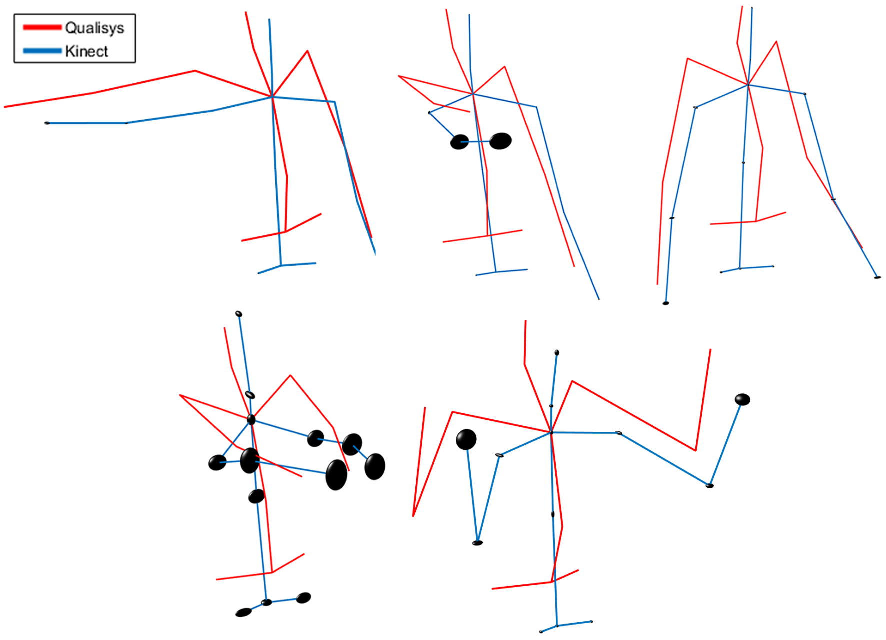


Figure 4.6: Depiction of the SD of the error for all repetitions of each exercise modelled as ellipsoids. Exercise order from top left; Arm to Side, Arm to Front, Shoulder Shrug, Twist, Drying Off.

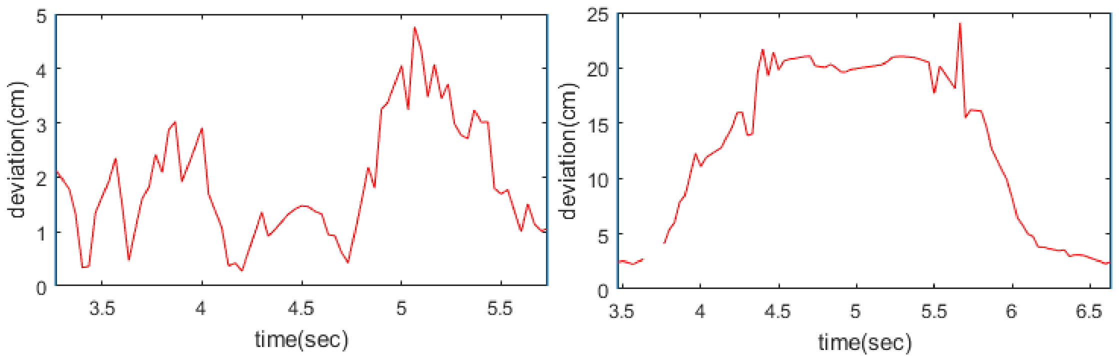


Figure 4.7: Plots showing a participant's WristRight joint deviation from the ground truth joint position. Arm to side (left) and arm to front (right).

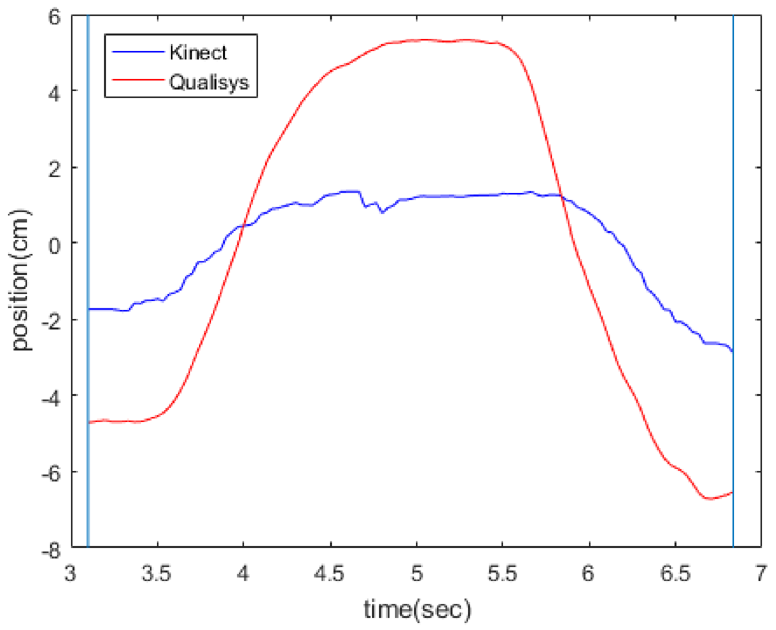


Figure 4.8: Plot showing a participant's ShoulderRight joint position in the Y axis when performing the shoulder shrug exercise.

4.3.2 Quantitative analysis results

The joint names described in this section are taken from Kinect SDK. The 'Twist' exercise had a relatively high SD, as shown in Table 4.4 and Figure 4.6, and shows how the pose estimation algorithm struggles with poses with limited depth data of the joint and surrounding areas, such as when the arms were extended towards the depth sensor. When comparing the exercises Arm to Side and Shoulder Shrug against the exercises Arm to Front and Twist, limb joint positions, for exercises performed along the Y and Z axes, are more inaccurate than along the X and Y axes. This appears to be due to occlusion.

In Figure 4.7, the error of the joint positions are larger for the Arm to Front exercise than the Arm to Side, this is understandable, as there would be limited depth data for the arm as it is raised to 90 degrees.

Figure 4.8 shows the algorithm struggling to track the true movement of the shoulder joint even though there is no occlusion in the depth data around the shoulder joint.

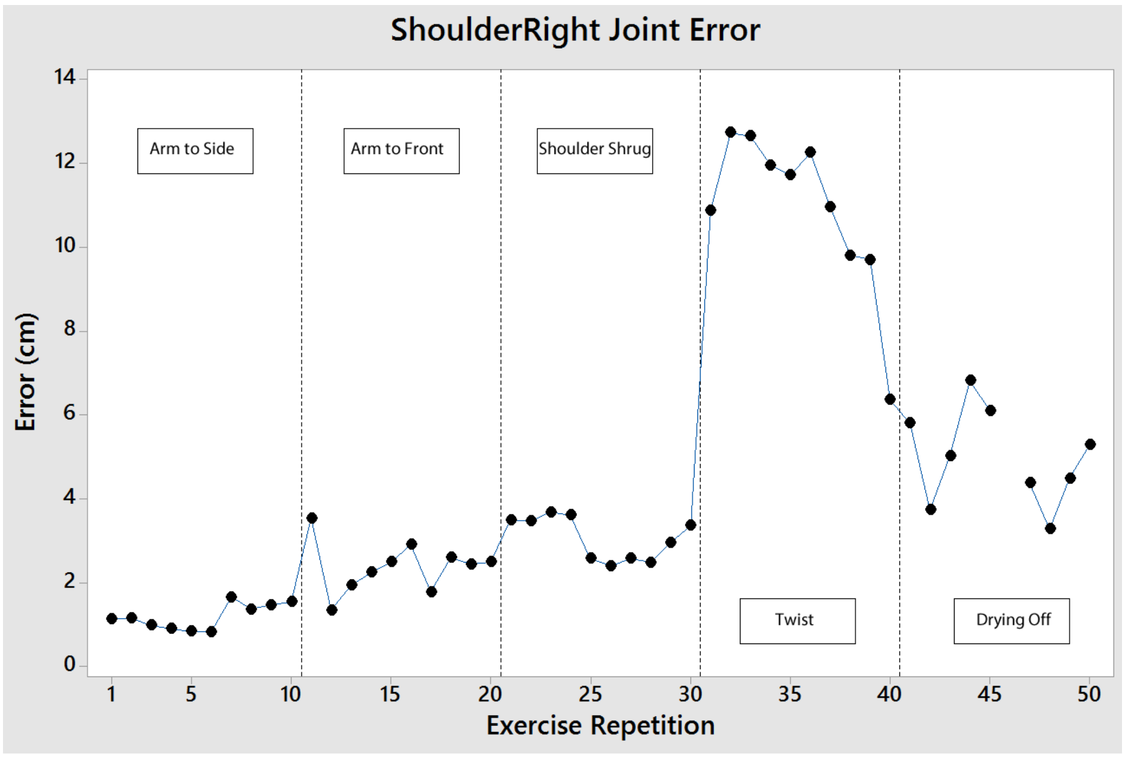


Figure 4.9: Exercise repetition errors of the ShoulderRight joint.

This could be due to the pose estimation algorithm being trained on a dataset containing no or limited data of correctly labelled elevated shoulder joints.

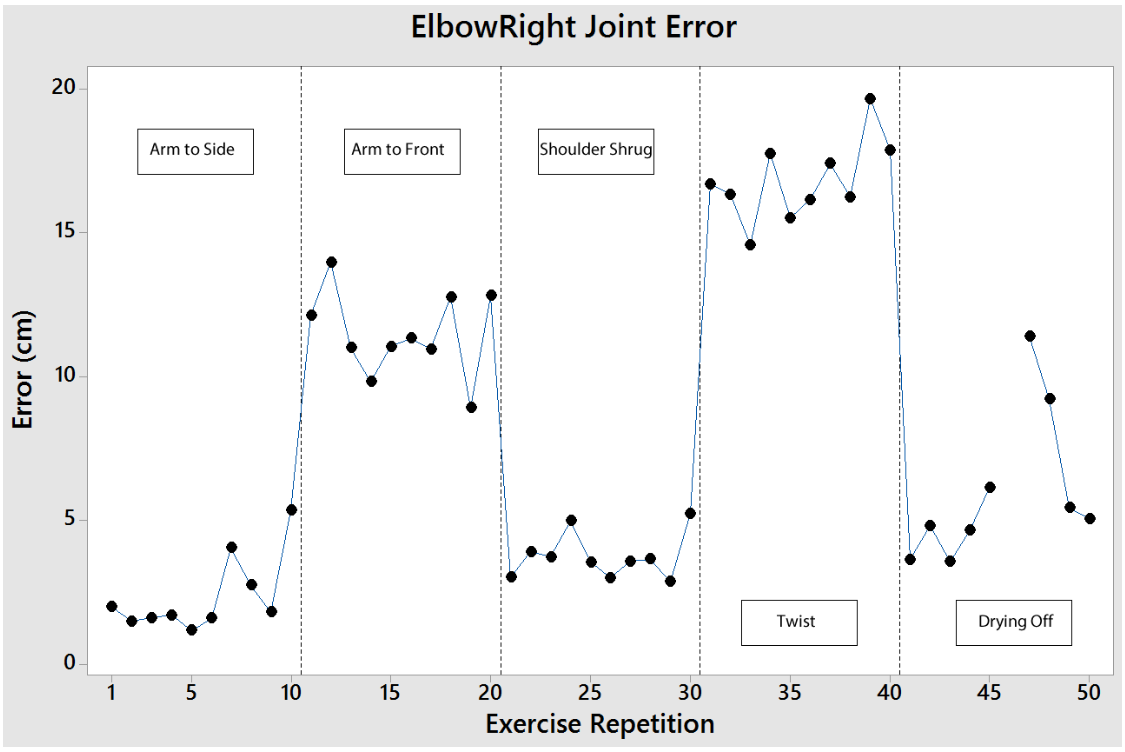


Figure 4.10: Exercise repetition errors of the ElbowRight joint.

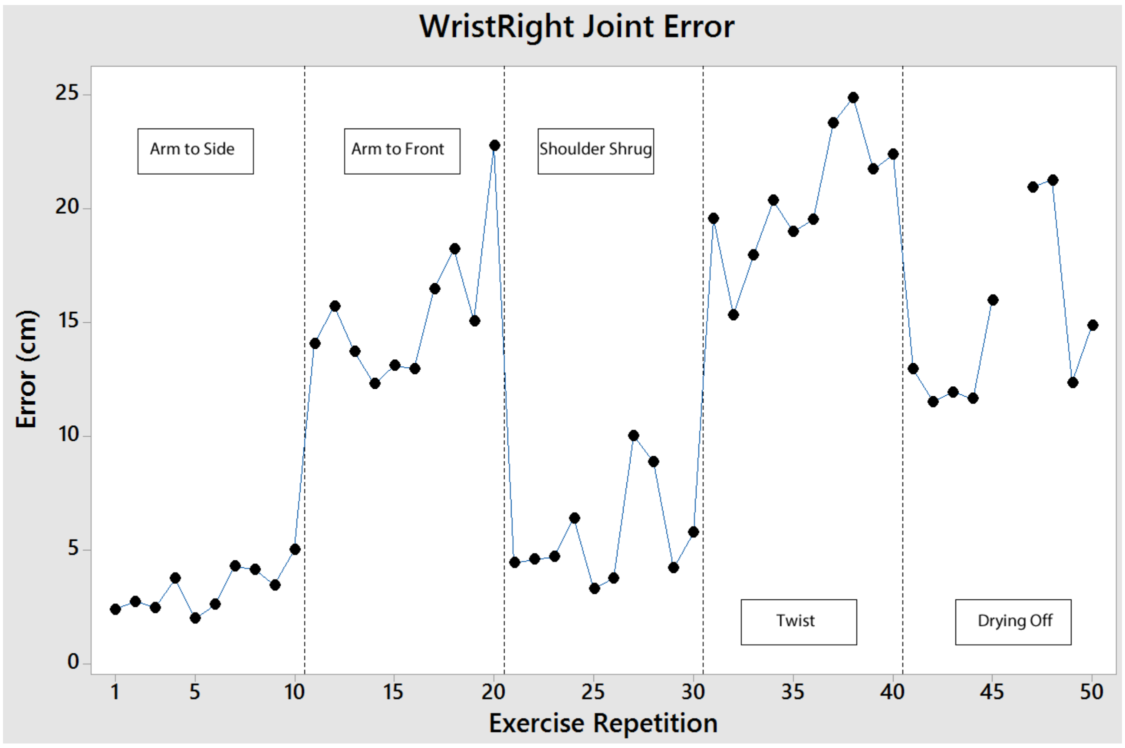


Figure 4.11: Exercise repetition errors of the WristRight joint.

Figures 4.9, 4.10 and 4.11 show the errors for each exercise repetition for the shoulder, elbow and wrist joints; as these are considered the most important joints for assessing these exercises. Figure 4.9 shows the shoulder joint has a relatively large error when being tracked during the Twist exercise. Figure 4.10 shows the elbow joint with relatively large error on the Arm to Front and Twist exercise. Figure 4.11 shows the wrist joint has the largest mean error when compared to the elbow and shoulder joint. It also has a relatively large error during the Arm to Front, Twist and Drying Off exercises. Interestingly in Figure 4.9 the ShoulderRight joint does not appear to be relatively erroneous for the Shoulder Shrug exercise, but the physiotherapists reported UT and MT for this joint on this exercise. This suggests absolute joint error is not a definitive measure of acceptability.

4.4 Inter-rater agreement of clinical analysis

4.4.1 Inter-rater agreement methodology

In order to determine the inter-rater agreement among the physiotherapists, their ratings on the accuracy of the joint position estimations, shown in Table 4.3, were statistically evaluated. The statistical measure used was the krippendorff's alpha [114], this measure shows the level of agreement among multiple raters, can handle different levels of measurements and handles missing data. Reasons for choosing Krippendorff's alpha over other reliability measures is discussed in more detail in [115]. This metric could be used to improve and compare criteria used to rate technology for clinical use.

Further to the statistical measure, we graphed the physiotherapists' ratings for each observation, showing the observations with the least and most agreement. Finally, the number of ratings for each category in the criterion are summed for each

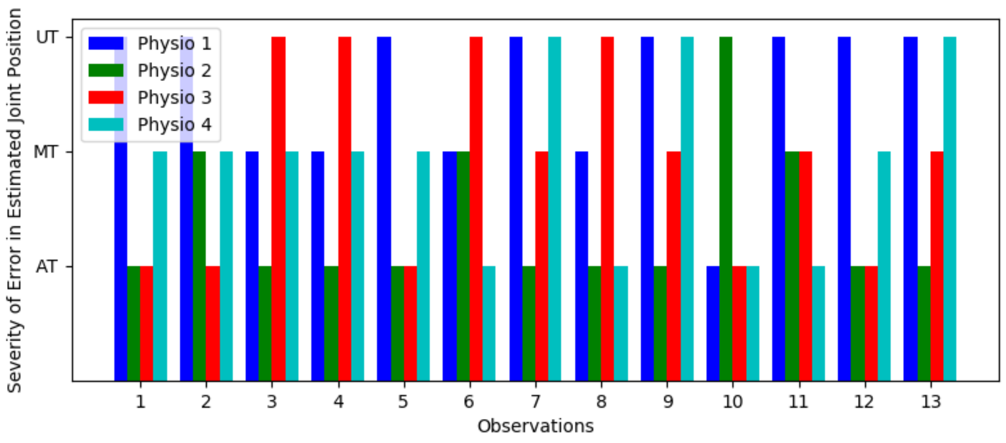


Figure 4.12: Observations with the most divergence in opinion. See Table 4.2 for the definitions of AT, MT and UT.

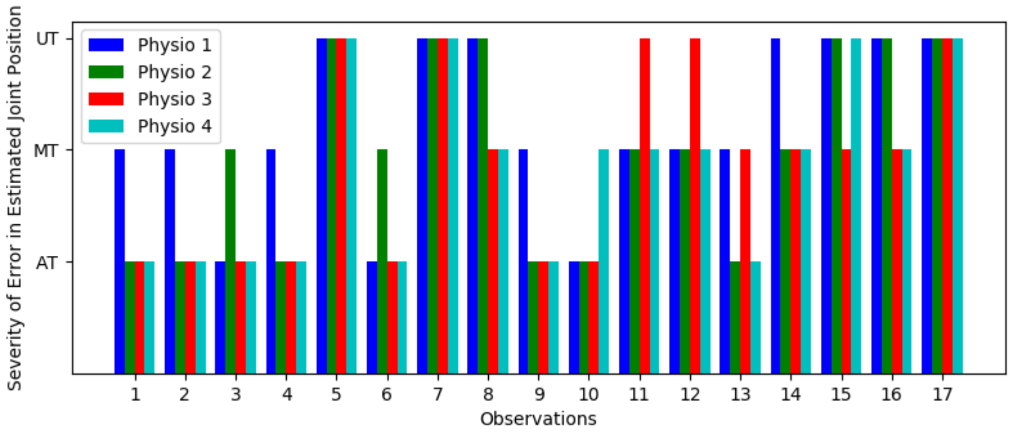


Figure 4.13: Observations with the most agreement in opinion. See Table 4.2 for the definitions of AT, MT and UT.

physiotherapist and presented in a table to show the individual physiotherapist's acceptability in the joint position estimations.

4.4.2 Inter-rater agreement analysis

Figure 4.12 shows the observations with the most divergence of opinion i.e. the observations with a standard deviation of 0.71 and above. The first observation shows that physiotherapist 1 believed the estimated joint position led to a significant misclassification in the assessment, whereas physiotherapists 2 and 3 believed the estimated joint position did not lead to a misclassification in the assessment.

Table 4.5: The number of ratings for each category each physiotherapist has rated the estimated joint positions for all observations.

	AT	**MT**	**UT**
P1	4	11	15
P2	15	8	7
P3	12	9	9
P4	11	12	7

Figure 4.13 shows the observations with the most agreement in opinion. Of all the thirty observations only three were found to have agreement by all four physiotherapists and these observations were rated as unacceptable tracking.

Krippendorff's alpha [115] provides a statistical measure of inter-rater agreement. These observations have a Krippendorff alpha of 0.28. Landis, et al [116] stated that an alpha of 0.21-0.4 suggests a fair level of agreement, however they state that this strength of agreement is somewhat arbitrary. Krippendorff [117] suggests that conclusions can be tentatively made for values between 0.67 and 0.8, but states cut-offs can vary. Zapf, et al [118] warn against making interpretations solely on simple generalised cut-offs. When used relatively, the Krippendorff's alpha can be used as a measure of improvement to the raters' criterion e.g. rerunning the assessment given a new criterion that achieves a higher Krippendorff alpha, this would indicate that the criterion has improved as raters have a higher level of agreement. It should also be considered that our criterion has two categories, MT and UT, that are given when there is a misclassification in the assessment, so borderline MT and UT ratings given by two different physiotherapists would decrease Krippendorffs alpha, and therefore the level of agreement, but still suggest the accuracy of the joint position estimations are unsuitable.

Table 4.5 shows the number of ratings each physiotherapist gave to each category. P1 tends to be much harsher, flagging more observations as showing 'unacceptable tracking' than the other physiotherapists. Of all the 120 ratings, only 42 (35%)

Table 4.6: Example of an updated criterion from the original in Table 4.2, incorporating objective measures to guide the rater. The 2 and 8 cm of error are examples that would come from an objective measure of the technology.

Evaluation criterion	
Acceptable Tracking (AT)	The joint's estimated positions' have less than 2cm of error. The error in the position does not lead to misclassification in the assessment, e.g. a limb is showing no flexion when no flexion is occurring.
Moderately Acceptable Tracking (MT)	The joint's estimated positions' have between 2cm and 8cm of error. The error in the position leads to a minor misclassification in the assessment, e.g. a limb is showing minor flexion when no flexion is occurring.
Unacceptable Tracking (UT)	The joint's estimated positions' have above 8cm of error. This change in position leads to a significant misclassification in the assessment, e.g. a limb is showing severe flexion when no flexion is occurring.

were considered acceptable tracking, thus the majority of joint position estimations are not suitably accurate enough to make a correct assessment on the performance of the exercise.

It should be noted that only a subset of joints considered important for assessing the exercise were observed by the physiotherapists, for example limb joints. These results highlight the difficulty for clinicians to accurately assess the suitability of technology intended for clinical use. However, this was a small study. It remains inconclusive whether the raters' variability in assessment was for example due to the medium of visualising the data, limited training on the task and/or the criterion they followed.

To aid raters' in the clinical assessment of technology, we propose the incorporation of objective measures within the rater criterion to ensure a more structured and objective approach to evaluation. For example, when assessing whether joint position estimations, tracking a human skeleton, from a depth sensor is sufficiently acceptable for clinical use, the joint estimations from a more robust technology could be used as a ground truth, such as a motion capture studio, in order to measure the accuracy of the depth sensor. This would provide the assessors with error distance measures of estimated joints helping guide their final decision. Table 4.6 shows an example of an updated criterion, incorporating the objective measures the raters could use to guide their assessment.

Furthermore, due to the literature indicating high inter-rater variability in expert clinical assessments, objective measures from technological tools could aid in the objective assessment of patient performance. For example, within stroke rehabilitation, tracking the patient's joint positions would provide measures such as arm flexion, by calculating the angle around the elbow joint, or calculating the range of motion. This approach could provide a fair assessment using calculated performance measures to determine the range of motion, severity of compensatory movements and stability in movements, etc.

We have evaluated the inter-rater reliability of clinicians rating the suitability of a new technology intended for clinical use without objective measures to guide opinion. The results indicate a low level of inter-rater agreement suggesting that relying solely on the opinions of clinical experts for determining the clinical validity of new technology may be insufficient. It is still unclear what factors may have contributed to the inter-rater variability, such as the medium used for assessment, insufficient training on the new task and/or an inadequate grading criterion. However, we hypothesise that the introduction of technological tools that provide objective measures can be incorporated into the assessment criterion to guide clinicians' decision making, encourage an objective evaluation and improve inter-rater agreement. Further research is required to validate this hypothesis.

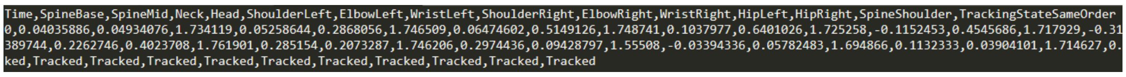
4.5 Summary

Based on the clinical evaluation supported by the quantitative measures we conclude that the pose estimations are mostly inadequate for correctly assessing stroke rehabilitation exercises. When performing upper-body gross exercises the shoulder joints act as indicators for incorrect movement of limbs. For example, elevated shoulders are a common compensatory movement among stroke patients and needs to be detected during rehabilitation exercises. However, the pose estimation algorithm failed to accurately track the true movement of the shoulder joints even when the joints were in a tracked state. This could be improved by retraining the pose estimation algorithm with correctly labelled shoulder joints that contain training data with elevated shoulders. Partial or full occlusion in the depth data surrounding a joint causes unacceptable jitter and tracking. The pose estimation algorithm can misclassify depth pixels of objects that are required for an exercise, e.g. a towel used in Drying Off, as body parts, resulting in unacceptable jitter and tracking error. Similarly, for seated exercises it is recommended a user be seated on a perching stool to eliminate the chances of seat arms being tracked as limb joints. When assessing the suitability of a pose estimation algorithm intended for rehabilitation applications, solely performing a quantitative analysis does not provide conclusive answers, a clinical evaluation supported by a quantitative analysis is required to determine suitability. This is because the measured accuracy of the joint estimations does not take into account that joints require a varying degree of accuracy to correctly assess a given exercise. This is evident by the shoulder joints for the Arm to Front and Shoulder Shrug exercises, whereby the ShoulderRight joint displays similar error in Figure 4.9 for these exercises but has a mean classification of AT and MT/UT respectively from the clinicians, as presented in Table 4.3. Future pose estimation algorithms should consider using temporal information to infer joint positions that have full or partial occlusion. This could reduce the possibility of inferred joint positions displaying sudden and extreme changes in position. We are currently working on techniques that use temporal information in a scalable way to improve joint tracking. Estimating joint rotation should be considered for a more in-depth and correct assessment of a patent's performance. Constraining joint estimations to within the anatomical limits of the human body should ameliorate severe tracking error and solve the issue of anatomically impossible poses. In order to make the task of automatically assessing the quality of an exercise easier, clinicians should be consulted on selecting exercises that are useful for rehabilitation but provide fewer or easier challenges for pose estimation algorithms. For example, clinicians highlighted using less obtrusive objects such as a rod or walking stick to perform the Drying Off exercise, to reduce the chance of severe tracking error.

5

Segmentation of the signals from depth sensors for stroke exercises

This chapter proposes a segmentation algorithm that requires only a single exemplar exercise repetition from a healthy participant to segment subsequent repetitions from other subjects, including those with impaired mobility. The accuracy of this segmentation algorithm was evaluated on a public exercise dataset and our own stroke rehabilitation dataset containing real stroke patient exercise data captured on Kinect V2.



```
Time,SpineBase,SpineMid,Neck,Head,ShoulderLeft,ElbowLeft,WristLeft,ShoulderRight,ElbowRight,WristRight,HipLeft,HipRight,SpineShoulder,TrackingStateSameOrder
0,0.04035886,0.04934076,1.734119,0.05258644,0.2868056,1.746509,0.06474602,0.5149126,1.748741,0.1037977,0.6401026,1.725258,-0.1152453,0.4545686,1.717929,-0.31
389744,0.2262746,0.4023708,1.761901,0.285154,0.2073287,1.746206,0.2974436,0.09428797,1.55508,-0.03394336,0.05782483,1.694866,0.1132333,0.03904101,1.714627,0.
ked,Tracked,Tracked,Tracked,Tracked,Tracked,Tracked,Tracked,Tracked,Tracked,Tracked
```

Figure 5.1: Example of the joint data of stroke patients performing rehabilitation exercises stored as comma-separated values. Each frame contains a timestamp, 3D coordinates of each joint and the tracking state of each joint.

5.1 Stroke rehabilitation dataset

This section describes the methodology for collecting patient exercise data, including the experimental set-up, justification for the selected exercises and the type of data collected.

This research required a dataset containing movement data of stroke patients performing rehabilitation exercises captured from a commercial depth sensor. A suitable dataset was not publicly available and thus we collected and produced a suitable dataset.

Ethical approval was obtained from the University's ethical approval board. A participant recruitment pack [1] was created and distributed at Nottingham CityCare's stroke group rehabilitation sessions.

5.1.1 Data collection methodology

A data collection tool was developed in C# [2] to record depth, RGB and joint position data in real-time from a Kinect sensor. All Kinect V2's joints, as shown in Figure 4.1, are recorded, this includes the X, Y and Z coordinates and a timestamp. The tracking state of each joint is also recorded for each frame. The tracking state categorises the confidence of each joint position estimation as tracked when the confidence is high and inferred when the confidence is low. An example of the recorded joint data is shown in Figure 5.1.

[1]See appendices for the participant information pack.
[2]See appendices for a snippet of the code.

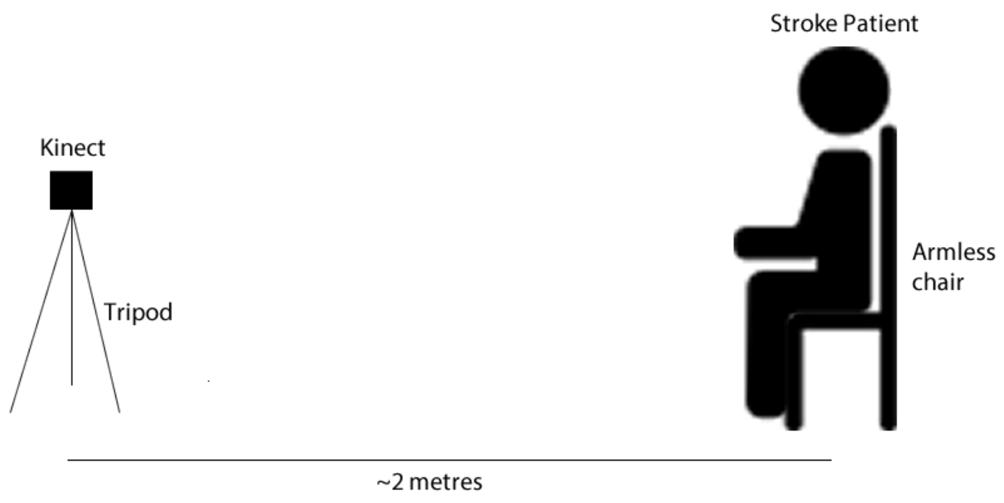


Figure 5.2: Depiction of the data collection set-up performed within the home.

Five stroke participants, as shown in Figure 5.3, were visited at home. The set-up of the experiment is depicted in Figure 5.2. The Kinect depth sensor was attached to a tripod and setup facing the participant. They were asked to perform at least three repetitions of five exercises from the GRASP manual [4], a stroke rehabilitation exercise manual. The exercises were selected by a physiotherapist with the aim of selecting a set of upper-body exercises with a range of difficulty. An exercise requiring a towel to be grasped was included to test the depth sensor's pose estimation algorithm (PEA) when an object is held by the participant. The five exercises the participants were asked to perform were, as labelled in the GRASP manual, Arm to Side, Arm to Front, Shoulder Shrug, Twist and Drying Off. Physiotherapists from Nottingham CityCare categorise the absolute performance of stroke patients into three levels. Level one categorises stroke patients with severe levels of impairment, level two categorises patients with moderate impairment and level three categorises patients with minor impairment. Of the five stroke participants, four participants were considered level one and one participant was considered level two. All data collection sessions were observed by a physiotherapist.

The initial development did not include the ability to record the RGB data, this resulted in the first participant's data not being used in the dataset as this data

Figure 5.3: Four of the stroke patients participating in the data collection. Note, the fifth participant did not have RGB data recorded and therefore was not included in the final dataset.

medium was required for humans to validate, for example, the video data can be used to find the ground truth start and end of an exercise repetition to verify the accuracy of the segmentation algorithm.

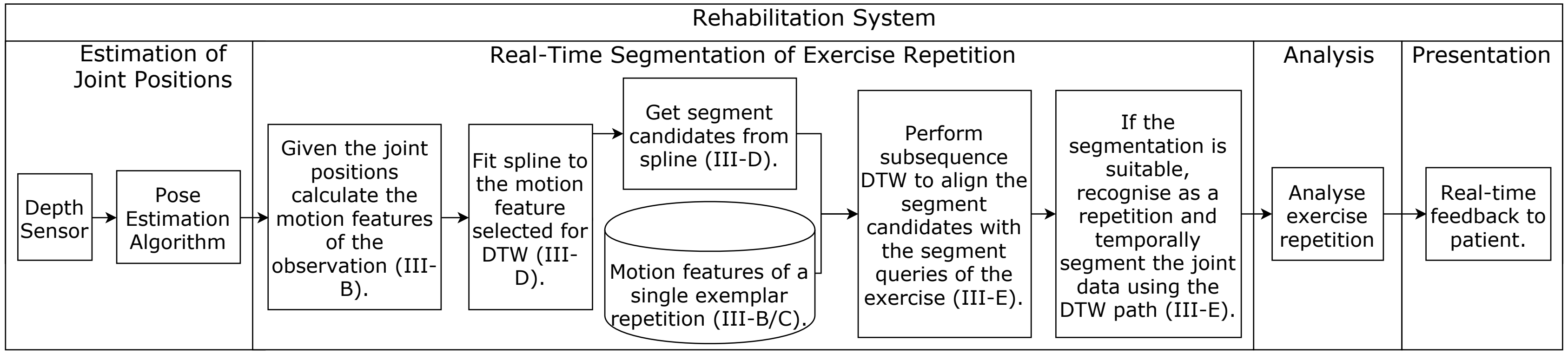

Rehabilitation System
Estimation of Joint Positions
Real-Time Segmentation of Exercise Repetition
Analysis
Presentation
Depth Sensor
Pose Estimation Algorithm
Given the joint positions calculate the motion features of the observation (III-B).
Fit spline to the motion feature selected for DTW (III-D).
Get segment candidates from spline (III-D).
Motion features of a single exemplar repetition (III-B/C).
Perform subsequence DTW to align the segment candidates with the segment queries of the exercise (III-E).
If the segmentation is suitable, recognise as a repetition and temporally segment the joint data using the DTW path (III-E).
Analyse exercise repetition
Real-time feedback to patient.

Figure 5.4: Flowchart of the segmentation algorithm and how it fits into a proposed rehabilitation system.

5.2 Segmentation algorithm

5.2.1 Algorithm overview

Figure 5.4 depicts a high-level flowchart of a proposed rehabilitation system to incorporate the segmentation algorithm.

The challenge is to find the start and end of an exercise repetition in real-time, thus enabling analysis of the repetition and responsive feedback to the user. This approach requires only a single exemplar repetition from a single subject for each exercise. This approach assumes that the exercise being performed is known; this follows current rehabilitation exercise regimes where the exercises are presented to the patient in a specified order. The term exemplar refers to the motion data of a single repetition of an exercise performed correctly and the term observation refers to the motion data collected in real-time of a patient or other subject.

Our proposed algorithm is the combination of several components, inspired from literature, to solve the challenge of real-time segmentation of repetitions from subjects with impaired mobility. This approach evaluates a subset of sequences in the observation using subsequence dynamic time warping (SDTW) on a 1-D human motion feature ranked as the most informative. If the alignment is similar, the warping path is used to measure the similarity of the other 1-D motion features in the human motion descriptor to determine if a repetition of an exercise was performed, meaning SDTW is only performed on the most informative 1-D feature.

The following list provides a brief overview of the key aspects of the proposed algorithm:

- Exercise repetitions represented as a temporal array of unit direction vectors between joints rotated to a local coordinate system. These features make the system invariant to skeleton size, position and plane while remaining selective to the exercise. Detailed in section 5.2.2.

- Pre-processing of the exemplar repetition to rank features by importance and find the total cost threshold for each motion feature. Detailed in section 5.2.3.
- Fitting a spline to the DTW feature using cubic spline interpolation, as proposed by [119]. This improves invariance to noise and instability in movements by capturing the general trend of the movement. Detailed in section 5.2.4.
- Feature extraction, achieved by extracting key features from the spline, such as zero-velocity crossings. This ensures the algorithm can run in real-time. Detailed in section 5.2.4.
- Normalisation of the DTW motion feature to zero-mean and unit variance, as proposed in [120]. This scales the DTW features to comparable ranges before performing DTW alignment. Detailed in section 5.2.5.
- Alignment of the exemplar to the observation using SDTW. This ensures the segmentation is invariant to speed as rehabilitation subjects tend to perform repetitions slowly. Detailed in section 5.2.5.
- Segmentation confirmation/rejection is performed using the DTW warping path on other motion features to quickly calculate their total cost and compare against the total cost threshold. This reduces false positive segmentations. Detailed in section 5.2.5.

5.2.2 Human motion descriptor

The human motion descriptor consists of a set of 1-D human motion features. A single motion feature is a temporal array of a single component of a unit direction vector between joints e.g. the X axis of a unit direction vector between the shoulder joint and the elbow joint. This motion feature is more selective, when compared to

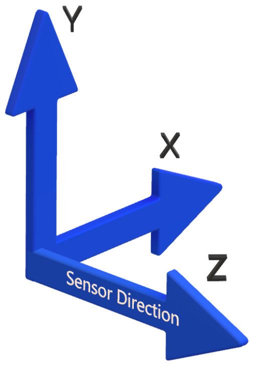


Figure 5.5: Depiction of the global coordinate system used in this paper, this follows the same coordinate system used by Kinect.

angles, as they describe the direction in which a bone is moving, as well as its speed along the given axis. The motion feature is invariant to body size and selective to direction. A set of these motion features are used to perform a segmentation but only one of these motion features is used for finding the warping path using SDTW.

To achieve viewpoint invariance, a local coordinate system is used as proposed by [41]. For each frame in the features, the direction vector between the hip joints is aligned parallel to the X axis of the global coordinate system, shown in Figure 5.5. Then, given this rotation, each of the features is rotated around the Y (vertical) axis by the same rotation.

5.2.3 Pre-processing of exemplar repetition

Given an exemplar repetition of an exercise and a set of joints, suitable sets of joints are selected to become candidates for the automatic generation of motion features e.g. the squat exercise may use the sets {{HipLeft, KneeLeft}, {HipRight, KneeRight}}. Using an approach similar to [40], the motion features are ranked based on the most informative i.e. most change over time. The motion features taken from the exemplar exercise are ranked by their change over time, as follows:

$$FR = \sum_{i=2}^{len(F)} |F_i - F_{i-1}| \tag{5.1}$$

where FR is the feature rank and F is the motion feature. The FR of each F is stored in a set FRS ordered by FR. The motion feature with the most change over time is used as the DTW feature for alignment.

Total cost thresholds for each motion feature are defined as follows:

$$TCT = ((FR/max(FRS)) * DM) + DB \tag{5.2}$$

where TCT is the total cost threshold that the same motion feature in the observation must be lower than to be considered a segmentation, DM is the distance multiplier and DB is the distance base. DM and DB are parameters that need setting for each exercise. This ensures motion features taken from joints with more movement have a higher TCT. This approach reduces the number of parameters that require setting for each exercise. The FR and TCT are calculated once from the exemplar repetition of the exercise.

Given that the exemplar represents a repetition of the exercise with complete range of motion, a minimum scale parameter is calculated to reject subsequences of the observation with a small range, calculated as follows:

$$MinScale = (max(DF) - min(DF)) * MinScalePerc \tag{5.3}$$

where $MinScalePerc$ is the percentage of the range, DF is the DTW feature of the exemplar and $MinScale$ is the minimum scale attained by a subsequence to be considered for segmentation.

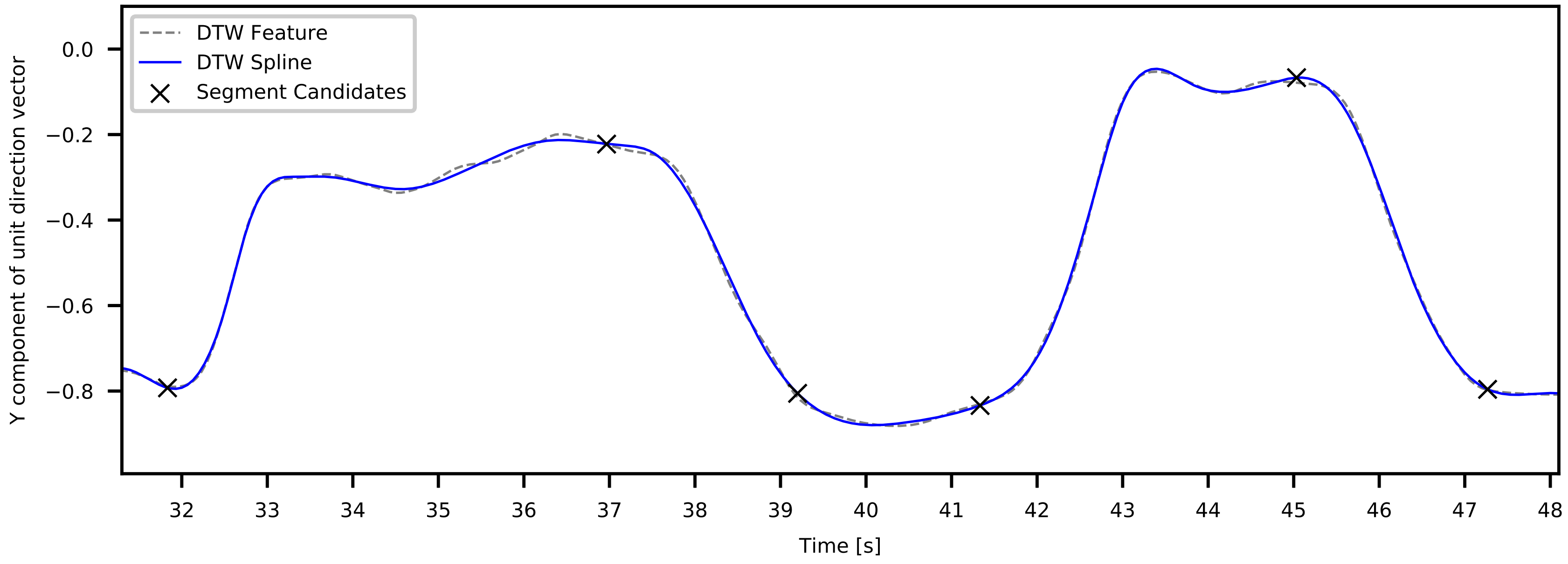


Figure 5.6: Plot showing a cubic spline fitted to the DTW feature and the segment candidates selected for DTW alignment.

5.2.4 Feature extraction

The motion features will likely contain superfluous information for the task of segmentation as joint data is captured many times a second, e.g. Microsoft Kinect captures joint data at 30Hz. Feature extraction enables a real-time implementation and improves accuracy of segmentation by selecting the most informative motion features which we define in this section. The feature extraction occurs on the DTW feature, chosen using the feature ranking method in section 5.2.3, of both the exemplar and observation. This results in a subset of values from the DTW feature consisting of only the key motion features, these features will be referred to as segment queries and segment candidates respectively and collectively referred to as segment points; these concepts are explained in detail in this section.

Before extracting the segment points, the general trend of the DTW feature is calculated by fitting a cubic spline to produce a DTW spline, as shown in Figure 5.6. The first derivative of the DTW spline is also calculated to retrieve the velocity.

Segment queries are then extracted using the first two methods while segment candidates are extracted using all of the following methods:

1. Zero-velocity crossings. Given the spline representing the velocity, extract the DTW spline values where the velocity crosses zero, as proposed in [121].
2. The first and last values in the DTW spline. As we are dealing with real-time segmentation, when new joint data arrives, the very latest value is a potential repetition end segment candidate. The oldest value is a potential repetition start segment candidate. The first and last values of the exemplar will of course be segment queries that represent the start and end of a repetition.
3. Values in the observation's DTW spline that cross through the first value in the exemplar's DTW spline. This is required because if movement occurs before the repetition starts then method 1 will miss a potential segment

Algorithm 1 Pseudocode to find the optimal warp path for a single time step. One-based indexing.

```
Input: C {Segment candidates}
Input: Q {Segment queries}
 1: $m \leftarrow len(Q)$
 2: $l \leftarrow len(C)$
 3: $Q \leftarrow Normalise(Q)$ {Normalise query to zero-mean and unit variance. This
    step can be performed once and stored as the query does not change}
 4: $OW \leftarrow []$ {Optimal warp path i.e. path with lowest total cost}
 5: $OTC \leftarrow \infty$ {Total cost of OW}
 6: for $i \leftarrow l; i > 0; i \leftarrow i - 1$ do
 7:   $S \leftarrow C[c_i, ..., c_l]$ {Subsequence of candidate}
 8:   if $max(S) - min(S) < MinScale$ then
 9:     continue
10:   end if
11:   $S \leftarrow Normalise(S)$
12:   $W, A \leftarrow GetWarpPath(Q, S)$ {A is costs matrix}
13:   $TC \leftarrow GetTotalCost(W, A, m)$
14:   if $TC < OTC$ then
15:     $OTC \leftarrow TC$
16:     $OW \leftarrow W$
17:   end if
18: end for
19: return OW, OTC
```

candidate or will segment early.

The segment points are further reduced by removing segment points that have similar values to surrounding segment points with the latest segment point kept. Explicitly, a rolling absolute difference of the segment points is calculated, and if this difference is below a threshold then the oldest segment points are removed. This can be seen in Figure 5.6 between ≈43 to ≈45 seconds where multiple segment points would have been proposed at the peak but only the latest segment point was kept. These segment points become the DTW features for alignment.

5.2.5 Segmentation

Algorithm 1 presents the pseudocode for finding the DTW warping path and is

Table 5.1: Glossary of the main variables presented in this section.

Variable/s	**Definition**
Q, m	Query sequence with length m
C, l	Candidate sequence with length l
S, n	Subsequence of C with length n
A	DTW costs matrix with length $m \times n$
D	DTW cumulative costs matrix with length $m \times n$
i, j	Indices referring to an element in a one or two dimensional array e.g. $D_{i,j}$ refers to the ith row and jth column of D
W, k	DTW warping path with length k. Each element consists of a pair of indices relating to elements in Q and S that are aligned, e.g. W_k containing a pair of indices $(i, j) = W_{k_{i,j}}$ = the alignment between Q_i and S_j

described in more detail in this section.

The problem formulation is as follows. Given two sequences, a query sequence Q with length m and a candidate sequence C with length l, find an alignment between Q and a subsequence S with length n, of C that minimises the DTW cost between Q and S. Q consists of the segment queries and C consists of the segment candidates described in section 5.2.4. Note that the segment points in C and Q will likely have differing distances temporally, e.g. C_{i-1} and C_i may be temporally closer together than C_i and C_{i+1}, when mapped back to the original motion feature. Further definitions are used in this section. W refers to the warping path, of length k, consisting of the indices of the segment points from Q and S that are aligned. A and D refer to the costs and cumulative costs matrices between Q and S respectively. i and j are indices referring to an element in a one or two dimensional array, e.g. $D_{i,j}$ would refer to the ith row and jth column of D. Note that the indices are stored in W instead of the values of the segment points, e.g. $W_1 = (1, 1)$ instead of $W_1 = (Q_1, S_1)$, this allows us to map the warping path indices to other motion features in order to calculate their total cost. These variables are defined in table 5.1.

This approach uses SDTW for the sequence matching between Q and a suitable

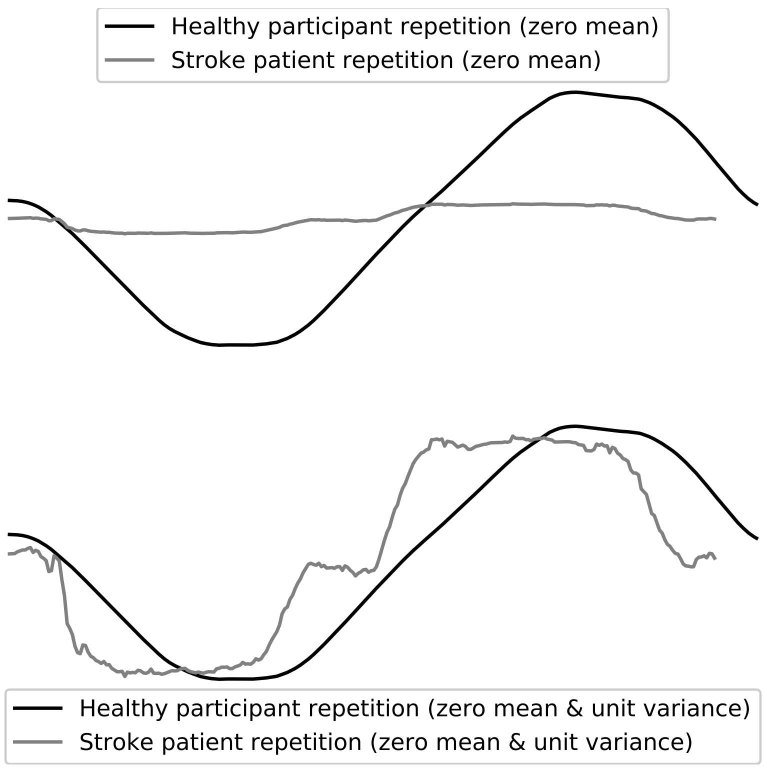


Figure 5.7: The top graph shows a repetition of a healthy and stroke patient DTW motion feature scaled to zero mean. The bottom graph shows the result of normalising the same features to zero mean and unit variance.

subsequence of C that meets the conditions described in this section.

Sequences Q and S are normalised to zero-mean and unit variance, using the following equation defined in [120]:

$$\hat{T} = \{\hat{t}_1, ..., \hat{t}_{end}\} \text{ where } \hat{t}_i = \frac{(t_i - \mu)}{\sigma} \tag{5.4}$$

Given input sequence $T = \{t_1, ..., t_{end}\}$ where μ and σ are the mean and standard deviation of T respectively. This step is shown on lines 3 and 11 in algorithm 1. Sart, et al. [120] note that some researchers suggest normalising between a range of [0,1] or [-1, 1] but state that this approach is sensitive to noise. Figure 5.7 shows the result of normalising a Twist repetition from a healthy subject and a repetition from a stroke patient to zero mean and unit variance. It can be seen that this normalisation step brings the motion feature of the healthy subject and stroke patient subject into a comparable range.

Before performing DTW on Q and S, the scale of S must meet a minimum scale threshold. Specifically, the distance between the minimum and maximum value in S is checked against the minimum scale value calculated in equation 5.3. This ensures the motion was significant enough to be considered as a repetition and is performed before normalising S, see line 8 in algorithm 1.

For each iteration in the *for* loop on line 6 algorithm 1; a new subsequence is created on line 7, DTW alignment between Q and S is performed on line 12 and the total cost of the warp path is calculated on line 13. It can be seen that for each iteration, the length of S increases by one, starting with the latest segment candidate in C, this approach finds the latest repetition.

The DTW presented here follows the original implementation [87] with modifications and conditions as follows:

- Start and end boundary condition [87]: The first and last indices of Q and S are respectively placed in the first and last element in the warping path e.g. $W_1 = (1, 1)$ and $W_k = (m, n)$. Note that DTW aligns Q to S instead of C.
- Continuity condition [87]: The indices in the warp path advance at most one index, i.e. step size is 1, this ensures all indices of the segment points in Q and S are in W; e.g. $W_k - W_{k-1} \in \{(1, 0), (0, 1), (1, 1)\}$
- Normalisation to zero-mean and unit variance, as proposed in [122]. Q and S are normalised to zero-mean and unit variance before alignment. This step ensures segment candidates from a patient with a low range of motion are correctly aligned to the segment queries from a healthy subject by normalising the peaks and troughs.

- Early abandoning of subsequence alignment [122]. By storing the total cost of the best subsequence so far, we can abandon the calculation of the new subsequence if the total cost has gone higher than the best so far.
- The total cost of the warp path between Q and S differs from the original DTW algorithm [87], later defined in this section.

As proposed in [87], to find the optimal warp path W, the costs, i.e. distances, between each segment point in Q and S needs to be calculated. The Manhattan distance is used to calculate the difference. A and D are $m \times n$ matrices that store the costs and cumulative costs respectively. A, D and W are calculated as follows:

$$A_{i,j} = |Q_i - S_j| \tag{5.5}$$

$$D_{i,j} = \begin{cases} A_{i,j} \text{ if } i = 1 \text{ and } j = 1, \\ A_{i,j} + D_{i,j-1} \text{ if } i = 1 \text{ and } j > 1, \\ A_{i,j} + D_{i-1,j} \text{ if } i > 1 \text{ and } j = 1, \\ A_{i,j} + \min(D_{i-1,j-1}, \\ \qquad\qquad D_{i-1,j}, \\ \qquad\qquad D_{i,j-1}) \text{ otherwise} \end{cases} \tag{5.6}$$

$$W_{index} = (i,j) = \begin{cases} (1,1) \text{ if } i = 1 \text{ and } j = 1, \\ (m,n) \text{ if } i = m \text{ and } j = n, \\ (i, j-1) \text{ if } i = 1 \text{ and } j > 1, \\ (i-1, j) \text{ if } i > 1 \text{ and } j = 1, \\ \textit{indices} \text{ otherwise} \end{cases} \tag{5.7}$$

$$\textit{indices} = \begin{cases} (i-1, j) \text{ if } D_{i-1,j} \leq D_{i,j-1} \wedge \\ \qquad\qquad D_{i-1,j} < D_{i-1,j-1} \\ (i, j-1) \text{ if } D_{i,j-1} < D_{i-1,j} \wedge \\ \qquad\qquad D_{i,j-1} < D_{i-1,j-1} \\ (i-1, j-1) \text{ otherwise} \end{cases}$$

These calculations are performed on line 12 in algorithm 1. Note, W is calculated backwards starting with (m,n) until $(1,1)$ is appended to W.

The total cost TC, i.e. DTW distance, between Q and S given W, is calculated differently from the original DTW proposal. TC is calculated as follows:

$$TC = \frac{\sum_{p=1}^{m} \overline{\{A_{i,j} | (i,j) \in W \wedge i = p\}}}{m} \tag{5.8}$$

The average Manhattan distance of all segment candidates in S that align to a segment query in Q is calculated, this occurs for each Q_i and the averages summed. Finally, the result is normalised by m. This calculation occurs on line 13 in algorithm 1.

Window constraints, such as the Sakoe-Chiba band [87] and Itakura parallelogram [123], should be used with caution as there can be large and varying differences in time between segment points.

Algorithm 2 Pseudocode to confirm a segmentation of a repetition after performing subsequence DTW.

```
Input: $OW$ {Optimal Warp Path for this time step, see Algorithm 1}
Input: $OTC$ {Optimal Warp Path Total Cost for this time step, see Algorithm 1}
Input: $Paths$ {Optimal Warp Paths for the last $NT$ time steps. Global variable
    initialised to empty set}
Input: $Costs$ {Optimal Warp Path Total Cost for the last $NT$ time steps. Global
    variable initialised to empty set}
Input: $WRE$ {Wait for Repetition End boolean. Global variable initialised to
    FALSE}
 1: if $WRE = TRUE$ or $OTC < SegmentThreshold$ then
 2:   if $WRE = FALSE$ then
 3:     for all $F \in Features$ do
 4:       $TC \leftarrow GetFeatureTotalCost(OW, F, Q)$ {Subsequence of F is retrieved
          using indices in OW}
 5:       if $TC > TCT$ then
 6:         return
 7:       end if
 8:     end for
 9:     $WRE \leftarrow TRUE$
10:   end if
11:   $Paths.append(OW)$
12:   $Costs.append(OTC)$
13:   if $len(Costs) - argmin(Costs) > MaxLookAhead$ then {MaxLookAhead is
      the number of future time steps to check for a lower DTW cost}
14:     $RWP = Paths[argmin(Costs)]$ {Repetition Warp Path}
15:     $WRE \leftarrow FALSE$
16:     $Paths.clear()$
17:     $Costs.clear()$
18:     return RWP {Segmentation confirmed, RWP can retrieve joint positions
        for the repetition.}
19:   end if
20: end if
21: return
```

When the optimal warp path has been found for a single time step, its total cost is checked against a minimum total cost parameter. If it is below this parameter then the subsequence can be considered for segmentation.

Finally, as DTW alignment has only been performed on one motion feature, confirmation of the segmentation is performed as follows. Several motion features with the highest feature ranks, as described in equation 5.1, have their total cost calculated using the indices in the optimal warp path. If any of these values are above their total cost threshold, calculated in equation 5.2, then the segmentation is rejected. Otherwise the segmentation is confirmed for this time step. The pseudocode for confirming a segmentation is presented in Algorithm 2.

Future joint information is required to determine if this is in fact the very end of the exercise repetition. Thus, the trend of the DTW motion feature's total cost is measured over several time steps to check if it is still decreasing. Once the total cost begins to rise the segmentation of the observation occurs on the time step with the lowest total cost.

5.2.6 Parameters

The following list describes the key parameters of the segmentation algorithm:

- Minimum Scale Percentage: When performing DTW, the scale of the subsequence must be at least this percentage of the query scale to be considered for segmentation, see equation 5.3.
- Segmentation Threshold (ST): The optimal warp path's total cost must be less than this threshold to be considered for segmentation.
- DTW Distance Multiplier (DM): A multiplier to give motion features with more change over time a higher total cost threshold, see equation 5.2.
- DTW Distance Base (DB): A base value each motion feature is given when calculating the total cost threshold, see equation 5.2.

5.3 Segmentation algorithm results

5.3.1 General evaluation against public dataset

Three exercises were selected from the CMU Graphics Lab Motion Capture Database (CMU) [73] based on their similarity to rehabilitation exercises which tend to have repetitive motions. Thus, actions that were more activity based were ignored as they are out of scope of the proposed algorithm. During rehabilitation, patients follow an exercise regime presenting them with exercises to perform. Therefore, unlike the following papers [58,60,61,63,66,69,70] that deal with the recognition and segmentation of actions, our proposed algorithm is aimed at segmenting repetitions of a known exercise given a correct repetition of the exercise.

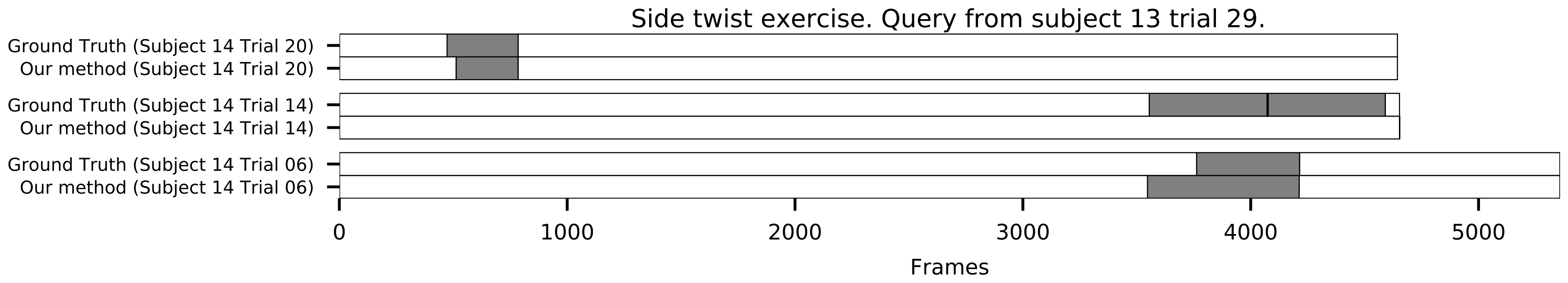


Figure 5.8: Segmentation performance of side twist exercise using query from subject 13.

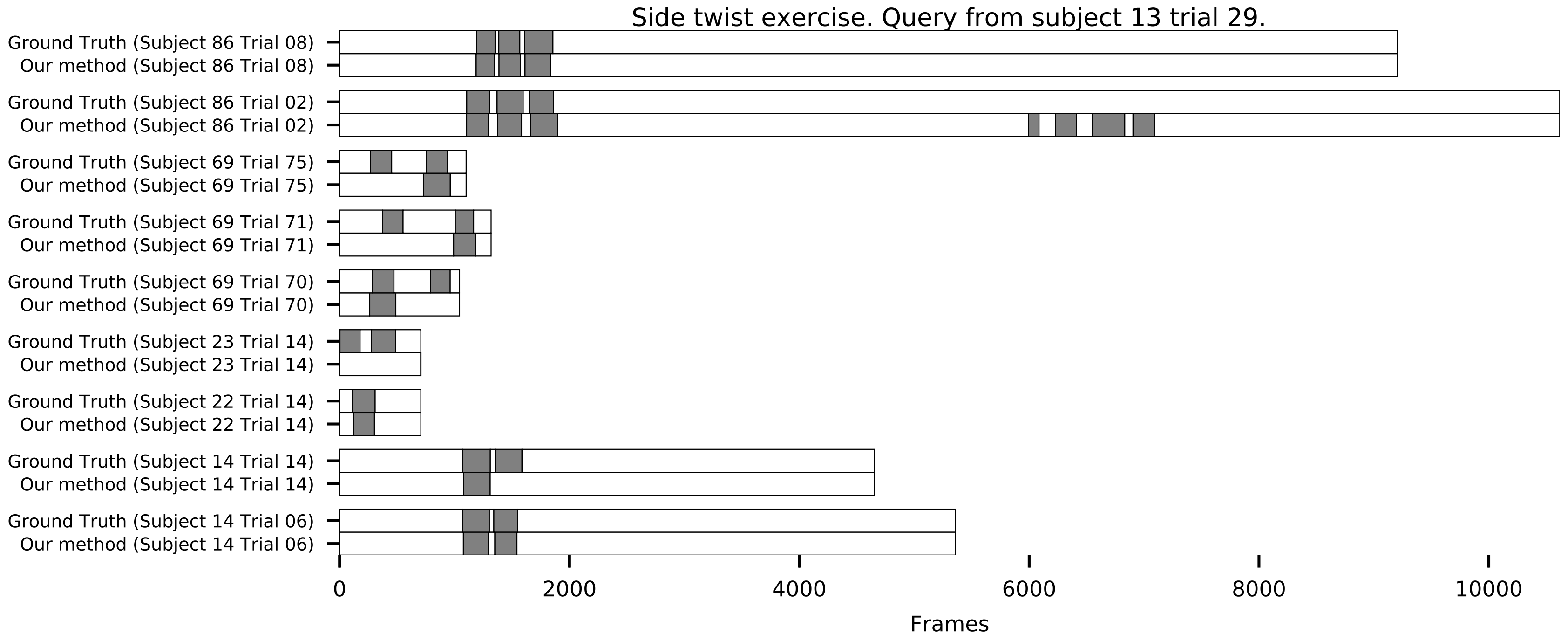


Figure 5.9: Segmentation performance of squat exercise using query from subject 13.

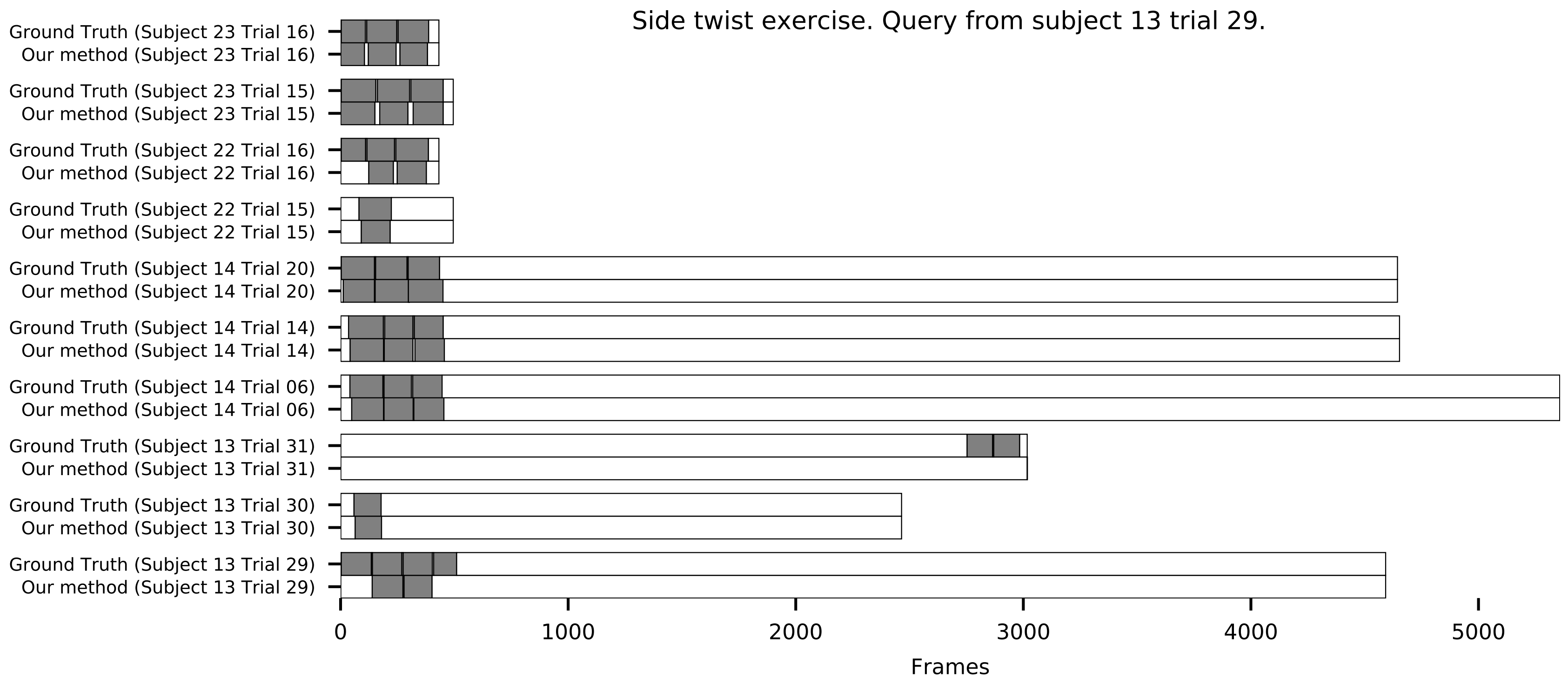
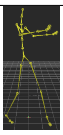
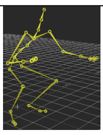
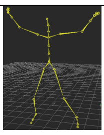


Figure 5.10: Segmentation performance of jumping jack exercise using query from subject 86.

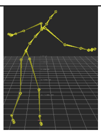
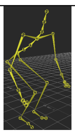
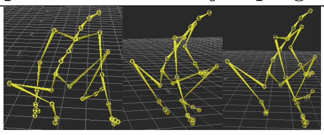
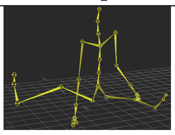
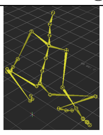
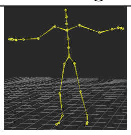
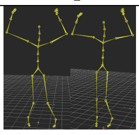
Table 5.2: Summary of CMU segmentation results using a single exemplar repetition from a single subject.

Exercise	**No. of Candidate Subjects**	**No. of GT Segments**	**TP Segments**	**FN Segments**	**FP Segments**	**Exemplar Sub_Trial**
Side Twist	1	4	2	2	0	 13_29
Squat	5	19	13	6	4	 13_29
Jumping Jacks	4	26	21	5	0	 86_05

Figures 5.8, 5.9 and 5.10 show the segmentation performance on the CMU database. The grey bars represent the segmentation of a repetition and white spaces represent no repetition. The bars are in pairs with the ground truth segmentations at the top and the algorithmic segmentations on the bottom. Most of the trials contained more than one exercise/action which occurred during the large white spaces in the bars. The algorithm was tested on all frames of the trials to test the algorithm on its ability to avoid false positives. Where the ground truth grey bars do not have an algorithmic grey bar below, this is a false negative. Likewise, algorithmic grey bars without a ground truth grey bar above is a false positive.

Table 5.2 summarises the results of the segmentations. The jumping jacks, squat and side twist exercises achieved 21/26, 13/19 and 2/4 correct segmentations from 4, 5 and 1 candidate subjects respectively.

Table 5.3: Reasons for failures of CMU segmentations using a single exemplar repetition from a single subject.

Exercise Subject_Trial	**Failure**	**Comment**
Side Twist 14_14	False Negative	 Side twists are performed with arm extension and leaning.
Squat 86_02	False Positive	 When performing a jump on the spot, a squat action is performed before jumping.
Squat 69_70, 69_71, 69_75	False Negative	 Tracking is lost at the knees and feet joints, causing them to rise up and align along the X axis.
Squat 23_14	False Negative	 Tracking loss of the leg joints.
Squat 14_14	False Negative	 Tracking loss of knee joints.
Jumping Jacks 22_16	False Negative	 Recording starts after the first repetition has begun, subsequent repetitions are correctly segmented.
Jumping Jacks 13_29, 13_31	False Negative	 The repetitions are performed incorrectly, with the legs moving together as the arms are raised.

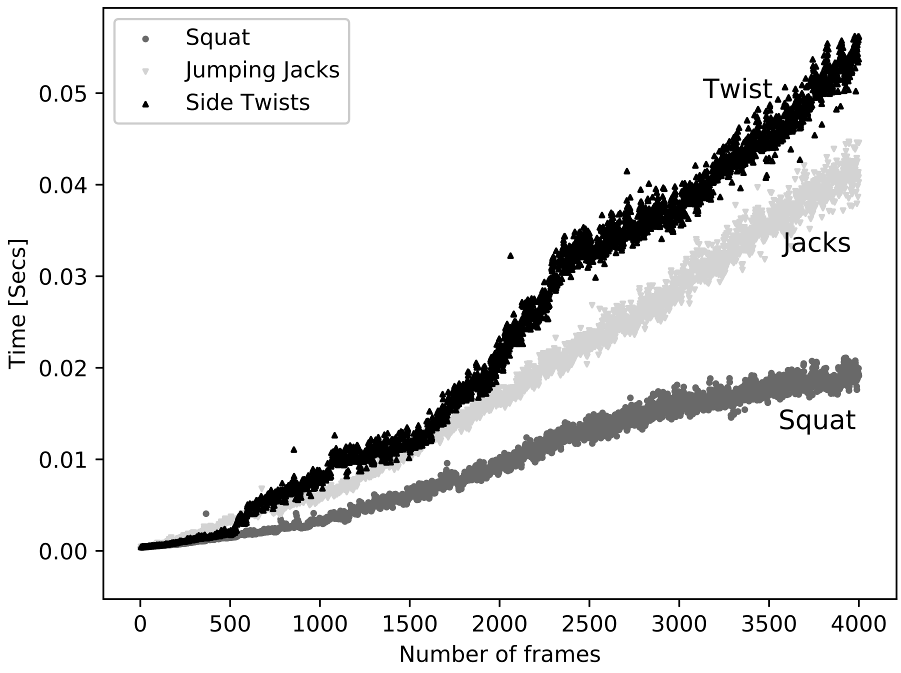


Figure 5.11: Execution performance of the segmentation algorithm when finding the latest repetition. This includes: pre-processing the motion features, feature extraction, finding the optimal warp path and segmentation rejection/confirmation using the highly ranked motion features.

It should be noted that the segmentation algorithm was not the cause of most of the failures, as shown in table 5.3. The reason for the false negative segments on the squat exercise was poor joint tracking e.g. squat 14_14, 23_14 and 69_70/71/75. Jumping jacks and side twist exercises failed due to the exercise being incorrectly performed e.g. side twist 14_14 and jumping jacks 13_29/31, and one case of the recording starting after the repetition had begun e.g. jumping jacks 22_16. One set of false positive segments occurred on the squat exercise, 86_02, where the subject performed a squatting action before a jump. This is the only case of false positive segments although many other exercises and actions were performed during the trials. This demonstrates that the segmentation algorithm can achieve suitable results from a single exemplar repetition.

5.3.2 Execution performance evaluation

Figure 5.11 shows the execution performance of the whole segmentation algorithm segmenting the latest repetition of an exercise as the number of frames increases. The exercises squat, jumping jacks and side twists were evaluated. The DTW query of each exercise consisted of 3, 3 and 6 segment queries respectively.

The time complexity for the worst case scenario of the whole segmentation algorithm is $O(ml^2)$ as multiple subsequences are evaluated using SDTW. The function GetWarpPath in algorithm 1 on line 12 performs DTW which has a time complexity of $O(mn)$ and GetTotalCost on line 13 has a time complexity of $O(m)$.

The runtime performance on real-world examples of CMU exercises is depicted in Figure 5.11. The processing time taken does not increase as much as the $O(ml^2)$ term would suggest, this is because a subsequence must meet a minimum scale to be considered for segmentation, and the DTW alignment is abandoned when the total cost goes above the lowest total cost so far.

Considering a worst case scenario of segmenting stroke patient repetitions, where the movements are often slow. The longest repetition performed by a stroke patient was ≈13 seconds. Thus, segmenting the latest repetition given a 20 second window consisting of 600 frames, from a device with a 30Hz capture rate, would take ≈5ms given Figure 5.11. Although the algorithm can be run more infrequently, the processing time ≈5ms is well within the real-time capture rate of 33ms. If an exercise repetition was to be performed beyond the window size then the repetition will likely be missed or segment the start of the exercise late. To resolve this every n^{th} frame could be dropped until it is able to process the repetition in real-time at the expense of segmentation accuracy, however, the evaluation shows the algorithm maintained real-time processing well beyond the sequence length of real exercise repetitions.

The performance evaluation took place on a Intel Core i7 4790K at 4.00GHz using a single threaded Python implementation. Note that this execution performance evaluation is an example of the worst-case performance as certain optimisations have been left out such as: DTW window constraints; online normalisation [122]; compilation to native machine code; and clearing the buffer when a repetition is detected. But for the purposes of a real-world implementation of a system designed to segment exercises in real-time, it can be seen that even an unoptimised implementation is sufficiently fast.

5.3.3 Detailed evaluation on rehabilitation exercises

Evaluation technique

The dataset used for evaluation contains three exercises performed by four stroke patients undergoing rehabilitation at home. Ethical approval for this study was obtained via the University's ethical approval board. The three exercises within the dataset are arm to side, arm to front and twist exercise. The exercises were taken from the Graded Repetitive Arm Supplementary Program (GRASP) manual [124], an exercise program developed for stroke patients. Each subject performed three repetitions of each exercise. For evaluation, the data passed to the segmentation algorithm simulates a real-time implementation, i.e. the algorithm receives data frame by frame and cannot see future data. For each exercise being evaluated, joint data of all exercises were included in the evaluation to ensure the algorithm was robust to false positives. We follow the methodology presented in [60] as they evaluated a similar clinical population as ours, thus allowing an almost direct comparison of the effectiveness of both algorithms. The evaluation methodology is as follows:

1. Simulate a real-time implementation by sending observation motion data

frame by frame to the segmentation algorithm. Once the full observation has been processed, a list of segments is returned. This approach tests the segmentation algorithm for early segmentation.

2. The following definitions are used in the evaluation of the segmentation algorithm's accuracy:
   (a) Ground truth segment (GT): Ground truth segment envelopes are added to the observations to represent the ground truth of the start and end of an exercise repetition. A GT has an envelope of acceptability with varying sizes as the start and end of an exercise repetition is often ambiguous, as mentioned in [60]. For this evaluation, video data was used to determine the start and end of an exercise, timestamps were used to temporally align the video data and joint data.

   (b) Time Error (TE): Time error is a variable that increases the temporal width of the GT envelopes by X time. This is to allow algorithmic segmentations that were close to a GT without a TE to be considered a true positive segment.

   (c) True positive segment (TP): If an algorithmic segment is within the TE of a GT then it is classed as a TP; e.g. if TE is set to 1 second and an algorithmic segment is within 1 second of a GT, then the number of TPs is incremented by 1.

   (d) False positive segment (FP): If an algorithmic segment exists where there should not be a segment; e.g. an algorithmic segment is not within a TE of a GT, then the number of false positive segments is incremented by 1.

   (e) False negative segment (FN): If no algorithmic segment exists where there should be one; e.g. no algorithmic segment is within the TE of a GT, then the number of false negative segments is incremented by 1.

Note that if an algorithmic segment is just outside of the TE envelope to a TP segment then the number of FP and FN segments are incremented by one. Algorithmic segments that represent the start of a repetition cannot be classed as TP if they fall within the TE of a GT representing the end of a repetition. Similarly, algorithmic segments that represent the end of a repetition cannot be classed as TP if they fall within the TE of a GT representing the beginning of a repetition.

Table 5.4: Kinect Exercise 1

TE (s)	TP (%)	FP (%)	FN (%)	ESS		LSS		EES		LES	
				%	MAE (m)	%	MAE (m)	%	MAE (m)	%	MAE (m)
0.0	54.2	54.2	45.8	0.0	0.0000	0.0	0.0	0.0	0.000	0.0	0.0000
0.1	75.0	33.3	25.0	11.1	0.0016	0.0	0.0	11.1	0.012	5.6	0.0009
0.2	83.3	25.0	16.7	10.0	0.0016	0.0	0.0	15.0	0.022	10.0	0.0029
0.3	87.5	20.8	12.5	14.3	0.0021	0.0	0.0	14.3	0.022	9.5	0.0029
0.4	91.7	16.7	8.3	13.6	0.0021	0.0	0.0	18.2	0.042	9.1	0.0029
0.5	95.8	12.5	4.2	13.0	0.0021	0.0	0.0	21.7	0.048	8.7	0.0029

Table 5.5: Kinect Exercise 2

TE (s)	TP (%)	FP (%)	FN (%)	ESS		LSS		EES		LES	
				%	MAE (m)	%	MAE (m)	%	MAE (m)	%	MAE (m)
0.0	75.0	25.0	17.9	0.0	0.0000	0.0	0.0	0.0	0.0	0.0	0.0000
0.1	82.1	17.9	10.7	4.3	0.0021	0.0	0.0	0.0	0.0	4.3	0.0017
0.2	85.7	14.3	7.1	4.2	0.0021	0.0	0.0	0.0	0.0	8.3	0.0020
0.3	89.3	10.7	3.6	4.0	0.0021	0.0	0.0	0.0	0.0	12.0	0.0180
0.4	89.3	10.7	3.6	4.0	0.0021	0.0	0.0	0.0	0.0	12.0	0.0180
0.5	89.3	10.7	3.6	4.0	0.0021	0.0	0.0	0.0	0.0	12.0	0.0180

Table 5.6: Kinect Exercise 3

TE (s)	TP (%)	FP (%)	FN (%)	ESS		LSS		EES		LES	
				%	MAE (m)	%	MAE (m)	%	MAE (m)	%	MAE (m)
0.0	45.5	45.5	45.5	0.0	0.0	0.0	0.000	0.0	0.000	0.0	0.000
0.1	59.1	31.8	31.8	0.0	0.0	15.4	0.083	0.0	0.000	7.7	0.037
0.2	81.8	9.1	9.1	0.0	0.0	22.2	0.130	5.6	0.018	16.7	0.120
0.3	86.4	4.5	4.5	0.0	0.0	26.3	0.160	5.3	0.018	15.8	0.120
0.4	90.9	0.0	0.0	0.0	0.0	25.0	0.160	10.0	0.046	15.0	0.120
0.5	90.9	0.0	0.0	0.0	0.0	25.0	0.160	10.0	0.046	15.0	0.120

Table 5.7: Qualisys Exercise 1

TE (s)	TP (%)	FP (%)	FN (%)	ESS		LSS		EES		LES	
				%	MAE (m)	%	MAE (m)	%	MAE (m)	%	MAE (m)
0.0	81.0	19.0	19.0	0.0	0.0	0.0	0.0000	0.0	0.0	0.0	0.000
0.1	88.1	11.9	11.9	0.0	0.0	2.7	0.0037	0.0	0.0	5.4	0.044
0.2	95.2	4.8	4.8	0.0	0.0	2.5	0.0037	0.0	0.0	12.5	0.043
0.3	97.6	2.4	2.4	0.0	0.0	2.4	0.0037	0.0	0.0	14.6	0.045
0.4	97.6	2.4	2.4	0.0	0.0	2.4	0.0037	0.0	0.0	14.6	0.045
0.5	100.0	0.0	0.0	0.0	0.0	2.4	0.0037	0.0	0.0	16.7	0.048

Table 5.8: Qualisys Exercise 2

TE (s)	TP (%)	FP (%)	FN (%)	ESS		LSS		EES		LES	
				%	MAE (m)	%	MAE (m)	%	MAE (m)	%	MAE (m)
0.0	45.0	55.0	55.0	0.0	0.000	0.0	0.000	0.0	0.000	0.0	0.000
0.1	80.0	20.0	20.0	12.5	0.027	12.5	0.024	6.2	0.023	12.5	0.044
0.2	100.0	0.0	0.0	10.0	0.027	10.0	0.024	5.0	0.023	30.0	0.054
0.3	100.0	0.0	0.0	10.0	0.027	10.0	0.024	5.0	0.023	30.0	0.054
0.4	100.0	0.0	0.0	10.0	0.027	10.0	0.024	5.0	0.023	30.0	0.054
0.5	100.0	0.0	0.0	10.0	0.027	10.0	0.024	5.0	0.023	30.0	0.054

Table 5.9: Qualisys Exercise 3

TE (s)	TP (%)	FP (%)	FN (%)	ESS		LSS		EES		LES	
				%	MAE (m)	%	MAE (m)	%	MAE (m)	%	MAE (m)
0.0	65.0	35.0	35.0	0.0	0.0	0.0	0.00	0.0	0.00	0.0	0.00
0.1	95.0	5.0	5.0	0.0	0.0	21.1	0.11	10.5	0.06	0.0	0.00
0.2	100.0	0.0	0.0	0.0	0.0	20.0	0.11	10.0	0.06	5.0	0.03
0.3	100.0	0.0	0.0	0.0	0.0	20.0	0.11	10.0	0.06	5.0	0.03
0.4	100.0	0.0	0.0	0.0	0.0	20.0	0.11	10.0	0.06	5.0	0.03
0.5	100.0	0.0	0.0	0.0	0.0	20.0	0.11	10.0	0.06	5.0	0.03

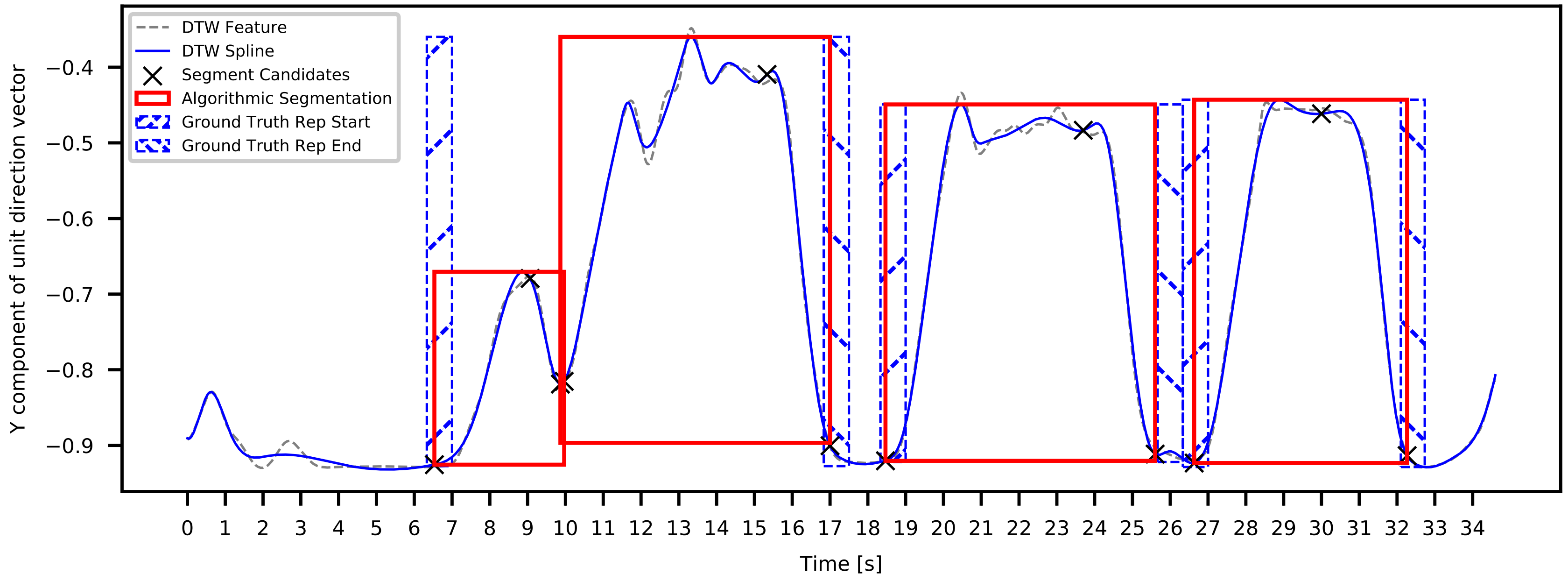


Figure 5.12: Plot showing the algorithmic segmentations of a stroke patient performing the Arm to Side exercise taken from GRASP, a stroke rehabilitation manual.

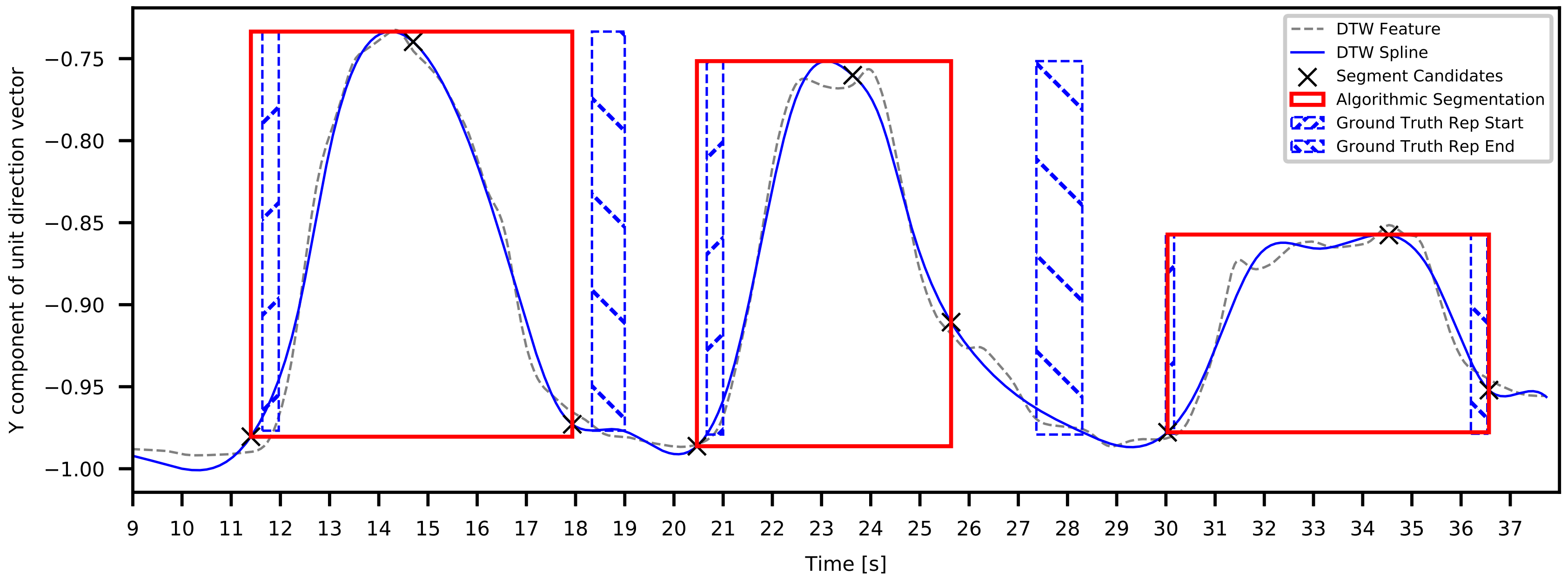


Figure 5.13: Plot showing the algorithmic segmentations of a stroke patient performing the Arm to Front exercise taken from GRASP, a stroke rehabilitation manual.

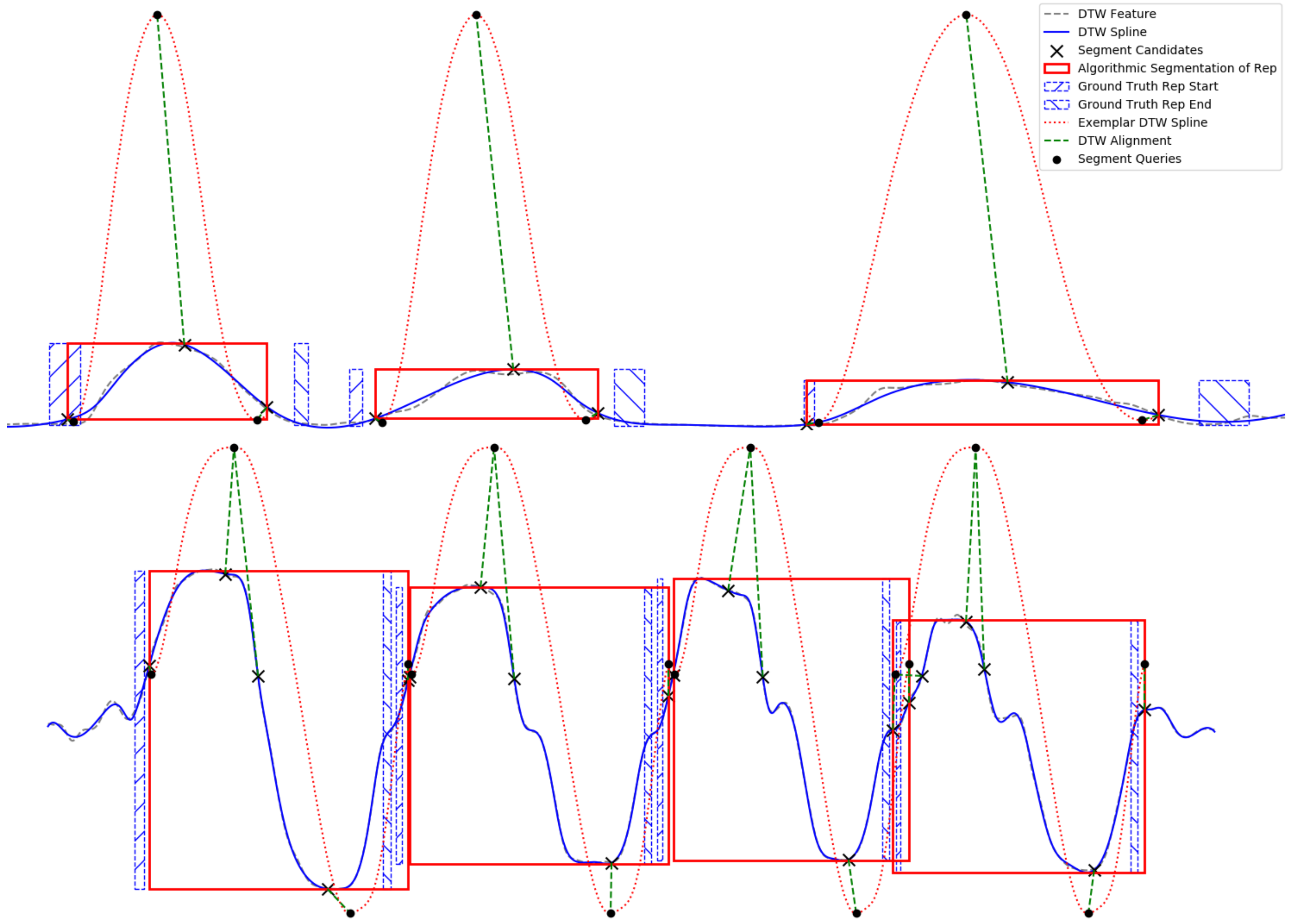


Figure 5.14: Plots showing the algorithmic segmentations and DTW alignments of a stroke patient performing the Arm to Side and Twist exercise taken from GRASP, a stroke rehabilitation manual. The exemplar repetition was performed by a healthy subject.

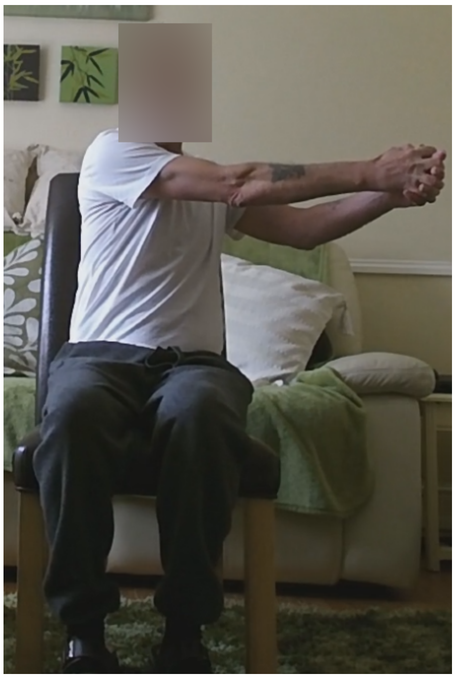


Figure 5.15: A stroke patient performing the twist exercise.

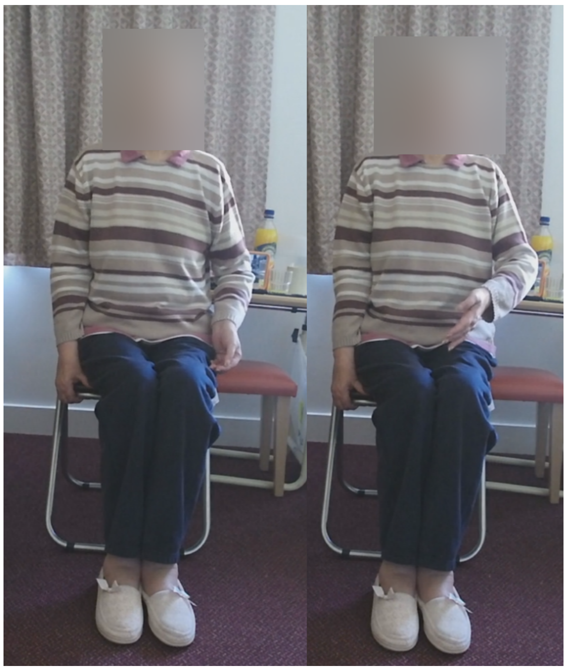


Figure 5.16: A stroke patient performing the arm to front exercise.

Segmentation performance on rehabilitation exercises

Tables 5.4, 5.5, 5.6, 5.7, 5.8, 5.9 show the performance of the segmentation algorithm at different Time Errors (TE) starting at 0 seconds and incrementing by 0.1 up to 0.5 seconds. The results of the segments are represented as a percentage of the total GT. The mean average error (MAE) is displayed, where error is the minimum difference in the algorithmic segment direction to the GT segment direction.

The patient dataset we have evaluated is challenging due to the common low range of motion, instability and variance of the patient movements, as shown in Figures 5.15 and 5.16. The accuracy of the joint positions can be unreliable due to occlusion and contain a lot of jitter.

Tables 5.4, 5.5, 5.6 shows the performance of the segmentation algorithm on the Kinect patient dataset, where it manages to achieve correct segmentation for 90% of the data with a time error of ≈0.3 seconds. There are more false positives in the arm to side and arm to front exercises than the twist exercise. This is due to the similar arm movements performed during these exercises. The MAE of all exercises are negligible which suggests a time error of at least 0.5 seconds and likely larger are acceptable.

Tables 5.7, 5.8, 5.9 show the performance of the segmentation algorithm segmenting joint exercise data of healthy participants performing the same exercises captured by Qualisys, a marker-based motion capture system with a reported sub-millimetre accuracy [125]. The algorithm achieves 100% correct segmentation on all exercises within 0.5 seconds. The MAE for all exercises at different time errors is negligible.

Figure 5.16 depicts a patient performing the arm to front exercise with the accompanying plot depicted in Figure 5.13. The left image in Figure 5.16 shows the patient's starting position; this is captured at ≈30 seconds in Figure 5.13. The right image in Figure 5.16 shows the patient's arm extended to the highest achievable height, captured at ≈34 seconds in Figure 5.13. This demonstrates the robustness of the algorithm as the exercise query expects the arm to be extended to 90 degrees in the sagittal plane. Each of these repetitions has a low range of motion but still achieves good segmentation. Note that at ≈25 seconds the algorithmic segment is early to segment, due to a segment candidate being produced as the arm's velocity slows down. The ground truth repetition start and end rectangles have different widths because the start and end of a repetition is often ambiguous [60]; faster and smoother movements usually result in a smaller ground truth rectangle.

Figure 5.15 shows a patient performing the twist exericse. Smoother motions and a larger range of motion are generally found in this exercise as the paretic limb is grasped with the healthy limb.

Figure 5.12 plots the direction of the arm in the Y axis for the arm to side exercise, at ≈10 seconds the algorithm incorrectly segments the repetition as the patient's arm dips down during the exercise.

Figure 5.14 shows the alignment between the candidate and query sequences for exercises arm to front (top) and twist (bottom). In the top image the range of motion of the candidate is lower than the query but DTW still finds a suitable alignment.

The results show that even for people with limited movement, we have achieved good segmentation accuracy of exercise repetitions. All but one of the segmentation algorithms highlighted in the literature review have tested their algorithms only on healthy participants [58, 61–63, 66–70]. Lin and Kulic [60] tested their algorithm on 4 total joint replacement patients undergoing lower-body rehabilitation and they achieved 79% segmentation accuracy whereas our approach achieved 86.4-89.3% accuracy within the same 0.3 seconds time error threshold. We also found the MAE of late and early segmentations to be negligible even with a 0.5 second TE, which gave a correct segmentation percentage of 89.3%-95.8%. They mention that "DTW provides an accurate method of segmentation that is robust against temporal variations, but is too computationally expensive to be employed on-line.", however we have presented a real-time DTW segmentation algorithm. Chaun-Jun, et al. [62] tested their algorithm on rehabilitation exercises but with healthy participants. Due to the variability of stroke patient movements, it is important that segmentation algorithms are tested on real clinical data of rehabilitation exercises.

Table 5.10: Parameter evaluation results presenting the top 5 parameter combinations.

Row	F_1	TP	FP	ST	DM	DB
1	0.77	0.85	0.36	0.1	0.1	0.3
2	0.77	0.85	0.36	0.2	0.1	0.3
3	0.75	0.85	0.4	0.3	0.1	0.3
4	0.75	0.85	0.4	0.1	0	0.4
5	0.75	0.85	0.4	0.2	0	0.4

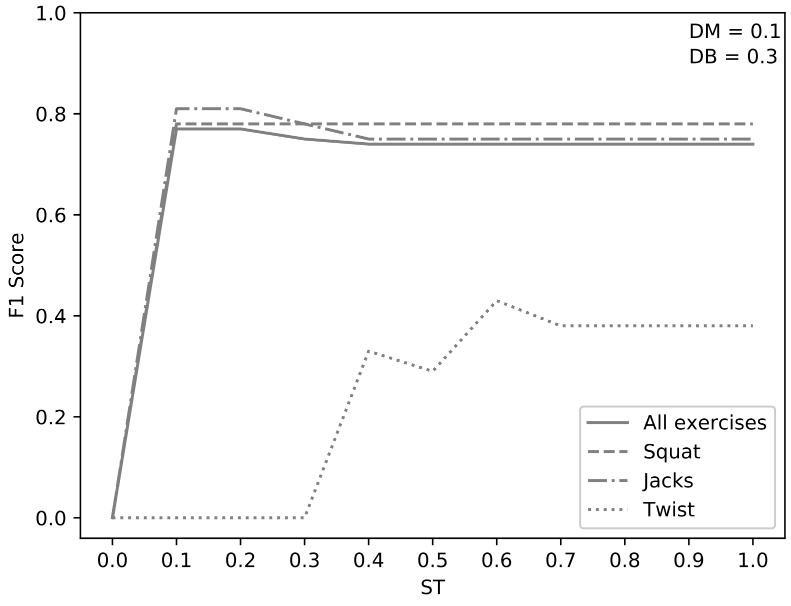


Figure 5.17: Plot showing the change in the *F_1* score with respect to the ST parameter and given the best (highest *F_1* score) DM and DB parameter combination.

5.3.4 Parameter evaluation

The parameters described in section 5.2.6 were set via the following process. The range of motion of stroke patient repetitions were measured and compared with healthy subjects to establish an appropriate baseline. It was found that setting the minimum scale percentage to 0.08, i.e. 8%, was required to ensure that stroke patient repetitions with the lowest range of motion could be adequately segmented. The segmentation threshold (ST) is the maximum value the total cost of a normalised subsequence can be from the normalised query to be considered for segmentation. This total cost is essentially a measure of the abnormality in the movements between the subsequence and query; therefore a total cost below the segmentation threshold

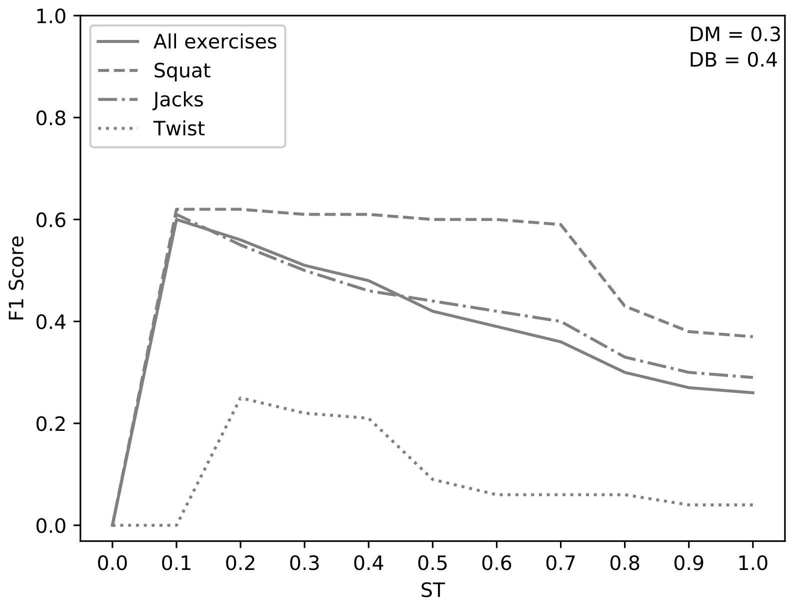


Figure 5.18: Plot showing the change in the *F1* score with respect to the ST parameter and given an average (mean *F1* score) DM and DB parameter combination.

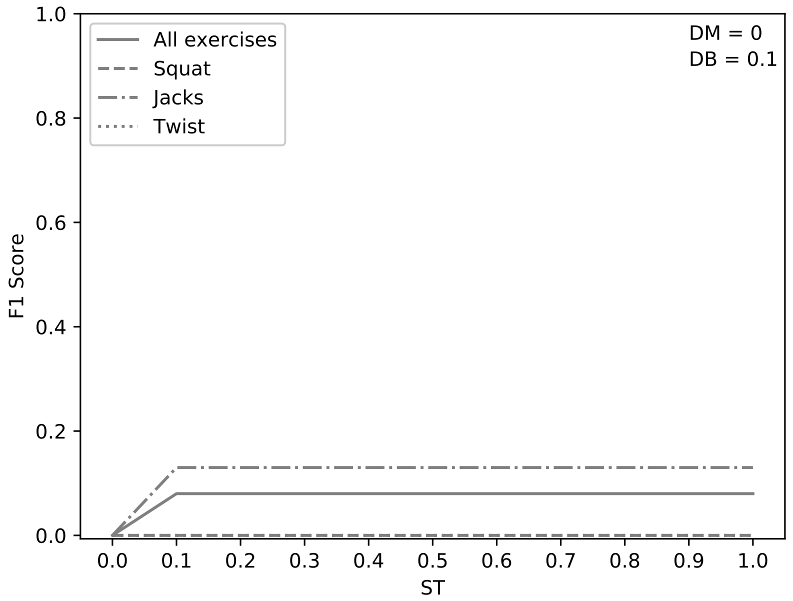


Figure 5.19: Plot showing the change in the *F1* score with respect to the ST parameter and given the worst (lowest *F1* score) DM and DB parameter combination (excluding combinations with an *F1* score of zero).

indicates that the patient's movements are an attempted repetition of the exemplar exercise. The distance multiplier (DM) and distance base (DB) adjust the total cost thresholds, as described in equation 5.8, for each of the motion features based on their rate of change, i.e. motion features that exhibit the most movement. The total cost of each motion feature in a subsequence must be lower than their total cost threshold to be considered for segmentation.

A parameter evaluation was performed on parameters ST, DM and DB (described in Section 5.2.6) using grid search with a step size of 0.1 within the range 0 to 1. By evaluating the parameter space using a linear step size we can see how the algorithm's accuracy changes with respect to the changes in the parameter values. Each parameter combination was cross-validated over CMU exercises: jumping jacks, squat and side twist. To determine accuracy we use F_1 (Acc_{F_1}) score presented in [126] and expressed as:

$$\frac{2 \cdot TP}{2 \cdot TP + FN + FP} \tag{5.9}$$

The parameter combinations with the highest F_1 accuracy are presented in Table 5.10. TP and FP segments are presented as percentages of the total GT segments. Given the top five combinations, the ST values range from 0.1 to 0.3, DM range from 0 to 0.1 and DB range from 0.3 to 0.4. To further understand how the F_1 accuracy changes with respect to these parameters, we have graphed three plots (Figures 5.17, 5.18 and 5.19) with each plot showing a representative DM and DB value combination that achieves the best, average and worst accuracy respectively, and plotted the F_1 accuracy with respect to the ST parameter value.

A greater value of ST results in candidate sequences that are less similar to the query sequence being treated as a potential segmentation, but with suitable DM and DB values potential segmentations that would be considered as a FP are mostly rejected. This can be seen in Figure 5.17 whereby the F_1 accuracy does not decrease much as ST increases to a value of 1. As greater values are used for DM and DB the number of FP segments increases resulting in a lower F_1 accuracy, as can be seen in Figure 5.18. Using very low values for DM and DB results in fewer FP segments but also fewer TP segments and thus a low F_1 accuracy, as can be seen in Figure 5.19.

5.4 Summary

We have presented an algorithm for segmenting exercise repetitions in real-time. This approach addresses the limitations of previous approaches in that it requires only a single exemplar and has shown robustness to repetitions with low range of motion, instability in the movements and noise in the sensor data. The algorithm was evaluated on 10 subjects performing 3 exercises from a publicly available dataset (CMU) and we showed that it was capable of segmenting the repetitions. Further evaluation was performed on our own datasets of a healthy population and a stroke population performing stroke rehabilitation exercises. We showed that the algorithm correctly segments all the healthy population exercise repetitions within 0.5 seconds and the stroke patient exercise repetitions to 90% TP segments within 0.3-0.4 seconds. Our next step is to develop a pose estimation algorithm designed for clinical use to combine with this new segmentation algorithm, in order to enable accurate real-time feedback for stroke patients undergoing rehabilitation.

6 Conclusions and future work

6.1 Conclusions

This thesis has contributed the following:

- A quantitative and clinical evaluation of a state-of-the-art pose estimation algorithm from a compact and cost-effective depth sensor (Kinect v2). To determine if the accuracy of the joint position estimations are good enough for correctly assessing patient exercises and what the limitations are. The findings and highlights were:
  - The joint position estimations are mostly inadequate for correctly assessing stroke rehabilitation exercises.
  - A quantitative evaluation on its own is not sufficient for determining the

  suitability of joint position estimations for clinical applications, a clinical evaluation is required as each joint for different exercises does not have the same level of importance.

  - Joint rotations are required for a more in-depth and correct assessment.
  - Exploiting temporal information may ameliorate joint tracking issues caused by occlusion.
  - Joint position estimations should be constrained within the anatomical limitations of the human body.
  - Clinicians should be consulted to select exercises that provide the pose estimation algorithm with fewer challenges but still achieve correct functional improvements for the patient.

- Evaluated the inter-rater agreement between the physiotherapists' clinical evaluation. The findings and highlights were:
  - The results indicate the physiotherapists had a low level of inter-rater agreement.
  - Introducing objective measures to the grading criterion may help guide raters.
  - Further research required to determine factors that may have contributed to the low agreement as the medium used for assessment, insufficient training on the new task and/or an inadequate grading criterion.

- Proposed a segmentation algorithm for segmenting exercise repetitions that requires only a single exemplar repetition from a healthy subject to segment subsequent repetitions from other subjects, including those with impaired mobility. This algorithm was evaluated on our own stroke rehabilitation dataset of stroke patients performing upper-body rehabilitation exercises, and a publicly available dataset CMU [73]. The findings and highlights were:

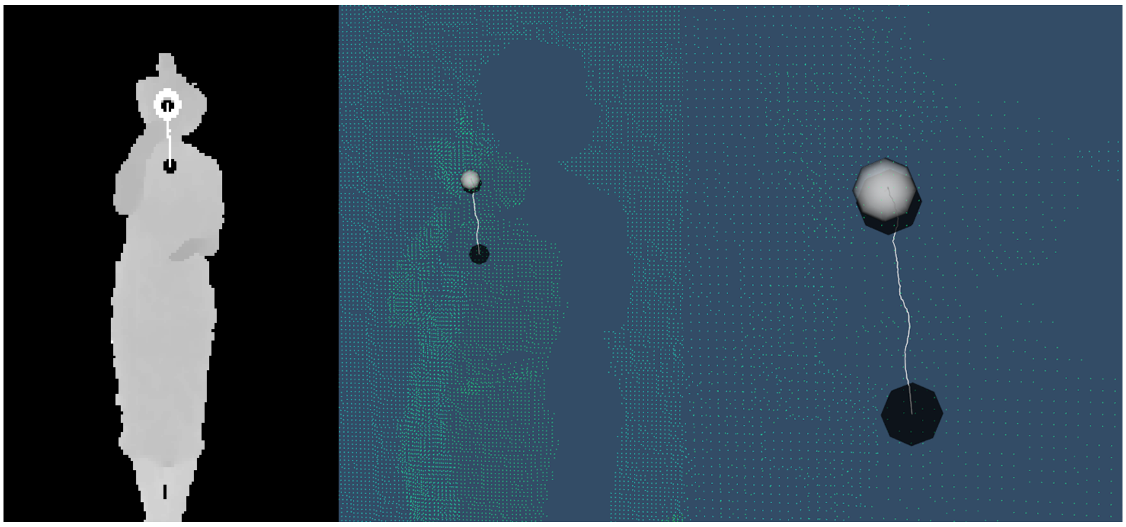


Figure 6.1: Visualisation of our pose estimation algorithm currently in investigation and design stage. Left image is the 2D depth map representation, middle and right images are the 3D point cloud visualisation.

- 90% of the stroke patient exercise repetitions were correctly segmented within 0.3-0.4 seconds of the ground truth envelope.
- The parameter combinations that achieved a high accuracy were robust across different exercises.
- The algorithm is capable of being run in real-time on a modest computer, achieving segmentation of a 20 second window at 30Hz in $\approx 5ms$.

6.2 Future work

6.2.1 Pose estimation algorithm

Based on our research on the accuracy and limitations of current state-of-the-art pose estimation algorithms (PEA) intended for clinical use, we are designing a PEA [1] that overcomes these limitations. An early prototype can be seen in Figure 6.1 whereby the centre of the head is being tracked. Like Kinect V2's pose estimation

[1]See appendices for code snippets of the pose estimation algorithm program.

algorithm [15] a decision tree is trained for each joint position but instead of classifying each pixel in the depth map into body part categories, the tree is trained on the probability distribution towards a ground truth joint position. The white line represents the walk through the depth map to the white sphere that represents the final predicted joint position for the centre head joint, the two black spheres represent the ground truth SpineMid (bottom) and centre head (top) joints. Spatial and temporal constraints are then intended to ensure anatomically correct poses and realistic motion.

6.2.2 Automatically assessing patient exercise performance

Replacing conventional assessment criteria, such as the Fugl-Meyer Assessment [82], with automatic quantifiable measures to evaluate physical performance. By tracking patient joint positions and possibly joint rotations, the assessment of physical performance can be quantified. Research should investigate suitable performance metrics that can be use to baseline a patient's physical performance and track improvements, such as range of motion, deviation from desired motion, stability and the severity of compensatory movements.

6.2.3 Maximising adherence

Technology-assisted stroke rehabilitation interventions often focus on the physical aspects of recovery, but reasons for low adherence are likely a result of mental challenges that are faced by patients experiencing the sudden and debilitating conditions from stroke. Further research should investigate these mental challenges and design technology-assisted solutions that can maximise adherence, such as developing game-based rehabilitation sessions that provide patients with an engaging and suitably challenged exercise program. Positive forms of feedback should be investigated such as displaying the patients functional improvements.

Bibliography

[1] J. Sarsfield, D. Brown, N. Sherkat, C. Langensiepen, J. Lewis, M. Taheri, C. McCollin, C. Barnett, L. Selwood, P. Standen, P. Logan, C. Simcox, C. Killick, and E. Hughes, "Clinical Assessment of Depth Sensor Based Pose Estimation Algorithms for Technology Supervised Rehabilitation Applications," *International Journal of Medical Informatics*, nov 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1386505618312759

[2] J. Sarsfield, D. Brown, N. Sherkat, C. Langensiepen, J. Lewis, M. Taheri, L. Selwood, P. Standen, and P. Logan, "Segmentation of Exercise Repetitions Enabling Real-Time Patient Analysis and Feedback Using a Single Exemplar," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8688440

[3] J. Sarsfield, D. Brown, C. Langensiepen, N. Sherkat, J. Lewis, and P. Standen, "Reducing Clinical Subjective Discrepancies in Evaluation of Clinical Technology using Objective Measures," in *12th International Conference on Disability, Virtual Reality & Associated Technologies*, 2018.

[4] J. E. Harris, J. J. Eng, W. C. Miller, and A. S. Dawson, "A self-administered graded repetitive arm supplementary program (GRASP) improves arm function during inpatient stroke rehabilitation: A multi-site randomized controlled trial," *Stroke*, vol. 40, no. 6, pp. 2123–2128, 2009. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/19359633

[5] E. S. Lawrence, C. Coshall, R. Dundas, J. Stewart, A. G. Rudd, R. Howard, and C. D. Wolfe, "Estimates of the prevalence of acute stroke impairments and disability in a multiethnic population." *Stroke*, vol. 32, no. 6, pp. 1279–84, jun 2001. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/11387487

[6] C. J. L. Murray and Vos, "Disability-adjusted life years (DALYs) for 291 diseases and injuries in 21 regions, 1990-2010: A systematic analysis for the Global Burden of Disease Study 2010," *The Lancet*, vol. 380, no. 9859, pp. 2197–2223, 2012.

[7] M. Shaughnessy, B. M. Resnick, and R. F. Macko, "Testing a model of post-stroke exercise behavior." pp. 15–21, 2006.

[8] P. Langhorne, F. Coupar, and A. Pollock, "Motor recovery after stroke: a systematic review," *The Lancet Neurology*, vol. 8, no. 8, pp. 741–754, 2009. [Online]. Available: http://dx.doi.org/10.1016/S1474-4422(09)70150-4

[9] et al. Selzer, Michael, *Textbook of Neural Repair and Rehabilitation: Volume 2, Medical Neurorehabilitation.* Cambridge University Press, 2006.

[10] K. K. Miller, R. E. Porter, E. DeBaun-Sprague, M. Van Puymbroeck, and A. A. Schmid, "Exercise after stroke: Patient adherence and beliefs after discharge from rehabilitation," *Topics in Stroke Rehabilitation*, vol. 24, no. 2, pp. 142–148, 2017.

[11] "NHS Midlands and East Stroke Services Specification," Tech. Rep. [Online]. Available: https://www.england.nhs.uk/mids-east/wp-content/uploads/sites/7/2018/03/final-stroke-spec.pdf

[12] S. Pundik, J. P. McCabe, K. Hrovat, A. E. Fredrickson, C. Tatsuoka, I. J. Feng, and J. J. Daly, "Recovery of post stroke proximal arm function, driven by complex neuroplastic bilateral brain activation patterns and predicted by baseline motor dysfunction severity," *Frontiers in*

Human Neuroscience, vol. 9, no. July, pp. 1–13, 2015. [Online]. Available: http://journal.frontiersin.org/Article/10.3389/fnhum.2015.00394/abstract

[13] M. F. Levin, J. A. Kleim, and S. L. Wolf, "What do motor "recovery" and "compensation"mean in patients following stroke?" *Neurorehabilitation and Neural Repair*, vol. 23, no. 4, pp. 313–319, 2009. [Online]. Available: http://nnr.sagepub.com/content/early/2008/12/31/1545968308328727.short{%}5Cnfiles/279/1545968308328727.html

[14] "NHS England: Day Services Clinical Guidance - Stroke." [Online]. Available: https://www.england.nhs.uk/south/wp-content/uploads/sites/6/2017/02/stroke-guidance-7ds.pdf

[15] J. Shotton and E. al., "Real-time human pose recognitiom in parts from single depth images," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).* IEEE, 2011.

[16] A. Haque, B. Peng, Z. Luo, A. Alahi, S. Yeung, and L. Fei-Fei, "Viewpoint Invariant 3D Human Pose Estimation with Recurrent Error Feedback," *arXiv.org*, 2016. [Online]. Available: http://arxiv.org/abs/1603.07076

[17] C. Keskin, F. Kiraç, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1228–1234, 2011.

[18] Y. Wu and Q. Ji, "Facial Landmark Detection: A Literature Survey," *International Journal of Computer Vision*, pp. 1–28, 2018.

[19] S. Gokturk, H. Yalcin, and C. Bamji, "A Time-Of-Flight Depth Sensor - System Description, Issues and Solutions," p. 35, 2004.

[20] BOYER K. L. and KAK A. C., "Color-Encoded Structured Light for Rapid Active Ranging," *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 1987.

[21] S. Collins, “Stereoscopic Depth Perception,” *Photogrammetric Engineering*, vol. 47(1), pp. 45–52, 1981.

[22] P. J. Standen, D. J. Brown, S. Battersby, M. Walker, L. Connell, A. Richardson, F. Platts, K. Threapleton, and A. Burton, “A study to evaluate a low cost virtual reality system for home based rehabilitation of the upper limb following stroke,” *Virtual Reality*, pp. 139–146, 2010.

[23] Y. J. Chang, S. F. Chen, and J. D. Huang, “A Kinect-based system for physical rehabilitation: A pilot study for young adults with motor disabilities,” *Research in Developmental Disabilities*, vol. 32, no. 6, pp. 2566–2570, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.ridd.2011.07.002

[24] “Mindmotion GO.” [Online]. Available: https://www.mindmotionweb.com/mindmotion-go/

[25] S. Patel, R. Hughes, and T. Hester, “A novel approach to monitor rehabilitation outcomes in stroke survivors using wearable technology,” *Proceedings of the . . .*, 2010. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=5420034

[26] W. H. Chang and Y.-H. Kim, “Robot-assisted Therapy in Stroke Rehabilitation,” *Journal of Stroke*, vol. 15, no. 3, p. 174, 2013. [Online]. Available: http://j-stroke.org/journal/view.php?doi=10.5853/jos.2013.15.3.174

[27] W.-S. Kim, S. Cho, S. H. Park, J.-Y. Lee, S. Kwon, and N.-J. Paik, “A low cost kinect-based virtual rehabilitation system for inpatient rehabilitation of the upper limb in patients with subacute stroke,” *Medicine*, vol. 97, no. 25, p. e11173, 2018. [Online]. Available: http://insights.ovid.com/crossref?an=00005792-201806220-00061

[28] S. Woodman, H. Hiden, M. Turner, S. Dowsland, and P. Watson, “Monitoring

of upper limb rehabilitation and recovery after stroke: An architecture for a cloud-based therapy platform," *Proceedings - 11th IEEE International Conference on eScience, eScience 2015*, pp. 381–390, 2015.

[29] A. Lotfi, C. Langensiepen, and S. Yahaya, "Socially Assistive Robotics: Robot Exercise Trainer for Older Adults," *Technologies*, vol. 6, no. 1, p. 32, 2018. [Online]. Available: http://www.mdpi.com/2227-7080/6/1/32

[30] Y. J. Fan, Y. H. Yin, L. D. Xu, Y. Zeng, and F. Wu, "IoT-based smart rehabilitation system," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1568–1577, 2014.

[31] A. Jaume-I-Capó, P. Martinez-Bueso, B. Moya-Alcover, and J. Varona, "Interactive rehabilitation system for improvement of balance therapies in people with cerebral palsy," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 2, pp. 419–427, 2014.

[32] Z. Saenz-De-Urturi and B. Garcia-Zapirain Soto, "Kinect-based virtual game for the elderly that detects incorrect body postures in real time," *Sensors (Switzerland)*, vol. 16, no. 5, 2016.

[33] A. K. Roy, Y. Soni, and S. Dubey, "Enhancing effectiveness of motor rehabilitation using kinect motion sensing technology," in *2013 IEEE Global Humanitarian Technology Conference: South Asia Satellite (GHTC-SAS).* IEEE, aug 2013, pp. 298–304. [Online]. Available: http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=6629934

[34] "MindMotion Pro." [Online]. Available: https://www.mindmaze.com/mindmotion/

[35] P. Polygerinos, K. C. Galloway, E. Savage, M. Herman, K. O. Donnell, and C. J. Walsh, "Soft Robotic Glove for Hand Rehabilitation and Task Specific

Training," *Nature Biotechnology*, vol. 32, no. 9, pp. 851–851, 2014. [Online]. Available: http://www.nature.com/doifinder/10.1038/nbt0914-851b

[36] M. Borghetti, E. Sardini, and M. Serpelloni, "Sensorized glove for measuring hand finger flexion for rehabilitation purposes," *IEEE Transactions on Instrumentation and Measurement*, vol. 62, no. 12, pp. 3308–3314, 2013.

[37] Y. Mengüç, Y. L. Park, H. Pei, D. Vogt, P. M. Aubin, E. Winchell, L. Fluke, L. Stirling, R. J. Wood, and C. J. Walsh, "Wearable soft sensing suit for human gait measurement," *International Journal of Robotics Research*, vol. 33, no. 14, pp. 1748–1764, 2014.

[38] L. YINBEI, "Singapore's robot exercise coach for the elderly." [Online]. Available: https://www.bbc.co.uk/news/av/world-asia-35149344/singapore-s-robot-exercise-coach-for-the-elderly

[39] M. E. Hussein, M. Torki, M. a. Gowayyed, and M. El-Saban, "Human action recognition using a temporal hierarchy of covariance descriptors on 3D joint locations," *IJCAI International Joint Conference on Artificial Intelligence*, pp. 2466–2472, 2013.

[40] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition," *Journal of Visual Communication and Image Representation*, vol. 25, pp. 24–38, 2014.

[41] R. Vemulapalli, F. Arrate, and R. Chellappa, "Human action recognition by representing 3D skeletons as points in a lie group," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 588–595, 2014.

[42] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "Mining actionlet ensemble for action recognition with depth cameras," *Proceedings of the IEEE Computer Society*

Conference on Computer Vision and Pattern Recognition, pp. 1290–1297, 2012.

[43] J. F. Hu, W. S. Zheng, J. Lai, and J. Zhang, “Jointly Learning Heterogeneous Features for RGB-D Activity Recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 11, pp. 2186–2200, 2017.

[44] L. Xia, C. C. Chen, and J. K. Aggarwal, “View invariant human action recognition using histograms of 3D joints,” *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 20–27, 2012.

[45] A. Fernández-Baena, A. Susín, and X. Lligadas, “Biomechanical validation of upper-body and lower-body joint movements of kinect motion capture data for rehabilitation treatments,” *Proceedings of the 2012 4th International Conference on Intelligent Networking and Collaborative Systems, INCoS 2012*, pp. 656–661, 2012.

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

[47] G. Kurillo, A. Chen, R. Bajcsy, and J. J. Han, “Evaluation of upper extremity reachable workspace using Kinect camera,” *Technology and Health Care*, vol. 21, no. 6, pp. 641–656, 2013. [Online]. Available: http://iospress.metapress.com/content/m640m7k5736611u5/fulltext.pdf

[48] X. Xu and R. W. McGorry, “The validity of the first and second generation Microsoft Kinect™ for identifying joint center locations during static postures,” *Applied Ergonomics*, vol. 49, pp. 47–54, 2015. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0003687015000149

[49] Q. Wang, Q. Wang, and G. Kurillo, "Evaluation of Pose Tracking Accuracy in the First and Second Generations of Microsoft Kinect Evaluation of Pose Tracking Accuracy in the First and Second Generations of Microsoft Kinect," no. November, 2015.

[50] L. F. Yeung, K. C. Cheng, C. H. Fong, W. C. C. Lee, and K.-Y. Tong, "Evaluation of the Microsoft Kinect as a clinical assessment tool of body sway." *Gait & posture*, vol. 40, no. 4, pp. 532–8, 2014. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/25047828

[51] D. Webster and O. Celik, "Experimental evaluation of Microsoft Kinect's accuracy and capture rate for stroke rehabilitation applications," *IEEE Haptics Symposium, HAPTICS*, no. February 2014, pp. 455–460, 2014.

[52] A. Mobini, S. Behzadipour, M. S. Foumani, and M. Saadat Foumani, "Accuracy of Kinect's skeleton tracking for upper body rehabilitation applications." *Disability and rehabilitation. Assistive technology*, vol. 9, no. 4, pp. 1–9, 2013. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/23786360{%}5Cnhttp://informahealthcare.com/idt

[53] E. A. Suma, B. Lange, S. Rizzo, D. Krum, and M. Bolas, "FLEXIBLE ACTION AND ARTICULATED SKELETON TOOLKIT." [Online]. Available: http://projects.ict.usc.edu/mxr/faast/

[54] J. Barth, W. E. De Boer, J. W. Busse, J. L. Hoving, S. Kedzia, R. Couban, K. Fischer, D. Y. Von Allmen, J. Spanjer, and R. Kunz, "Inter-rater agreement in evaluation of disability: Systematic review of reproducibility studies," 2017.

[55] A. E., B. K.L., H. M.A., S. M., R. E.M., and C. M., "Validity of a clinical test of assessing dynamic knee alignment during a single-limb mini squat," *Osteoarthritis and Cartilage*, vol. 18, p. S74, 2010. [Online]. Available: http://www.embase.com/search/results?

subaction=viewrecord{&}from=export{&}id=L70312176{%}5Cnhttp://dx.doi.org/10.1016/S1063-4584(10)60177-X{%}5Cnhttp://mgetit.lib.umich.edu/sfx{_}locater?sid=EMBASE{&}issn=10634584{&}id=doi:10.1016{%}2FS1063-4584{%}2810{%}2960177-X{&}atitle=Val

[56] M. Blackburn, P. van Vliet, and S. Mockett, "People With Stroke Modified Ashworth Scale in the Lower Extremities of Reliability of Measurements Obtained With the," *Phys Ther*, vol. 82, no. 1, pp. 25–34, 2002. [Online]. Available: https://academic.oup.com/ptj/article-abstract/82/1/25/2836978

[57] T. L. Chmielewski, M. J. Hodges, M. Horodyski, M. D. Bishop, B. P. Conrad, and S. M. Tillman, "Investigation of Clinician Agreement in Evaluating Movement Quality During Unilateral Lower Extremity Functional Tasks: A Comparison of 2 Rating Methods," *Journal of Orthopaedic & Sports Physical Therapy*, vol. 37, no. 3, pp. 122–129, 2007. [Online]. Available: http://www.jospt.org/doi/10.2519/jospt.2007.2457

[58] D. Gong, G. Medioni, and X. Zhao, "Structured Time Series Analysis for Human Action Segmentation and Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1414–1427, 2014.

[59] J. Kohlmorgen and S. Lemm, "A Dynamic HMM for On-line Segmentation of Sequential Data," *NIPS - Advances in Neural Information Processing Systems*, vol. 1, pp. 793–800, 2001. [Online]. Available: http://www-2.cs.cmu.edu/Groups/NIPS/NIPS2001/papers/psgz/AA10.ps.gz

[60] J. F. S. Lin and D. Kulic, "Online segmentation of human motion for automated rehabilitation exercise analysis," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 1, pp. 168–180, 2014.

[61] D. Wu, L. Pigou, P. J. Kindermans, N. D. H. Le, L. Shao, J. Dambre, and J. M. Odobez, "Deep Dynamic Neural Networks for Multimodal Gesture

Segmentation and Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 8, pp. 1583–1597, 2016.

[62] C.-J. Su, C.-Y. Chiang, and J.-Y. Huang, "Kinect-enabled home-based rehabilitation system using Dynamic Time Warping and fuzzy logic," *Applied Soft Computing*, vol. 22, no. November 2010, pp. 652–666, 2014. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1568494614001859

[63] C. M. De Souza Vicente, E. R. Nascimento, L. E. C. Emery, C. A. G. Flor, T. Vieira, and L. B. Oliveira, "High performance moves recognition and sequence segmentation based on key poses filtering," *2016 IEEE Winter Conference on Applications of Computer Vision, WACV 2016*, 2016.

[64] B. Krüger, J. Tautges, A. Weber, and A. Zinke, "Fast Local and Global Similarity Searches in Large Motion Capture Databases," *Eurographics / ACM SIGGRAPH Symposium on Computer Animation*, pp. 1–10, 2010. [Online]. Available: http://portal.acm.org/citation.cfm?id=1921427.1921429

[65] J. Baumann, R. Wessel, B. Krüger, and A. Weber, "Action Graph: A Versatile Data Structure for Action Recognition," *International Conference on Computer Graphics Theory and Applications (GRAPP)*, 2014.

[66] B. Krüger, A. Vögele, T. Willig, A. Yao, R. Klein, and A. Weber, "Efficient Unsupervised Temporal Segmentation of Motion Data," *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 797–812, 2017.

[67] Q. Wang, G. Kurillo, F. Ofli, and R. Bajcsy, "Unsupervised Temporal Segmentation of Repetitive Human Actions Based on Kinematic Modeling and Frequency Analysis," *Proceedings - 2015 International Conference on 3D Vision, 3DV 2015*, no. 1111965, pp. 562–570, 2015.

[68] J. F. e. S. Lin, V. Joukov, and D. Kulic, "Human motion segmentation by data point classification," *Conference proceedings : ... Annual International*

Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference, vol. 2014, pp. 9–13, 2014.

[69] M. Devanne, H. Wannous, P. Pala, S. Berretti, M. Daoudi, and A. Del Bimbo, "Combined shape analysis of human poses and motion units for action segmentation and recognition," *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, pp. 1–6, 2015. [Online]. Available: http://ieeexplore.ieee.org/document/7284880/

[70] J. Shan and S. Akella, "3D Human Action Segmentation and Recognition using Pose Kinetic Energy," *Coitweb.Uncc.Edu*, 2014. [Online]. Available: http://coitweb.uncc.edu/{~}sakella/papers/ShanAkellaARSO2014.pdf

[71] E. Dolatabadi, Y. X. Zhi, B. Ye, M. Coahran, G. Lupinacci, A. Mihailidis, R. Wang, and B. Taati, "The toronto rehab stroke pose dataset to detect compensation during stroke rehabilitation therapy," *Proceedings of the 11th EAI International Conference on Pervasive Computing Technologies for Healthcare - PervasiveHealth '17*, pp. 375–381, 2017. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3154862.3154925

[72] A. Vakanski, H.-p. Jun, D. Paul, and R. Baker, "A Data Set of Human Body Movements for Physical Rehabilitation Exercises," *Data*, vol. 3, no. 1, p. 2, 2018. [Online]. Available: http://www.mdpi.com/2306-5729/3/1/2

[73] "CARNEGIE MELLON UNIVERSITY GRAPHICS LAB: CMU Motion Capture Database." [Online]. Available: mocap.cs.cmu.edu

[74] M. Müller, A. Baak, and H.-P. Seidel, "Efficient and Robust Annotation of Motion Capture Data," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA)*, New Orleans, USA, aug 2009, pp. 17–26.

[75] "Berkeley MHAD Dataset." [Online]. Available: http://tele-immersion.citris-uc.org/berkeley{_}mhad

[76] "Microsoft Research Cambridge-12 dataset (MSRC-12)." [Online]. Available: https://www.microsoft.com/en-us/download/details.aspx?id=52283

[77] "MSR-Daily Activity Dataset." [Online]. Available: https://uowmailedu-my.sharepoint.com/

[78] "MSR-Action3D Dataset." [Online]. Available: https://uowmailedu-my.sharepoint.com/

[79] "Cornell Activity Datasets: CAD-60 & CAD-120." [Online]. Available: http://pr.cs.cornell.edu/humanactivities/data.php

[80] "UTKinect-Action3D Dataset." [Online]. Available: http://cvrc.ece.utexas.edu/KinectDatasets/HOJ3D.html

[81] C. Ellis, S. Z. Masood, M. F. Tappen, J. J. Laviola, and R. Sukthankar, "Exploring the trade-off between accuracy and observational latency in action recognition," *International Journal of Computer Vision*, vol. 101, no. 3, pp. 420–436, 2013.

[82] A. R. Fugl-Meyer, L. Jääskö, I. Leyman, S. Olsson, and S. Steglind, "The post-stroke hemiplegic patient. 1. a method for evaluation of physical performance." *Scandinavian journal of rehabilitation medicine*, vol. 7, no. 1, pp. 13–31, 1975. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/1135616

[83] C. Gowland, P. Stratford, M. Ward, J. Moreland, W. Torresin, S. Van Hullenaar, J. Sanford, S. Barreca, B. Vanspall, and N. Plews, "Measuring physical impairment and disability with the chedoke-mcmaster stroke assessment," *Stroke*, vol. 24, no. 1, pp. 58–63, 1993.

[84] K. Daley and S. Wood-dauphine, "Reliability of Scores on the Stroke Rehabilitation Assessment of Movement (STREAM) Measure," *Physical*

Therapy, vol. 79, no. 1, 1999.

[85] D. Carroll, “a Quantitative Test of Upper Extremity,” *J. chron. Dis*, vol. 18, pp. 479–491, 1965.

[86] V. Mathiowetz and K. Weber, “Adult N o rills for the Box and Block,” *the American Journal of Ocupational Therapy*, vol. 39, no. 6, pp. 387–391, 1985.

[87] H. Sakoe and S. Chiba, “Dynamic Programming Algorithm Optimization for Spoken Word Recognition,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[88] J. Aach and G. M. Church, “Aligning gene expression time series with time warping algorithms,” *Bioinformatics*, vol. 17, no. 6, pp. 495–508, 2001. [Online]. Available: https://watermark.silverchair.com/170495.pdf

[89] Z. Bar-Joseph, G. Gerber, D. K. Gifford, T. S. Jaakkola, and I. Simon, “A New Approach to Analyzing Gene Expression Time Series Data,” in *Proceedings of the sixth annual international conference on Computational biology*, 2002, pp. 39–48. [Online]. Available: https://users.cs.duke.edu/{~}brd/Teaching/Bio/asmb/Papers/2000/Microarray/jaakkola-recomb02.pdf

[90] Z. Bar-Joseph, G. K. Gerber, D. K. Gifford, T. S. Jaakkola, and I. Simon, “Continuous Representations of Time-Series Gene Expression Data,” *JOURNAL OF COMPUTATIONAL BIOLOGY*, vol. 10, pp. 341–356, 2003. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.4.3554{&}rep=rep1{&}type=pdf

[91] L. Rabiner, A. Rosenberg, and S. Levinson, “Considerations in dynamic time warping algorithms for discrete word recognition,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 6, pp. 575–582, dec 1978. [Online]. Available: http://ieeexplore.ieee.org/document/1163164/

[92] C. Myers and L. Rabiner, “A level building dynamic time warping algorithm for connected word recognition,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 29, no. 2, pp. 284–297, apr 1981. [Online]. Available: http://ieeexplore.ieee.org/document/1163527/

[93] L. Muda, M. Begam, and I. Elamvazuthi, “Voice Recognition Algorithms using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) Techniques,” Tech. Rep., 2010. [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1003/1003.4083.pdf

[94] M. Parizeau and R. Plamondon, “A comparative analysis of regional correlation, dynamic time warping, and skeletal tree matching for signature verification,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 710–717, jul 1990. [Online]. Available: http://ieeexplore.ieee.org/document/56215/

[95] T. Rath and R. Manmatha, “Word image matching using dynamic time warping,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2. IEEE Comput. Soc, 2003, pp. II–521–II–527. [Online]. Available: http://ieeexplore.ieee.org/document/1211511/

[96] C. Bahlmann and H. Burkhardt, “The writer independent online handwriting recognition system frog on hand and cluster generative statistical dynamic time warping,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 3, pp. 299–310, mar 2004. [Online]. Available: http://ieeexplore.ieee.org/document/1262308/

[97] Ning Hu, R. Dannenberg, and G. Tzanetakis, “Polyphonic audio matching and alignment for music retrieval,” in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No.03TH8684)*. IEEE, 2003, pp. 185–188. [Online]. Available: http://ieeexplore.ieee.org/document/1285862/

[98] A. Youssef, T. Abdel-Galil, E. El-Saadany, and M. Salama, “Disturbance Classification Utilizing Dynamic Time Warping Classifier,” *IEEE Transactions on Power Delivery*, vol. 19, no. 1, pp. 272–278, jan 2004. [Online]. Available: http://ieeexplore.ieee.org/document/1256388/

[99] J. Lyons, N. Biswas, A. Sharma, A. Dehzangi, and K. K. Paliwal, “Protein fold recognition by alignment of amino acid residues using kernelized dynamic time warping,” *Journal of Theoretical Biology*, vol. 354, pp. 137–145, aug 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022519314001805

[100] A. Corradini, “Dynamic time warping for off-line recognition of a small gesture vocabulary,” in *Proceedings IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems.* IEEE Comput. Soc, 2001, pp. 82–89. [Online]. Available: http://ieeexplore.ieee.org/document/938914/

[101] A. Kuzmanic and V. Zanchi, “Hand shape classification using DTW and LCSS as similarity measures for vision-based gesture recognition system,” in *EUROCON 2007 - The International Conference on "Computer as a Tool".* IEEE, 2007, pp. 264–269. [Online]. Available: http://ieeexplore.ieee.org/document/4400350/

[102] E. F. Krause, *Taxicab geometry: An adventure in non-Euclidean geometry.*, 1986.

[103] F. Lv and R. Nevatia, “Recognition and Segmentation of 3-D Human Action Using HMM and Multi-class AdaBoost,” in *European Conference on Computer Vision.* Springer, Berlin, Heidelberg, 2006, pp. 359–372. [Online]. Available: http://link.springer.com/10.1007/11744085{_}28

[104] Qinfeng Shi, Li Wang, Li Cheng, and A. Smola, “Discriminative human action

segmentation and recognition using semi-Markov model," in *2008 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE, jun 2008, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/document/4587557/

[105] J. F.-S. Lin, V. Joukov, and D. Kulić, "Classification-based Segmentation for Rehabilitation Exercise Monitoring," *Journal of Rehabilitation and Assistive Technologies Engineering*, vol. 5, pp. 1–12, 2018.

[106] S. Sanei, *Adaptive Processing of Brain Signals*, 2013. [Online]. Available: https://books.google.co.uk/books?hl=en{&}lr={&}id=Kti1XX0EObIC{&}oi=fnd{&}pg=PT9{&}dq=Adaptive+processing+of+Brain+Signals{&}ots=S48T-jrf9g{&}sig=VSJqMgX7Dz5QCZ23AOpkLBUzyco

[107] C. Böhm, S. Berchtold, and D. A. Keim, "Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases," *ACM Computing Surveys*, vol. 33, no. 3, pp. 322–373, 2001. [Online]. Available: http://portal.acm.org/citation.cfm?doid=502807.502809

[108] E. Keogh, T. Palpanas, V. B. Zordan, D. Gunopulos, and M. Cardle, "Indexing Large Human-Motion Databases," *Proceedings of the 30th International Conference on Very Large Data Bases*, pp. 780–791, 2004. [Online]. Available: http://dl.acm.org/citation.cfm?id=1316757{%}0Ahttp://www.vldb.org/conf/2004/RS21P1.PDF

[109] Microsoft, "Kinect for Windows features." [Online]. Available: https://www.microsoft.com/en-us/kinectforwindows/meetkinect/features.aspx

[110] "Clinical Evaluation of Depth Sensor - YouTube." [Online]. Available: https://www.youtube.com/watch?v={_}oMcZC8rilM{&}feature=youtu.be

[111] Qualisys, "A complete Running Analysis solution," 2015. [Online]. Available: http://www.qualisys.com/applications/sports/running/

[112] L. Yang, L. Zhang, H. Dong, A. Alelaiwi, and A. E. Saddik, “Evaluating and Improving the Depth Accuracy of Kinect for Windows v2,” *IEEE Sensors Journal*, vol. 15, no. 8, pp. 4275–4285, 2015. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7067384

[113] W. Kabsch and IUCr, “A solution for the best rotation to relate two sets of vectors,” *Acta Crystallographica Section A*, vol. 32, no. 5, pp. 922–923, sep 1976. [Online]. Available: http://scripts.iucr.org/cgi-bin/paper?S0567739476001873

[114] K. Krippendorff, “ESTIMATING THE RELIABILITY, SYSTEMATIC ERROR AND RANDOM ERROR OF INTERVAL DATA,” pp. 61–70, 1970.

[115] A. F. Hayes and K. Krippendorff, “Answering the Call for a Standard Reliability Measure for Coding Data,” *Communication Methods and Measures*, vol. 1, no. 1, pp. 77–89, 2007. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/19312450709336664

[116] J. R. Landis and G. G. Koch, “The measurement of observer agreement for categorical data.” *Biometrics*, vol. 33, no. 1, pp. 159–74, mar 1977. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/843571

[117] K. Krippendorff, *Content analysis : an introduction to its methodology.* SAGE, 1980.

[118] A. Zapf, S. Castell, L. Morawietz, and A. Karch, “Measuring inter-rater reliability for nominal data - which coefficients and confidence intervals are appropriate?” *BMC medical research methodology*, vol. 16, p. 93, 2016. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/27495131http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4974794

[119] O. Arikan, “Compression of motion capture databases,” *Tog*, vol. 25, p. 890, 2006.

[120] D. Sart, A. Mueen, W. Najjar, E. Keogh, V. Niennattrakul, and E. Keogh, “Accelerating dynamic time warping subsequence search with GPUs and FPGAs,” *Proceedings - IEEE International Conference on Data Mining, ICDM*, no. December, pp. 1001–1006, 2010.

[121] A. Fod, M. J. Matari, and O. C. Jenkins, “Automated derivation of primitives for movement classification,” *Autonomous Robots*, vol. 12, no. 1, pp. 39–54, 2002.

[122] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, “Searching and mining trillions of time series subsequences under dynamic time warping,” *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 262–270, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2339576{%}5Cnhttp://practicalquant.blogspot.com/2012/10/mining-time-series-with-trillions-of.html

[123] F. Itakura, “Minimum prediction residual principle applied to speech recognition,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, no. 1, pp. 67–72, feb 1975. [Online]. Available: http://ieeexplore.ieee.org/document/1162641/

[124] Janice Eng and Jocelyn Harris, “GRASP (Graded Repetitive Arm Supplementary Program).” [Online]. Available: http://neurorehab.med.ubc.ca/grasp/

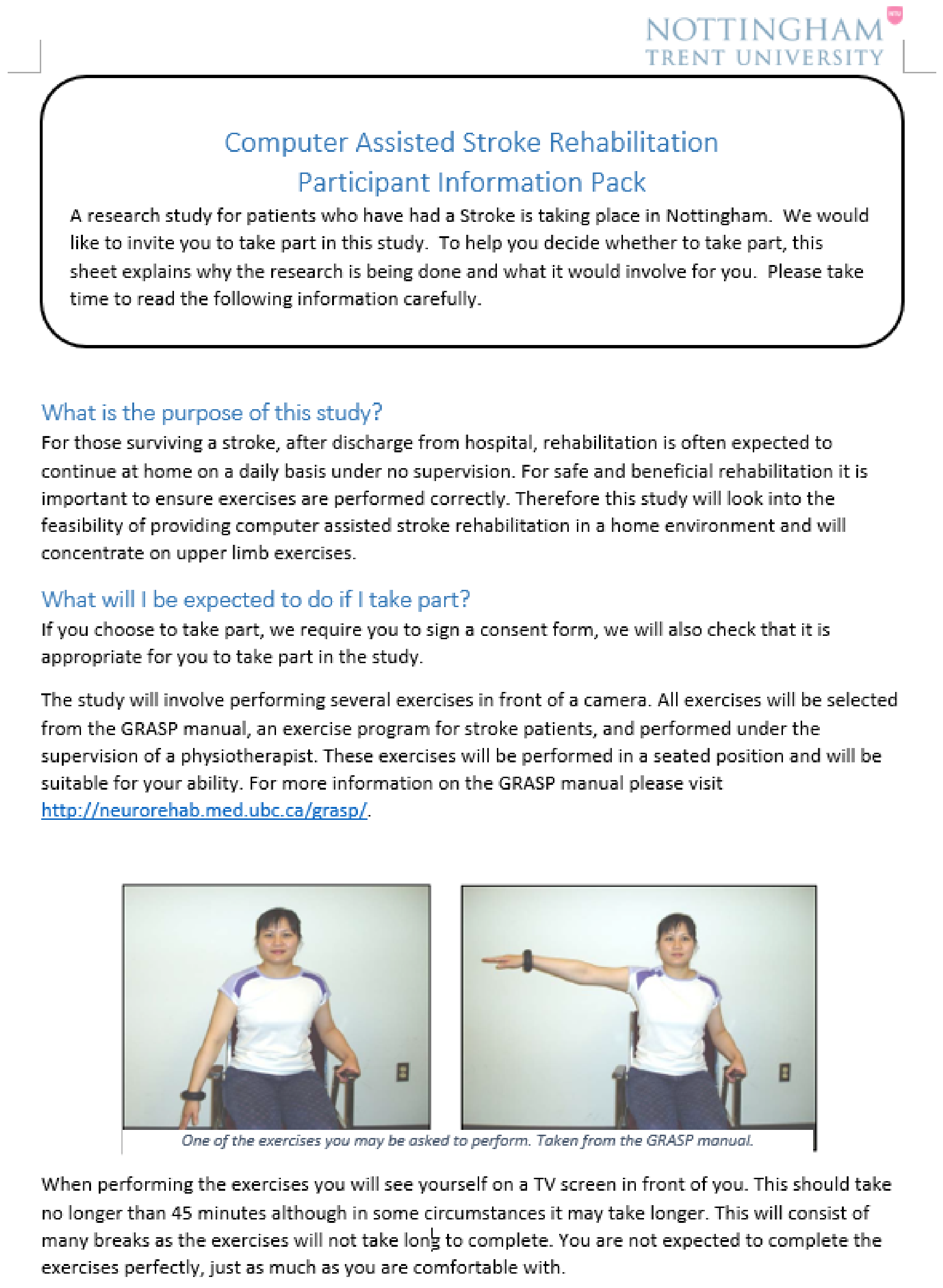
[125] Qualisys, “Qualisys Engineering.” [Online]. Available: https://www.qualisys.com/applications/engineering/

[126] J. F. S. Lin, M. Karg, and D. Kulić, “Movement Primitive Segmentation for Human Motion Modeling: A Framework for Analysis,” *IEEE Transactions on Human-Machine Systems*, 2016.

Appendices

A Participant information pack

The participant information pack handed out at Nottingham CityCare's stroke community groups to recruit stroke patients for data collection.



NOTTINGHAM TRENT UNIVERSITY

Computer Assisted Stroke Rehabilitation
Participant Information Pack

A research study for patients who have had a Stroke is taking place in Nottingham. We would like to invite you to take part in this study. To help you decide whether to take part, this sheet explains why the research is being done and what it would involve for you. Please take time to read the following information carefully.

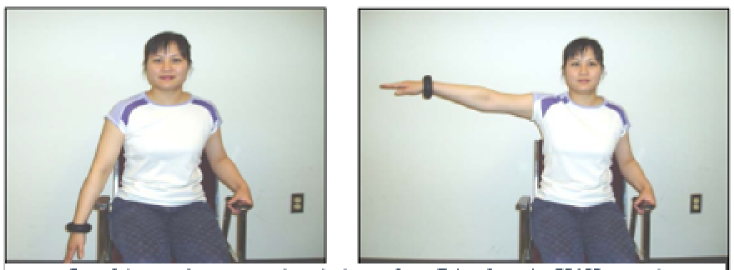
What is the purpose of this study?

For those surviving a stroke, after discharge from hospital, rehabilitation is often expected to continue at home on a daily basis under no supervision. For safe and beneficial rehabilitation it is important to ensure exercises are performed correctly. Therefore this study will look into the feasibility of providing computer assisted stroke rehabilitation in a home environment and will concentrate on upper limb exercises.

What will I be expected to do if I take part?

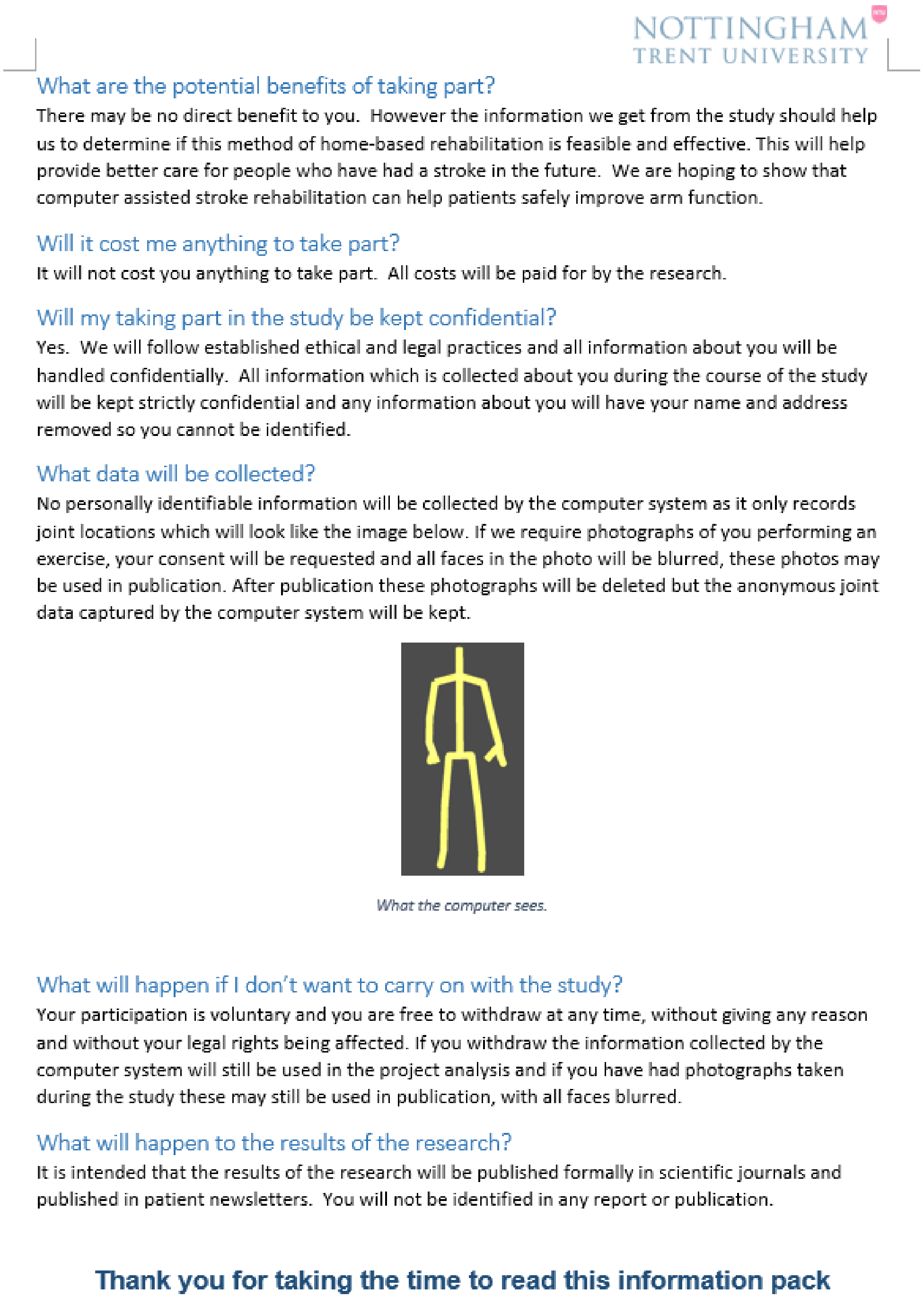
If you choose to take part, we require you to sign a consent form, we will also check that it is appropriate for you to take part in the study.

The study will involve performing several exercises in front of a camera. All exercises will be selected from the GRASP manual, an exercise program for stroke patients, and performed under the supervision of a physiotherapist. These exercises will be performed in a seated position and will be suitable for your ability. For more information on the GRASP manual please visit http://neurorehab.med.ubc.ca/grasp/.



One of the exercises you may be asked to perform. Taken from the GRASP manual.

When performing the exercises you will see yourself on a TV screen in front of you. This should take no longer than 45 minutes although in some circumstances it may take longer. This will consist of many breaks as the exercises will not take long to complete. You are not expected to complete the exercises perfectly, just as much as you are comfortable with.



NOTTINGHAM
TRENT UNIVERSITY

What are the potential benefits of taking part?

There may be no direct benefit to you. However the information we get from the study should help us to determine if this method of home-based rehabilitation is feasible and effective. This will help provide better care for people who have had a stroke in the future. We are hoping to show that computer assisted stroke rehabilitation can help patients safely improve arm function.

Will it cost me anything to take part?

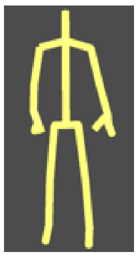
It will not cost you anything to take part. All costs will be paid for by the research.

Will my taking part in the study be kept confidential?

Yes. We will follow established ethical and legal practices and all information about you will be handled confidentially. All information which is collected about you during the course of the study will be kept strictly confidential and any information about you will have your name and address removed so you cannot be identified.

What data will be collected?

No personally identifiable information will be collected by the computer system as it only records joint locations which will look like the image below. If we require photographs of you performing an exercise, your consent will be requested and all faces in the photo will be blurred, these photos may be used in publication. After publication these photographs will be deleted but the anonymous joint data captured by the computer system will be kept.



What the computer sees.

What will happen if I don't want to carry on with the study?

Your participation is voluntary and you are free to withdraw at any time, without giving any reason and without your legal rights being affected. If you withdraw the information collected by the computer system will still be used in the project analysis and if you have had photographs taken during the study these may still be used in publication, with all faces blurred.

What will happen to the results of the research?

It is intended that the results of the research will be published formally in scientific journals and published in patient newsletters. You will not be identified in any report or publication.

Thank you for taking the time to read this information pack

B Data collection tool code

Code snippets from the C# data collection tool used to record data of stroke participants performing rehabilitation exercises.

```
if (PoseResolution.Options.OptionsWindow.RecordDepth())
{
    // Create live media object for converting in real-time
    fstream = new FileStream("actions/" + PoseResolution.Options.OptionsWindow.GetActionText() + "Video.mp4",
     FileMode.Create,
     FileAccess.Write, FileShare.None);
    ffMpeg = new NReco.VideoConverter.FFMpegConverter();
    ffMpegTask = ffMpeg.ConvertLiveMedia(Format.raw_video, fstream, Format.flv, new ConvertSettings()
    {
        VideoCodec = NReco.VideoConverter.Format.h264,
        VideoFrameSize = NReco.VideoConverter.FrameSize.hd1080,
        AppendSilentAudioStream = true,
        CustomInputArgs = " -r " + frameRate + " -pixel_format bgra -video_size 1920x1080 ",//CustomInputArgs = " -r 30 -f rawvideo -pix_fmt bgra -video_size 1920x1080 ",
        CustomOutputArgs = " -an -f flv -vcodec libx264 -pix_fmt yuv420p  -video_size 1920x1080 ", //-threads 7  -tune zerolatency -g -movflags frag_keyframe+empty_moov
    });
    ffMpegTask.Start();
    ffMpeg.LogReceived += Event_OutputDataReceived;
}
```

```
// Color
using (ColorFrame frame = colRef.AcquireFrame())
{
    if (frame != null)
    {
        // Calculate colour framerate e.g. is kinect in low light 15fps or not 30fps
        if (frame.RelativeTime.TotalSeconds - prevTimeStamp < 20)
            frameRate = 15;
        else
            frameRate = 30;
        prevTimeStamp = frame.RelativeTime.TotalSeconds;
        if (!bFirstFrame)
        {
            startTime = frame.RelativeTime.TotalMilliseconds;
            bFirstFrame = true;
        }
        if (_mode == CameraMode.Color)
        {
            if (bTakeSnapshot)
            {
                SaveImage(KinectCoordinateMapping.ColorExtensions.ToBitmap(frame), PoseResolution.Options.OptionsWindow.GetActionText()+snapshotNumber.ToString());
                bTakeSnapshot = false;
            }
            // If we are recording
            if (PoseResolution.Options.OptionsWindow.isStopEnabled() && PoseResolution.Options.OptionsWindow.RecordDepth())
            {
                byte[] frameArray = new byte[(1920 * 1080)*Constants.BYTES_PER_PIXEL];
                frame.CopyConvertedFrameDataToArray(frameArray,ColorImageFormat.Bgra);
                ffMpegTask.Write(frameArray,0,frameArray.Length);
                videoMeta.Add(Math.Round(frame.RelativeTime.TotalMilliseconds - startTime).ToString());
            }
            camera.Source = KinectCoordinateMapping.ColorExtensions.ToBitmap(frame);
        }
    }
}
```

```
// Joint Positions
foreach (Joint joint in body.Joints.Values)
{
    // Only process upper body joints and ignore hand joints
    if (!JointsToIgnore(joint.JointType))
    {
        // If stop is enabled then we are recording
        if (PoseResolution.Options.OptionsWindow.isStopEnabled())
        {
            joints += string.Format("{0},{1},{2},", joint.Position.X, joint.Position.Y, joint.Position.Z);
            trackingStates += string.Format("{0},", joint.TrackingState);
        }
        // 3D space point
        CameraSpacePoint jointPosition = joint.Position;

        // 2D space point
        Point point = new Point();

        if (_mode == CameraMode.Color)
        {
            ColorSpacePoint colorPoint = _sensor.CoordinateMapper.MapCameraPointToColorSpace(jointPosition);

            point.X = float.IsInfinity(colorPoint.X) ? 0 : colorPoint.X;
            point.Y = float.IsInfinity(colorPoint.Y) ? 0 : colorPoint.Y;
        }
        else if (_mode == CameraMode.Depth || _mode == CameraMode.Infrared) // Change the Image and Canvas dimensions to 512x424
        {
            DepthSpacePoint depthPoint = _sensor.CoordinateMapper.MapCameraPointToDepthSpace(jointPosition);

            point.X = float.IsInfinity(depthPoint.X) ? 0 : depthPoint.X;
            point.Y = float.IsInfinity(depthPoint.Y) ? 0 : depthPoint.Y;
        }

        Ellipse ellipse;

        if (PoseResolution.Options.OptionsWindow.isStopEnabled())
        {
            // Draw
            ellipse = new Ellipse
```

C Quantitative analysis code

Code snippets of the quantitative analysis of Kinect's pose estimation algorithm written in Matlab.

```matlab
function CalculateRelativeDeviation(hObject, EventData)
    % Get KSkel start and end frames
    KSkelMin = KSkel(startFramek:endFramek, :);
    % Get indices of MSkel that minimise time difference KSkel startFramek:endFramek
    QSkelIdx = arrayfun(@MinVal, KSkelMin(:,1:1));
    KSkelMin = KSkelMin*100;
    QSkelMin = MSkel(QSkelIdx, :)*100;
    % Remove NaNs from signal i.e. get indicies of non NAN values
    ids = find(~isnan(QSkelMin(:,(joint*3)+axes)));
    kRelChange = arrayfun(@RelChange,KSkelMin(ids(1:end-1),(joint*3)+axes),KSkelMin(ids(2:end),(joint*3)+axes));
    qRelChange = arrayfun(@RelChange,QSkelMin(ids(1:end-1),(joint*3)+axes),QSkelMin(ids(2:end),(joint*3)+axes));
    difference = arrayfun(@GetDifference,kRelChange,qRelChange);
    meanDifference = mean(difference);
    set(findobj('Tag','KinText'),'String',sprintf('%s%.2f','Rel Dev(cm): ',meanDifference));
end
```

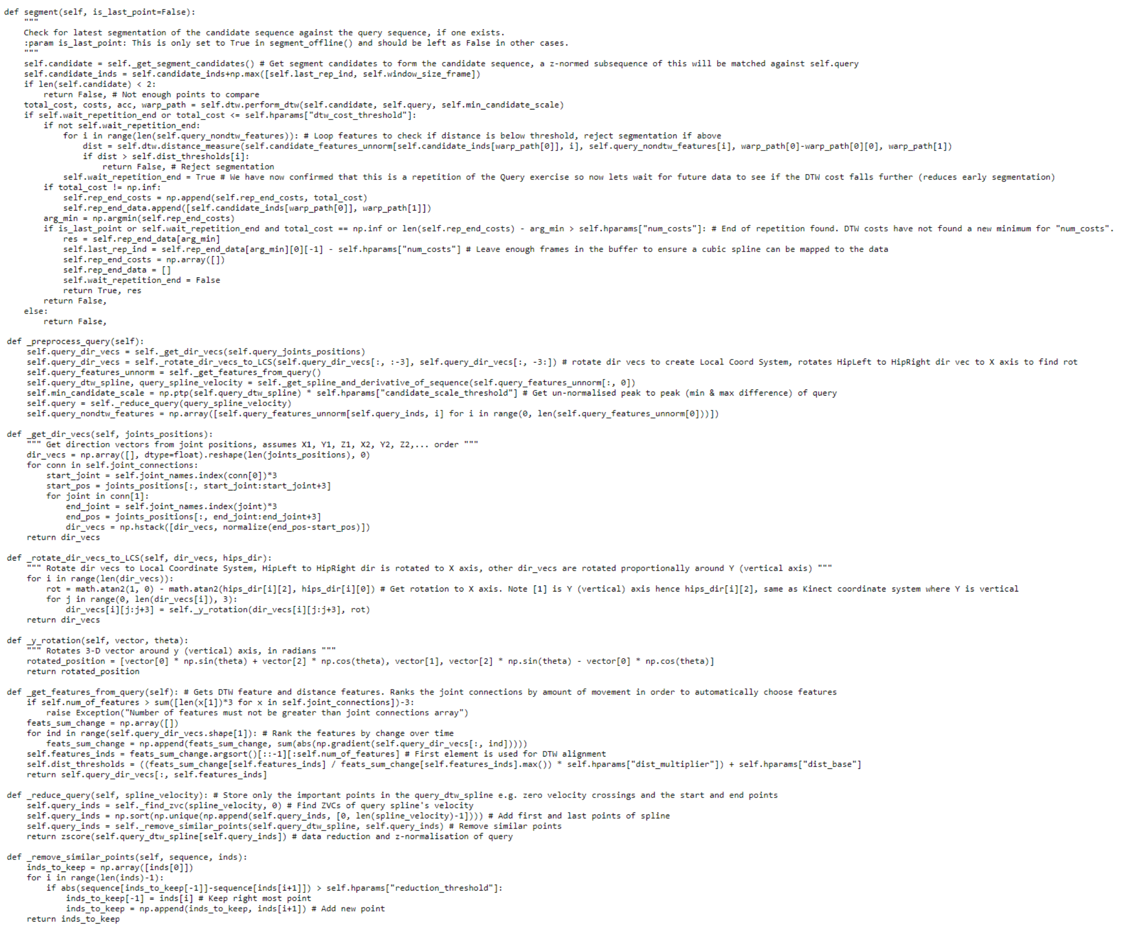
```matlab
   % HipLeft,HipRight,SpineShoulder used to align the datasets using Kabsch algorithm
   if (rotFrameK ~= -1)
       if (~any(isnan([xm(rotJoints),xmr(rotJoints)])) && ~any(isnan([ym(rotJoints),ymr(rotJoints)])) && ~any(isnan([zm(rotJoints),zmr(rotJoints)])))
           [U, r, lrms] = Kabsch([[xm(rotJoints),xmr(rotJoints)];[ym(rotJoints),ymr(rotJoints)];[zm(rotJoints),zmr(rotJoints)]],[[xk(rotJoints),xkr(rotJ
       end
   else
       if (~any(isnan(xm(rotJoints))) && ~any(isnan(ym(rotJoints))) && ~any(isnan(zm(rotJoints))))
           [U, r, lrms] = Kabsch([xm(rotJoints);ym(rotJoints);zm(rotJoints)],[xk(rotJoints);yk(rotJoints);zk(rotJoints)]);
       end
   end
   rot = U*[xm;ym;zm];
   xm = rot(1,:);
   ym = rot(2,:);
   zm = rot(3,:);

   % Redo Kabsch on different joints for a suitable translation
   if (~any(isnan(xm(13))) && ~any(isnan(ym(13))) && ~any(isnan(zm(13))))
       [a, r, b] = Kabsch([xm(13);ym(13);zm(13)],[xk(13);yk(13);zk(13)]);
   end
   xm = xm+r(1);
   ym = ym+r(2);
   zm = zm+r(3);
end
```

```matlab
% Model SDs as ellipsoids
if (bViewSD && ~isempty(stdDevEllipsoid))
    stdDevEllipsoid = reshape(stdDevEllipsoid,3,[])';
    for i = 1:size(stdDevEllipsoid,1)
        % Plot SD ellipsoid, divide by 2 as parameter is radius NOT
        % diameter and size parameters are X, Z, Y
        [x, y, z] = ellipsoid(xk(i),yk(i),zk(i),stdDevEllipsoid(i,1)/2,stdDevEllipsoid(i,3)/2,stdDevEllipsoid(i,2)/2,100);
        surf(x, y, z,'EdgeColor','none','FaceColor',[0 0 0],'SpecularExponent',4);
    end
    stdDevEllipsoid = reshape(stdDevEllipsoid',1,[]);
    %stdDevs = sqrt(sum(stdDevs.^2,2));
    %stdDevs = stdDevs';
end
```

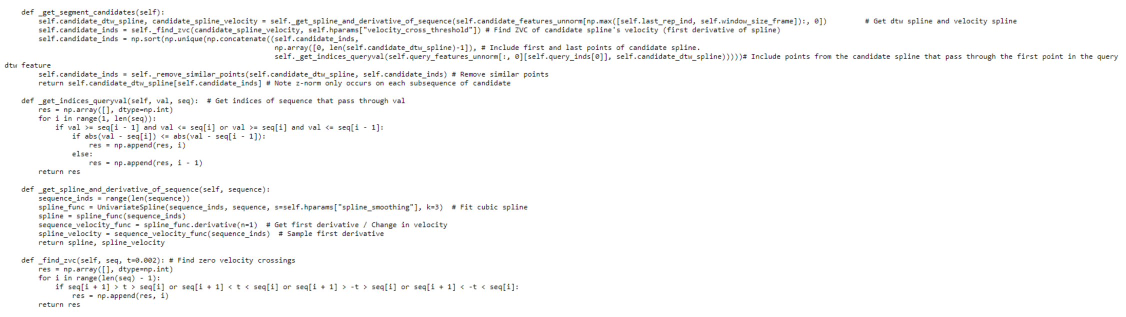
D Segmentation algorithm code

Code snippets of the segmentation algorithm developed in Python.

```
def perform_dtw(self, x, y, scale_threshold):
    """
    Find latest subsequence of x that matches y.
    :param x: candidate sequence (C)
    :param y: query sequence (Q)
    :param scale_threshold: minimum scale the un-normalised subsequence needs to be to be considered for alignment
    :return: best_cost (subsequence (S) with lowest cost), Cbest (cost matrix of S & Q), Dbest (accumulative cost matrix of S & Q), best_path (DTW warp path between S & Q)
    """
    best_path = None
    best_cost = inf
    Cbest = None
    Dbest = None
    for i in reversed(range(0, len(x) - 2)): # Start from latest data in x
        if np.ptp(x[i:]) < scale_threshold: # if size of un-normalised subsequence is below threshold then ignore (z-norm takes scale out of the equation)
            continue
        C, D = self._get_cost_matrices(zscore(x[i:]), y)  # Znorm this subsequence and get cost matrices
        path = self._get_path(i, D)  # Get path of subsequence
        # Calculate cost, multiple x elements aligned to a single y element are summed and averaged
        pat_path = path[0] - i
        gt_path = path[1]
        total_cost = 0
        costs_arr = np.array([])
        for j in range(len(gt_path)):
            costs_arr = np.append(costs_arr,
                                  C[gt_path[j], pat_path[j]])  # Note C[path[1],path[0]] is correct order
            if j == len(gt_path) - 1 or gt_path[j] != gt_path[j + 1]:
                total_cost += costs_arr.sum() / len(costs_arr)  # Average costs
                if total_cost / len(y) > best_cost:  # Give up if total cost is greater than cost so far
                    total_cost = inf
                    break
                costs_arr = np.array([])
        total_cost = total_cost / len(y)

        if total_cost < best_cost:
            best_path = path
            best_cost = total_cost
            Cbest = C
            Dbest = D
    return best_cost, Cbest, Dbest, best_path

def distance_measure(self, x, y, pathx, pathy):
    """
    Perform distance measure between two sequences.
    """
    total_cost = 0
    distances = np.array([])
    for i in range(len(pathy)):
        distances = np.append(distances, self._dist(x[pathx[i]], y[pathy[i]]))
        if i == len(pathy) - 1 or pathy[i] != pathy[i + 1]:
            total_cost += distances.sum() / len(distances)  # Average costs
            distances = np.array([])
    return total_cost / len(y)


def _dist(self, x, y): # Manhatten/absolute distance function for DTW
    return abs(x - y)

def _get_cost_matrices(self, x, y): # Get cost matrix (C) and accumulated cost matrix (D)
    r, c = len(x), len(y)
    r1, c1 = r - 1, c - 1
    D = np.zeros((c, r), dtype=np.float32)
    for i in range(c):
        for j in range(r):
            D[i, j] = self._dist(x[j], y[i])
    C = D.copy()
    for i in reversed(range(c)):
        for j in reversed(range(r)):
            if i == c1 and j == r1:
                continue
            i1, j1 = i + 1, j + 1
            D[i, j] += min(D[i1, j1] if i1 < c and j1 < r else inf, D[i, j1] if j1 < r else inf,
                           D[i1, j] if i1 < c else inf)
    return (C, D)

def _get_path(self, oi, D): # Get DTW path
    ilen, jlen = np.array(D.shape) - 1
    i, j = ilen, jlen
    p, q = [i], [j]
    while i > 0 or j > 0:
        i1, j1 = i - 1, j - 1
        tb = np.argmin((D[i1, j1] if i1 >= 0 and j1 >= 0 else inf, D[i1, j] if i1 >= 0 and j >= 0 else inf,
                        D[i, j1] if i >= 0 and j1 >= 0 else inf))
        if tb == 0:
            i -= 1
            j -= 1
        elif tb == 1:
            i -= 1
        else:
            j -= 1
        p.insert(0, i)
        q.insert(0, j)
    return np.array(q) + oi, np.array(p)
```



```
def segment(self, is_last_point=False):
    """
    Check for latest segmentation of the candidate sequence against the query sequence, if one exists.
    :param is_last_point: This is only set to True in segment_offline() and should be left as False in other cases.
    """
    self.candidate = self._get_segment_candidates() # Get segment candidates to form the candidate sequence, a z-normed subsequence of this will be matched against self.query
    self.candidate_inds = self.candidate_inds+np.max([self.last_rep_ind, self.window_size_frame])
    if len(self.candidate) < 2:
        return False, # Not enough points to compare
    total_cost, costs, acc, warp_path = self.dtw.perform_dtw(self.candidate, self.query, self.min_candidate_scale)
    if self.wait_repetition_end or total_cost <= self.hparams["dtw_cost_threshold"]:
        if not self.wait_repetition_end:
            for i in range(len(self.query_nondtw_features)): # Loop features to check if distance is below threshold, reject segmentation if above
                dist = self.dtw.distance_measure(self.candidate_features_unnorm[self.candidate_inds[warp_path[0]], i], self.query_nondtw_features[i], warp_path[0]-warp_path[0][0], warp_path[1])
                if dist > self.dist_thresholds[i]:
                    return False, # Reject segmentation
            self.wait_repetition_end = True # We have now confirmed that this is a repetition of the Query exercise so now lets wait for future data to see if the DTW cost falls further (reduces early segmentation)
        if total_cost != np.inf:
            self.rep_end_costs = np.append(self.rep_end_costs, total_cost)
            self.rep_end_data.append([self.candidate_inds[warp_path[0]], warp_path[1]])
        arg_min = np.argmin(self.rep_end_costs)
        if is_last_point or self.wait_repetition_end and total_cost == np.inf or len(self.rep_end_costs) - arg_min > self.hparams["num_costs"]: # End of repetition found. DTW costs have not found a new minimum for "num_costs".
            res = self.rep_end_data[arg_min]
            self.last_rep_ind = self.rep_end_data[arg_min][0][-1] - self.hparams["num_costs"] # Leave enough frames in the buffer to ensure a cubic spline can be mapped to the data
            self.rep_end_costs = np.array([])
            self.rep_end_data = []
            self.wait_repetition_end = False
            return True, res
        return False,
    else:
        return False,

 def _preprocess_query(self):
     self.query_dir_vecs = self._get_dir_vecs(self.query_joints_positions)
     self.query_dir_vecs = self._rotate_dir_vecs_to_LCS(self.query_dir_vecs[:, :-3], self.query_dir_vecs[:, -3:]) # rotate dir vecs to create Local Coord System, rotates HipLeft to HipRight dir vec to X axis to find rot
     self.query_features_unnorm = self._get_features_from_query()
     self.query_dtw_spline, query_spline_velocity = self._get_spline_and_derivative_of_sequence(self.query_features_unnorm[:, 0])
     self.min_candidate_scale = np.ptp(self.query_dtw_spline) * self.hparams["candidate_scale_threshold"] # Get un-normalised peak to peak (min & max difference) of query
     self.query = self._reduce_query(query_spline_velocity)
     self.query_nondtw_features = np.array([self.query_features_unnorm[self.query_inds, i] for i in range(0, len(self.query_features_unnorm[0]))])

 def _get_dir_vecs(self, joints_positions):
     """ Get direction vectors from joint positions, assumes X1, Y1, Z1, X2, Y2, Z2,... order """
     dir_vecs = np.array([], dtype=float).reshape(len(joints_positions), 0)
     for conn in self.joint_connections:
         start_joint = self.joint_names.index(conn[0])*3
         start_pos = joints_positions[:, start_joint:start_joint+3]
         for joint in conn[1]:
             end_joint = self.joint_names.index(joint)*3
             end_pos = joints_positions[:, end_joint:end_joint+3]
             dir_vecs = np.hstack([dir_vecs, normalize(end_pos-start_pos)])
     return dir_vecs

 def _rotate_dir_vecs_to_LCS(self, dir_vecs, hips_dir):
     """ Rotate dir vecs to Local Coordinate System, HipLeft to HipRight dir is rotated to X axis, other dir_vecs are rotated proportionally around Y (vertical axis) """
     for i in range(len(dir_vecs)):
         rot = math.atan2(1, 0) - math.atan2(hips_dir[i][2], hips_dir[i][0]) # Get rotation to X axis. Note [1] is Y (vertical) axis hence hips_dir[i][2], same as Kinect coordinate system where Y is vertical
         for j in range(0, len(dir_vecs[i]), 3):
             dir_vecs[i][j:j+3] = self._y_rotation(dir_vecs[i][j:j+3], rot)
     return dir_vecs

 def _y_rotation(self, vector, theta):
     """ Rotates 3-D vector around y (vertical) axis, in radians """
     rotated_position = [vector[0] * np.sin(theta) + vector[2] * np.cos(theta), vector[1], vector[2] * np.sin(theta) - vector[0] * np.cos(theta)]
     return rotated_position

 def _get_features_from_query(self): # Gets DTW feature and distance features. Ranks the joint connections by amount of movement in order to automatically choose features
     if self.num_of_features > sum([len(x[1])*3 for x in self.joint_connections])-3:
         raise Exception("Number of features must not be greater than joint connections array")
     feats_sum_change = np.array([])
     for ind in range(self.query_dir_vecs.shape[1]): # Rank the features by change over time
         feats_sum_change = np.append(feats_sum_change, sum(abs(np.gradient(self.query_dir_vecs[:, ind]))))
     self.features_inds = feats_sum_change.argsort()[::-1][:self.num_of_features] # First element is used for DTW alignment
     self.dist_thresholds = ((feats_sum_change[self.features_inds] / feats_sum_change[self.features_inds].max()) * self.hparams["dist_multiplier"]) + self.hparams["dist_base"]
     return self.query_dir_vecs[:, self.features_inds]

 def _reduce_query(self, spline_velocity): # Store only the important points in the query_dtw_spline e.g. zero velocity crossings and the start and end points
     self.query_inds = self._find_zvc(spline_velocity, 0) # Find ZVCs of query spline's velocity
     self.query_inds = np.sort(np.unique(np.append(self.query_inds, [0, len(spline_velocity)-1]))) # Add first and last points of spline
     self.query_inds = self._remove_similar_points(self.query_dtw_spline, self.query_inds) # Remove similar points
     return zscore(self.query_dtw_spline[self.query_inds]) # data reduction and z-normalisation of query

 def _remove_similar_points(self, sequence, inds):
     inds_to_keep = np.array([inds[0]])
     for i in range(len(inds)-1):
         if abs(sequence[inds_to_keep[-1]]-sequence[inds[i+1]]) > self.hparams["reduction_threshold"]:
             inds_to_keep[-1] = inds[i] # Keep right most point
             inds_to_keep = np.append(inds_to_keep, inds[i+1]) # Add new point
     return inds_to_keep
```



```
    def _get_segment_candidates(self):
        self.candidate_dtw_spline, candidate_spline_velocity = self._get_spline_and_derivative_of_sequence(self.candidate_features_unnorm[np.max([self.last_rep_ind, self.window_size_frame]):, 0])         # Get dtw spline and velocity spline
        self.candidate_inds = self._find_zvc(candidate_spline_velocity, self.hparams["velocity_cross_threshold"]) # Find ZVC of candidate spline's velocity (first derivative of spline)
        self.candidate_inds = np.sort(np.unique(np.concatenate((self.candidate_inds,
                                                                np.array([0, len(self.candidate_dtw_spline)-1]), # Include first and last points of candidate spline.
                                                                self._get_indices_queryval(self.query_features_unnorm[:, 0][self.query_inds[0]], self.candidate_dtw_spline))))# Include points from the candidate spline that pass through the first point in the query
dtw feature
        self.candidate_inds = self._remove_similar_points(self.candidate_dtw_spline, self.candidate_inds) # Remove similar points
        return self.candidate_dtw_spline[self.candidate_inds] # Note z-norm only occurs on each subsequence of candidate

    def _get_indices_queryval(self, val, seq):  # Get indices of sequence that pass through val
        res = np.array([], dtype=np.int)
        for i in range(1, len(seq)):
            if val >= seq[i - 1] and val <= seq[i] or val >= seq[i] and val <= seq[i - 1]:
                if abs(val - seq[i]) <= abs(val - seq[i - 1]):
                    res = np.append(res, i)
                else:
                    res = np.append(res, i - 1)
        return res

    def _get_spline_and_derivative_of_sequence(self, sequence):
        sequence_inds = range(len(sequence))
        spline_func = UnivariateSpline(sequence_inds, sequence, s=self.hparams["spline_smoothing"], k=3)  # Fit cubic spline
        spline = spline_func(sequence_inds)
        sequence_velocity_func = spline_func.derivative(n=1)  # Get first derivative / Change in velocity
        spline_velocity = sequence_velocity_func(sequence_inds)  # Sample first derivative
        return spline, spline_velocity

    def _find_zvc(self, seq, t=0.002): # Find zero velocity crossings
        res = np.array([], dtype=np.int)
        for i in range(len(seq) - 1):
            if seq[i + 1] > t > seq[i] or seq[i + 1] < t < seq[i] or seq[i + 1] > -t > seq[i] or seq[i + 1] < -t < seq[i]:
                res = np.append(res, i)
        return res
```

E Pose estimation algorithm code

Code snippets of a pose estimation algorithm currently in development - developed in Python.

```python
def predict_forest(self, img, gt_joint_from, num_steps, num_best_trees=10):
    walk = []  # Store walk Debugging
    walk.append(gt_joint_from)
    for step in range(num_steps):
        predictions = np.empty(shape=(0,3))
        # Get prediction from best (num_best_trees) trees in forest
        for i in range(0, num_best_trees):
            predictions = np.append(predictions, [self.trees[i].predict(img, walk[-1])], axis=0)
        # Average direction vector from all trees
        walk.append(np.mean(predictions, axis=0))
    joint_pos = sum(walk)/num_steps
    return walk[-1], walk


def train_forest(self, training_depth_imgs, gt_joints, point_cloud, num_random_points = 400, b_cluster=True, numTrees=100):
    """ Train all the trees in the forest, note each tree receives a subset of the samples. """
    pool = mp.Pool(self.num_cores)
    self.train_imgs = training_depth_imgs
    self.gt_joints = gt_joints
    self.point_cloud = point_cloud
    self.num_random_points = num_random_points
    utils.generate_offsets(num_random_points)
    self.produce_samples()   # Produce the samples for training
    self.trees = [RTW(i, [i[y] for y in (np.sort(random.sample(range(self.num_samples), round(self.num_samples * self.bootstrapSamplePercent))) for _ in range(1
    print("pool start")
    self.trees = pool.starmap(fit_tree_pool, ((tree, b_cluster, self.predictorsPercent) for tree in self.trees))
    pool.close()
    print("pool finished")
```

```python
def fit(self, b_cluster, predictors_percent):
    """ Fit tree by splitting data_points to minimise the overall variance to the ground truth position.
    Randomly sample predictors_percent of splits at decision node and evaluate."""
    print("Fitting tree: ", self.tree_index)
    # Build tree - Find leaf nodes
    while True:
        node_set = self.tree["nodes"][self.node_evaluating]["set"] # indices of samples in this set
        # Determine if leaf node i.e. if minimum number of samples in set or maximum number of leaf nodes is met
        if len(node_set) > self.node_min:
            # Evaluate splits
            best_split_ind = None  # Store set of best split
            lowest_val = math.inf  # Store value to determine best split
            #split_min = int(self.node_min/2) # Ensures potential splits adhere to minimum samples in a node
            #split_max = int(len(node_set)-split_min)
            # Get two random offets
            offsets = self.all_offsets[np.random.choice(self.all_offsets.shape[0], 2, replace=False)]
            # Calculate features.
            features = np.array([utils.calculate_feature(self.train_imgs[self.samples[0][i]],
                                                         self.samples[1][i],
                                                         offsets) for i in node_set], dtype="float32")
```

```python
def depth_map_to_image(depth_map, joints=None, joints3d=None):
    """
    Visualise depth image and joint prediction
    """
    img = cv2.normalize(depth_map, depth_map, 0, 1, cv2.NORM_MINMAX)
    img = np.array(img * 255, dtype = np.uint8)
    img = cv2.cvtColor(img, cv2.COLOR_GRAY2RGB)
    img = cv2.applyColorMap(img, cv2.COLORMAP_OCEAN)
    for j in range(15):
        x, y = joints[j, 0], joints[j, 1]
        cv2.circle(img, (x,y), 1, (255,255,255), thickness=2)
        cv2.circle(img, utils.point_cloud_to_image(joints3d[j][0],joi
        if j == 0:
            for k in range(400):
                cv2.circle(img, utils.point_cloud_to_image(joints3d[
        #cv2.putText(img, joint_id_to_name[j], (x+5, y+5), cv2.FONT_
    return img
```

```python
# K-Mean clusters at leaf nodes
if b_cluster:
    self.cluster_leaf_nodes()
self.train_imgs = None  # Save space when training
return self
```