Developing a Pipeline for Gait Analysis with a Side-View Depth Sensor

by

© Andrew Hynes

A thesis submitted to the

School of Graduate Studies

in partial fulfilment of the

requirements for the degree of

Master of Engineering

Faculty of Engineering and Applied Science

Memorial University of Newfoundland

May 2019

St. John's Newfoundland and Labrador Canada

Abstract

This thesis presents computational methods for conducting gait analysis with a side-view depth sensor. First, a method to segment human body parts in a depth image is presented. A standard supervised segmentation algorithm is run on a novel graph representation of the depth image. It is demonstrated that the new graph structure improves the accuracy of the segmentation. This contribution is intended to allow fast labelling of depth images for training a human joint predictor. Next, a method is presented to select accurate 3D positions of human joints from multiple proposals. These proposals are generated by a predictor from a side-view depth image. Finally, a gait analysis system is built on the joint selection process. The system calculates standard parameters used in clinical gait analysis. Walking trials have been measured concurrently by a pressure-sensitive walkway and a side-view depth sensor. The estimated gait parameters are validated against the ground truth parameters from the walkway. As future work, the initial segmentation process could be applied to multi-view depth images for training a view-invariant joint predictor. The proposed gait analysis system can then be applied to the predicted joints.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Stephen Czarnuch, who first convinced me to begin this masters degree, and provided invaluable guidance throughout its duration.

I thank the Faculty of Engineering and Applied Science at Memorial University and NSERC for their generous financial support.

I am grateful for the help I received from members of the Recovery and Performance Laboratory of Memorial University, who provided me with gait data and helped me to understand it.

I am thankful for my mother, who taught me writing, and for my father, who taught me math.

Finally, I am thankful for Alice, who accompanied me through every step of this degree.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

GUGT	Get Up and Go Test
HOG	Histogram of Oriented Gradients
ICC	Intraclass Correlation Coefficient
MAD	Median Absolute Deviation
MS	Multiple Sclerosis
PKMAS	ProtoKinetics Movement Analysis Software
RW	Random Walker
SDK	Software Development Kit
SMSW	Short Maximum Speed Walk

List of Symbols

Chapter 2

β	Parameter for weighting graph edges
B_{arm}	Binary arm segment
$B_{connect}$	Set of pixels in B_{inner} in lowest value cluster found by mean shift
B_{inner}	Set of pixels in B_{arm} that are in the four-neighbourhood of F_{other}
B_{outer}	Set of pixels in F_{other} that are in the four-neighbourhood of B_{arm}
$B_{union,\ i}$	Union of the eroded B_i and the erosion of its complement image
C	Cost of arm segment
$\mathbf{C}$	Confusion matrix
C_{min}	Minimum cost
D_{inner}	Depth values of pixels in B_{inner}
F	Foreground binary image
F_{other}	Foreground F outside of B_{arm}
G	Layered graph representation of image
G_B	Base graph layer
G_L, G_R	Left and right graph layers
G_X	Graph layer of arm X
I_{grad}	Image gradient of the probability matrix P
$I_{grad,\ inner}$	Set of gradient values on B_{inner}
I_{seg}	Image from segmenting probability matrix with mean shift
i_u	Value of pixel at node u
$\mathbf{JI}$	Jaccard Index
L_{AB}	Line of pixels between two image positions A and B

L_A, L_B	Pixels along line L_{AB} with labels A and B
N	Local neighbourhood of cellular automaton
n_F	Number of foreground pixels
n_G	Number of nodes in graph G
n_{parts}	Number of body parts to be segmented
P	Probability matrix returned by Random Walker algorithm
PC	Per-class accuracy
r	Ratio of the length of L_A to the total length of L_{AB}
V_A, V_B	Mean metric values from methods A and B
w	Weight of graph edge
X	Subscript for body side (L or R)

Chapter 3

Δ_{path}	Absolute relative difference between weights of paths W_{path} and W_{min}
C_{kin}	Total kinematic cost of a shortest path
f	Index of frame
l	Measured length
$\hat{l}$	Expected length
$\mathbf{p}_{kin,\ i}$	Position of $part_i$ predicted by kinematics
$\mathbf{p}_{path,\ i}$	Position of $part_i$ from the path being considered
$part_i$	Body part with index i
w	Weight of graph edge
W_{min}	Total weight of minimum shortest path
W_{path}	Total weight of a shortest path

Chapter 4

ρ	Spearman correlation coefficient
d_{head}	Distance between initial and final head positions in stride
F_i, F_f	Initial and final frame of a stride
k	Number of passes in front of camera
$\mathbf{p}_{head,\ i}$, $\mathbf{p}_{head,\ f}$	Initial and final head positions
$\mathbf{p}_{proj}$	Projected foot position
$\mathbf{p}_{stance}$	Stance foot position
$\mathbf{p}_{swing,\ i}$, $\mathbf{p}_{swing,\ f}$	Initial and final swing positions
$\mathbf{s}$	Vector from initial to final swing position
y_{min}	Lowest y coordinate of peak foot positions in trial

Chapter 5

α	Assignment of feet from previous frame to current frame
σ	Standard deviation
Φ	Signal used to detect stance phases
D	Distance for defining accuracy
$\mathbf{D}$	Distance matrix
f	Frame index
fps	Frame rate in frames per second
F_{first}, F_{last}	First and last frames of stance phase
G	Graph representation of joint proposals
l_{ij}	Measured length between proposals
$l_{t_i t_j}$	Expected length between the types of proposals

n_{foot}	Number of foot proposals in a frame
p	Dot product used for determining direction of walking motion
$\mathbf{p}_{a,i}$, $\mathbf{p}_{b,i}$	Positions of feet a and b for stride i
$\mathbf{p}_{b,i,proj}$	Projected position of foot b for stride i
$\mathbf{p}_{centroid}$	Centroid of the head positions in a walking pass
$\mathbf{p}_{foot,1}$, $\mathbf{p}_{foot,2}$	The two selected foot positions before assigning left/right sides
$\mathbf{p}_{head,i}$, $\mathbf{p}_{head,f}$	Initial and final head positions of a walking pass
$\mathbf{p}_i$	Position of joint proposal i
$\mathbf{P}_f$	matrix containing the two foot positions on frame f
$\mathbf{P}_{foot}$	All positions of one foot in a walking pass
$\mathcal{P}$	Set of all joint proposals on a frame
$\mathcal{P}_{pair}$	Set of joint proposals in a pair of shortest paths
$\mathcal{P}_{paths}$	Set of proposals in all shortest paths to feet
r	Radius of spheres
S_{ij}	Score of the link between proposals i and j
t	Index of body part type
$\mathbf{v}_{2\rightarrow 1}$	Vector from foot 1 to foot 2
$\mathbf{v}_{forward}$	Vector representing direction of motion for a walking pass
$\mathbf{V}_{foot}$	Vectors from $\mathbf{p}_{centroid}$ to $\mathbf{P}_{foot}$
v_{side}	Value indicating left/right side of foot
x	Ratio between expected and measured lengths
X_A, X_B	Sets of measurements from devices A and B
w_{ij}	Weight of the graph edge between proposals i and j

Chapter 1

Introduction and Overview

1.1 Background

Gait analysis is the systematic study of how a person walks [1]. Given that gait impairment is a symptom of many diseases [2], gait analysis is an important component of clinical diagnosis and assessment.

Techniques for gait analysis can be categorized into two classes: semi-subjective and objective [3]. Semi-subjective techniques are usually conducted by a clinician who observes the subject performing a gait related task. For example, the Timed Up and Go test requires a subject to stand from sitting on a chair, walk a short distance, then return to sit on the chair [4]. By contrast, objective tests use various devices to gather quantitative measures of gait. These devices include wearable sensors [5], pressure-sensitive walkways [6], and vision-based systems [7].

Highly accurate vision-based systems use markers attached to the body. The 3D position of a marker can be calculated by detecting the same marker with two or more cameras [8]. However, these systems are often unavailable in clinics due to their high cost [9], and they require a controlled environment. As an alternative, markerless systems track the human body in plain clothing. Markerless human body tracking has been investigated using 2D cameras. Multi-view systems use multiple cameras to reconstruct a 3D representation of the body [10]. Gait analysis has also been performed with a single camera, relying on the properties of the gait task and proportions of the human body [11].

A depth sensor offers advantages for markerless gait analysis compared to a regular RGB (colour) camera. The 3D positions of body parts can be obtained with a single sensor rather than multiple. Collecting depth images instead of RGB also adds a degree of privacy for in-home gait analysis. In 2009, gait analysis was performed with a time-of-flight camera (the depth is calculated using the time required by a light pulse to return to the sensor) [12]. Pose estimation was performed by combining the depth map with a shape prior to solve a Markov random field. Interest in depth-based gait analysis has since increased with the release of the Kinect in 2010 [13].

The Kinect is an RGB-D (colour and depth) sensor produced by Microsoft. It was originally developed for motion-based gaming on the Xbox 360, and was first released in North America on November 4, 2010 [14]. The depth sensor on the Kinect v1 consists of an infrared projector and infrared camera, which together produce a depth image. Microsoft later released the Kinect v2 with a depth sensor based on time-of-flight technology [15]. The depth image resolutions are 640×480 and 512×424 for the Kinect v1 and v2, respectively, while v2 features a larger RGB resolution of 1920×1080. Research has shown that the accuracy of Kinect v1 decreases exponentially with distance, while the accuracy of v2 remains constant with an offset of -18 mm [16].

Gait analysis with the Kinect was introduced by Stone and Skubic in 2011 [17], using two Kinect sensors in a home environment. Gait parameters including stride length and velocity were approximated by analyzing the centre of mass of the person in view.

The Kinect Software Development Kit (SDK) for Windows was released on February 1, 2012 [14], providing researchers with a means to directly track a skeleton model containing individual human joints. Gait analysis with the Kinect SDK became common [18–24]. However, the SDK is limited to function from a frontal perspective, which adds inconvenience to clinical walking trials. A participant must either walk only towards the camera (as in [22]), or two cameras must be set up at the ends of the walking mat (as in [23]).

The overarching objective of this thesis is to perform clinical gait analysis with a side-view depth sensor. We use a Kinect v1 camera is used to capture depth images, but do not rely on the Kinect SDK. Gait analysis with a side-view Kinect has been previously explored [25, 26], but the methods do not track individual joints for both sides of the body (e.g., left and right feet). The tracking algorithm developed in this thesis results in separate left and right foot positions, allowing for the calculation of gait parameters using the standard equations of [27].

The presented work builds upon a predictor originally developed for overhead hand tracking [28], which is based on the original machine learning algorithm for the Kinect [29]. The predictor has been trained on side-view depth images of a walking person. It can now output multiple proposals for the 3D positions of individual human joints from a single depth image. The predictor is trained using raw depth images and corresponding label images. The pixels of the label images are labelled by body part, as shown in Figure 1.1.

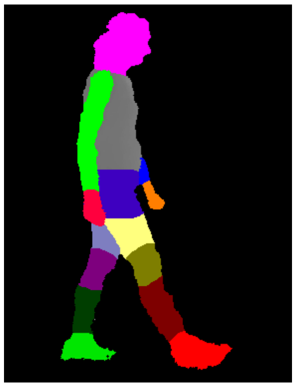


Figure 1.1: Example of label image used to train predictor.

The manual labelling of video data is a difficult and time-consuming process.

For this reason, the thesis first presents a method to fully label human parts in a depth image, given one seed pixel for each body part. The process of segmenting an image from seed pixels is referred to supervised segmentation [30]. We introduce a graph representation of the depth image that improves the accuracy of an established supervised segmentation algorithm.

The remaining work presents the development of a gait analysis system based on the joint proposals from the predictor. The predictor was already trained using a collection of manually labelled images, so our segmentation algorithm was not required for labelling. However, we suggest that future work uses the algorithm to produce training images from new depth data.

1.2 Thesis Outline and Contributions

The following chapters consist of four manuscripts and a conclusion.

Chapter 2 proposes a graph representation of human depth images with separate layers for the two arms. We show that human parts are segmented more accurately using this layered graph compared to a standard graph representation of an image. The manuscript was published in the peer-reviewed journal *Sensors* in 2018 [31].

Subjects with multiple sclerosis completed walking trials at The Recovery and Performance Laboratory of Memorial University. The participants completed multiple walking passes in both directions along a pressure-sensitive walkway, the Zeno Walkway. The trials were also recorded by a single Kinect v1 sensor pointed perpendicular to the walkway. Multiple joint proposals were generated on each depth image in the trials. The following three manuscripts use data from these trials.

Chapter 3 introduces a method to select optimal combinations of joints from the multiple proposals on an image frame, primarily by representing the proposals as a weighted graph and finding shortest paths through the graph. Data from previous frames are used to select the best joints on the current frame. The manuscript was accepted as a conference paper at the *International Symposium on Visual Computing* in 2016 [32].

Chapter 4 presents the preliminary results of conducting gait analysis with our system. By estimating accurate human joints based on Chapter 3, four gait parameters were calculated: stride length, step length, stride width, and stride velocity. The system was run on eight walking trials and validated against the Zeno Walkway. The manuscript was accepted as a conference paper at the *Newfoundland Electrical and Computer Engineering Conference* in 2017 [33].

Chapter 5 presents a method that improves upon both Chapter 3 and Chapter 4. While joint proposals are still represented as a weighted graph, the improved method employs a voting algorithm to select the best two foot positions on each frame using only spatial data (without previous frames). The feet are later assigned to left and right sides by estimating the general direction of walking motion. The new method calculates separate gait parameters for the left and sides of the body, while the parameters of Chapter 4 were not specific to a side. The new method is also capable of estimating stance percentage in addition to the parameters of Chapter 4. The system is tested on the full set of 52 walking trials measured concurrently by a Kinect and Zeno Walkway. The manuscript has been prepared for submission to a journal.

Chapter 6 presents the conclusions of the thesis and suggests ideas for further extending this research.

In summary, this thesis makes the following contributions:

1. A novel graph representation of a human depth image that improves the supervised segmentation of body parts.

2. A method of selecting accurate human joints from multiple proposals.
3. A method to calculate standard gait parameters from individual body parts with a side-view depth sensor.
4. Concurrent validation of our gait analysis compared to the Zeno Walkway.

1.3 Co-authorship Statement

I am the principal author of all manuscripts presented in this thesis, as well as the thesis as a whole. I developed the methods and analyzed the results in all manuscripts. Dr. Stephen Czarnuch revised all manuscripts and conceptualized the study. Data from the Zeno Walkway were provided by the Recovery and Performance Laboratory of Memorial University, under the direction of Dr. Michelle Ploughman. Megan Kirkland was involved with the collection of data at the Laboratory.

References

[1] M. W. Whittle, "Clinical gait analysis: A review," *Human Movement Science*, vol. 15, no. 3, pp. 369–387, 1996.

[2] V. Dietz, "Neurophysiology of gait disorders: present and future applications," *Electroencephalography and clinical Neurophysiology*, vol. 103, no. 3, pp. 333–355, 1997.

[3] A. Muro-De-La-Herran, B. Garcia-Zapirain, and A. Mendez-Zorrilla, "Gait analysis methods: An overview of wearable and non-wearable systems, highlighting clinical applications," *Sensors*, vol. 14, no. 2, pp. 3362–3394, 2014.

[4] S. Mathias, U. Nayak, and B. Isaacs, "Balance in elderly patients: the 'get-up and go' test." *Archives of physical medicine and rehabilitation*, vol. 67, no. 6, pp. 387–389, 1986.

[5] B. Mariani, C. Hoskovec, S. Rochat, C. Büla, J. Penders, and K. Aminian, "3d gait assessment in young and elderly subjects using foot-worn inertial sensors," *Journal of biomechanics*, vol. 43, no. 15, pp. 2999–3006, 2010.

[6] U. Givon, G. Zeilig, and A. Achiron, "Gait analysis in multiple sclerosis: characterization of temporal–spatial parameters using gaitrite functional ambulation system," *Gait & posture*, vol. 29, no. 1, pp. 138–142, 2009.

[7] A. Leu, D. Ristić-Durrant, and A. Gräser, "A robust markerless vision-based human gait analysis system," in *Applied Computational Intelligence and Informatics (SACI), 2011 6th IEEE International Symposium on.* IEEE, 2011, pp. 415–420.

[8] R. Baker, “Gait analysis methods in rehabilitation,” *Journal of neuroengineering and rehabilitation*, vol. 3, no. 1, p. 4, 2006.

[9] A. Pfister, A. M. West, S. Bronner, and J. A. Noah, “Comparative abilities of microsoft kinect and vicon 3d motion capture for gait analysis,” *Journal of medical engineering & technology*, vol. 38, no. 5, pp. 274–280, 2014.

[10] F. Caillette and T. Howard, “Real-time markerless human body tracking with multi-view 3-d voxel reconstruction,” in *Proc. BMVC*, vol. 2. Citeseer, 2004, pp. 597–606.

[11] M. Goffredo, R. D. Seely, J. N. Carter, and M. S. Nixon, “Markerless view independent gait analysis with self-camera calibration,” 2008.

[12] R. R. Jensen, R. R. Paulsen, and R. Larsen, “Analyzing gait using a time-of-flight camera,” in *Scandinavian Conference on Image Analysis*. Springer, 2009, pp. 21–30.

[13] S. Springer and G. Yogev Seligmann, “Validity of the kinect for gait assessment: A focused review,” *Sensors*, vol. 16, no. 2, p. 194, 2016.

[14] Z. Zhang, “Microsoft kinect sensor and its effect,” *IEEE multimedia*, vol. 19, no. 2, pp. 4–10, 2012.

[15] L. Yang, L. Zhang, H. Dong, A. Alelaiwi, and A. El Saddik, “Evaluating and improving the depth accuracy of kinect for windows v2,” *IEEE Sensors Journal*, vol. 15, no. 8, pp. 4275–4285, 2015.

[16] O. Wasenmüller and D. Stricker, “Comparison of kinect v1 and v2 depth images in terms of accuracy and precision,” in *Asian Conference on Computer Vision*. Springer, 2016, pp. 34–45.

[17] E. E. Stone and M. Skubic, "Evaluation of an inexpensive depth camera for passive in-home fall risk assessment," in *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2011 5th International Conference on*. Ieee, 2011, pp. 71–77.

[18] J. Behrens, C. Pfüller, S. Mansow-Model, K. Otte, F. Paul, and A. U. Brandt, "Using perceptive computing in multiple sclerosis-the short maximum speed walk test," *Journal of neuroengineering and rehabilitation*, vol. 11, no. 1, p. 89, 2014.

[19] M. Gabel, R. Gilad-Bachrach, E. Renshaw, and A. Schuster, "Full body gait analysis with kinect," in *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*. IEEE, 2012, pp. 1964–1967.

[20] F. Ahmed, P. Polash Paul, and M. L. Gavrilova, "Kinect-based gait recognition using sequences of the most relevant joint relative angles," 2015.

[21] A. A. Chaaraoui, J. R. Padilla-López, and F. Flórez-Revuelta, "Abnormal gait detection with rgb-d devices using joint motion history features," in *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, vol. 7. IEEE, 2015, pp. 1–6.

[22] S. Motiian, P. Pergami, K. Guffey, C. A. Mancinelli, and G. Doretto, "Automated extraction and validation of children's gait parameters with the kinect," *Biomedical engineering online*, vol. 14, no. 1, p. 112, 2015.

[23] E. Dolatabadi, B. Taati, and A. Mihailidis, "Concurrent validity of the microsoft kinect for windows v2 for measuring spatiotemporal gait parameters," *Medical engineering & physics*, vol. 38, no. 9, pp. 952–958, 2016.

[24] X. Xu, R. W. McGorry, L.-S. Chou, J.-h. Lin, and C.-c. Chang, "Accuracy of the microsoft kinect™ for measuring gait parameters during treadmill walking," *Gait & posture*, vol. 42, no. 2, pp. 145–151, 2015.

[25] G. Baldewijns, G. Verheyden, B. Vanrumste, and T. Croonenborghs, "Validation of the kinect for gait analysis using the gaitrite walkway," in *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE.* IEEE, 2014, pp. 5920–5923.

[26] E. Cippitelli, S. Gasparrini, S. Spinsante, and E. Gambi, "Kinect as a tool for gait analysis: validation of a real-time joint extraction algorithm working in side view," *Sensors*, vol. 15, no. 1, pp. 1417–1434, 2015.

[27] F. Huxham, J. Gong, R. Baker, M. Morris, and R. Iansek, "Defining spatial parameters for non-linear walking," *Gait & posture*, vol. 23, no. 2, pp. 159–163, 2006.

[28] S. Czarnuch and A. Mihailidis, "Development and evaluation of a hand tracker using depth images captured from an overhead perspective," *Disability and Rehabilitation: Assistive Technology*, vol. 11, no. 2, pp. 150–157, 2016.

[29] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on.* Ieee, 2011, pp. 1297–1304.

[30] L. Grady, "Random walks for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 11, pp. 1768–1783, 2006.

[31] A. Hynes and S. Czarnuch, "Human part segmentation in depth images with annotated part positions," *Sensors*, vol. 18, no. 6, p. 1900, 2018.

[32] A. Hynes and S. Czarnuch, "Combinatorial optimization for human body tracking," in *International Symposium on Visual Computing.* Springer, 2016, pp. 524–533.

[33] A. Hynes, M. C. Kirkland, M. Ploughman, and S. Czarnuch, "Comparing the gait analysis of a kinect system to the zeno walkway: Preliminary results," in *Newfoundland Electrical and Computer Engineering Conference*, 2017.

Chapter 2

Human Part Segmentation in Depth Images with Annotated Part Positions

Andrew Hynes and Stephen Czarnuch

Abstract

We present a method of segmenting human parts in depth images, when provided the image positions of the body parts. The goal is to facilitate per-pixel labelling of large datasets of human images, which are used for training and testing algorithms for pose estimation and automatic segmentation. A common technique in image segmentation is to represent an image as a two-dimensional grid graph, with one node for each pixel and edges between neighbouring pixels. We introduce a graph with distinct layers of nodes to model occlusion of the body by the arms. Once the graph is constructed, the annotated part positions are used as seeds for a standard interactive segmentation algorithm. Our method is evaluated on two public datasets containing depth images of humans from a frontal view. It produces a mean per-class accuracy of 93.55% on the first dataset, compared to 87.91% (random forest and graph cuts) and 90.31% (random forest and Markov random field). It also achieves a per-class accuracy of 90.60% on the second dataset. Future work can experiment with various methods for creating the graph layers to accurately model occlusion.

2.1 Introduction

Segmenting the parts or limbs of a human body has applications for pose estimation [1], robotic assistance [2] and gaming [3]. Recent research on human limb segmentation has used machine learning techniques to perform per-pixel classification of RGB (colour) and/or depth images, without any pre-annotation. However, these methods often require manually segmented images for training and testing. Systems for pose estimation and recognition also make use of manually segmented images. For large datasets, the manual annotation of human limbs becomes laborious. This paper introduces a method for assisting with the semi-supervised segmentation of humans in depth images. A user specifies one (x, y) position for each body part in a depth image, and our system classifies each pixel on the body with an appropriate label.

In this problem, we are provided a depth image, a foreground segmentation of the human figure and a set of image positions for the body parts. The objective is to assign each pixel in the foreground to a body part. Our approach is to use the part positions in conjunction with an interactive image segmentation algorithm.

Interactive image segmentation involves an n-dimensional image, which is classified into k distinct regions. A user specifies seed pixels (or voxels) in regions of interest that are given a label from one to k. The regions are segmented using the values of the pixels. In this application, k is the number of body parts, and the pixels have only depth values.

While interactive segmentation algorithms can segment an image into logical regions, it can be difficult to apply them to human part segmentation, due to the presence of self-occlusion: when some regions of the body are occluded by others. In response, we use a graph representation that is designed to handle occlusion. The assumption is made that the arms are the only regions that can potentially occlude other parts. We create a grid graph with one to three layers: representing the base body, the right arm and the left arm. In this formulation, a single pixel in the image can be represented by more than one node in the graph. A standard interactive segmentation algorithm can then be applied to this layered graph, resulting in a more accurate segmentation of human limbs.

The paper is organized as follows. Section 2.2 discusses recent work towards human limb segmentation and provides an overview of some standard interactive image segmentation algorithms. Section 2.3 describes the process of creating a layered graph representation of a human depth image and determining seed nodes and labels that are used for interactive segmentation. Section 2.4 presents experimental results on two public datasets of human depth images [4, 5]. The conclusions are presented in Section 2.5.

2.2 Related Works

2.2.1 Human Limb Segmentation

Relevant literature commonly refers to segmenting regions of the body as human limb segmentation. We refer to our system as human part segmentation, to make the distinction between limbs (e.g., arm) and parts (e.g., shoulder, elbow and hand).

Segmenting human limbs has been applied to single RGB images [6, 7], RGB video sequences [8] and images combining colour and depth, referred to as RGB-D [2, 9]. It has also been performed on pure depth images and videos [4, 10].

Human limb segmentation from RGB-D data has seen recent application to mobility assistance robots [2]. Using RGB-D images provided by a Kinect camera, the limbs are segmented with a deep learning approach. The researchers also developed a method to abbreviate the manual per-pixel annotation of limbs. By computing histogram of oriented gradient (HOG) features on each image, they were able to cluster images using pairwise distances in HOG space. One image would be randomly selected from the cluster and manually segmented using an image editing tool. The segmentation would then be propagated to the other images in the cluster.

The authors in [4] introduced a general method for object segmentation in depth images and applied this to human limb segmentation. Building on the pose estimation algorithm developed for the Kinect [11], per-pixel classification was performed with a learned random forest classifier. The classification was further optimized using graph cuts and tested on a new dataset of RGB and depth images with per-pixel segmented limbs. This approach built off their earlier work [12], which performed limb segmentation with an interacting user, focusing on correcting user inputs using graph cuts.

The work of [10] involved segmenting parts of the hand in a depth image. Similar to [4], a random forest was used for the initial per-pixel classification. Instead of graph cuts, the pixel classification was optimized by partitioning the hand into superpixels and smoothing with a Markov random field. The method was shown to generalize to human part segmentation by testing on the dataset from [4].

The algorithms of [2, 4, 10] all required segmented datasets for training. The intention of our work is to assist with creating these datasets using interactive segmentation. While [12] used an interactive segmentation algorithm for human limbs, they used a standard grid graph structure to represent the image, and required multiple seed pixels for each segment. Our main contribution is the proposal of a layered graph designed to handle occlusion, so that only one seed pixel is required for an interactive segmentation algorithm.

2.2.2 Interactive Image Segmentation

The following algorithms were developed for general interactive image segmentation. They all share the properties of operating on a grid structure, such as a graph or lattice. The segmentation results depend on user-defined seeds and labels and the difference in value between pixels in the same local neighbourhood.

Graph Cuts

The original graph cuts algorithm [13] was intended for segmenting an image into two regions. The grid graph representing pixels is transformed into a flow network by introducing a source and sink node. Each pixel node has an incoming edge from the source and an outgoing edge to the sink. Seed pixels are specified as belonging to either the background or foreground. Foreground seeds are severed from the sink node, and background seeds are severed from the source. By solving the max-flow/min-cut problem, a set of edges is found that partitions the graph into two. All pixels connected to the source are labelled as foreground, and vice versa.

Random Walker

The random walker (RW) algorithm [14] was designed for multi-label image segmentation, rather than only foreground/background. Similar to graph cuts, a grid graph is used to represent the image. The edges between pixels are weighted from zero to one, representing the probability that a random walker on the graph will cross that edge. The weight of an edge $u \leftrightarrow v$ is inversely proportional to the difference between the values of pixels u and v, i.e., a random walker is more likely to cross an edge between similar pixels. Each seed pixel is given a label from one to k.

The algorithm makes use of an equivalence theorem between random walks on a graph and electric potentials in a grid circuit. Each edge in the graph is represented by a resistance that is the inverse of the edge weight. For each label i, the seed pixels labelled i are given a unit potential while the other seeds are set to zero. The resulting potential at each unlabelled pixel u is the probability that a random walker beginning on u will first reach a seed pixel with label i. Once all k probabilities have been calculated, the pixel is assigned the label with the highest probability.

By representing the problem as an electric circuit, the image is segmented by solving a series of linear equations, rather than actually simulating a random walk.

GrowCut

Like the RW algorithm, GrowCut [15] is designed to handle multi-label segmentation. However, it does not use an explicit graph structure to segment the image. Instead, the image is treated as a cellular automaton, which evolves over time. A cellular automaton has three basic properties: a non-empty state set S, a local neighbourhood N that defines adjacent pixels and a transition function $S^N \rightarrow S$ that determines the state set in the subsequent iteration. Seed pixels are labelled from one to k in the image. At each time step, a transition function is applied to pixels in the local neighbourhood of the seeds, which allows non-seed pixels to be labelled. The authors make the analogy of k competing bacteria cultures, which grow from their seed positions and attempt to claim new pixels as territory. The transition function is designed so that the competing populations eventually reach a stable configuration, segmenting the full image.

Our review of related works has shown that while automated human segmentation systems exist, they still require annotated datasets for training [2, 4, 10]. We aim to simplify this annotation by applying a standard interactive segmentation algorithm to a novel graph representation of the human body. Thus, our method is a hybrid of manual and automatic annotation, that facilitates the creation of large training datasets.

GrowCut and RW were both initially considered for this application. RW was eventually chosen for its speed of computation and tendency to produce smooth boundaries between regions. As described in the following section, it is used twice: once for segmenting the arms to create the layered graph and finally to produce the output segmentation.

2.3 Methodology

2.3.1 Grid Graph

The depth image is converted into an undirected grid graph, with one node for each pixel in the foreground. The pixel at row i, column j is connected to its four-neighbourhood in the image: the pixels at $(i, j+1)$, $(i, j-1)$, $(i+1, j)$ and $(i-1, j)$. The connection is represented by an edge in the grid graph. For an undirected edge $u \leftrightarrow v$, the weight is:

$$w_{uv} = \exp\bigl(-\beta(i_u - i_v)^2\bigr) \tag{2.1}$$

where i_u denotes the value at pixel u. This is the standard weighting function for RW [14]. In this case, i_u is the depth at pixel u. The parameter β determines the significance of the difference in depth values. The $(i_u - i_v)^2$ values are first normalized across the image, as suggested in [14].

2.3.2 Arm Probability Matrix

We first aim to segment the two arm regions, as we assume these are the regions that are potentially occluding other parts of the body.

As described in Section 2.2.2, the RW algorithm produces a probability matrix for each seed label. The algorithm is first run to obtain a probability matrix for each arm. The two hand pixels are used as seeds. Running RW using these two seed pixels, labelled 1 and 2, results in two probability matrices. However, if an arm is in front of the torso, the probability values can be skewed to be close to either zero or one. This can be seen in Figure 2.1. The depth difference of the right arm acts as a barrier to a random walker. As a result, even pixels on the head are given a probability of nearly one for belonging to the right hand.

To increase the variance in probability values, an extra 'dummy' node is added to the graph. This node is connected to every node in the grid graph by an edge with a small weight w, as shown in Figure 2.2. A value of $w = 1 \times 10^{-3}$ was determined through experiment. This represents the probability that a random walker will cross the edge and land on the dummy node. The RW algorithm is run on this new graph structure using three seed nodes: the two hand nodes and the dummy node, labelled 1 to 3. The output probability values for the hand nodes now have more variance, as evident in Figure 2.1. The new matrix P for hand X is used to segment arm X (left or right).

2.3.3 Arm Segmentation

If the arm is occluding the body, there will be a significant difference in depth between pixels on the arm and neighbouring pixels on the body. This will also be evident in the probability matrix P found in Section 2.3.2.

The probability values of P are clustered using the mean shift algorithm [16] with a Gaussian kernel. The advantage of mean shift is its ability to find clusters without pre-specifying the number of clusters. For efficiency, only a small sample of

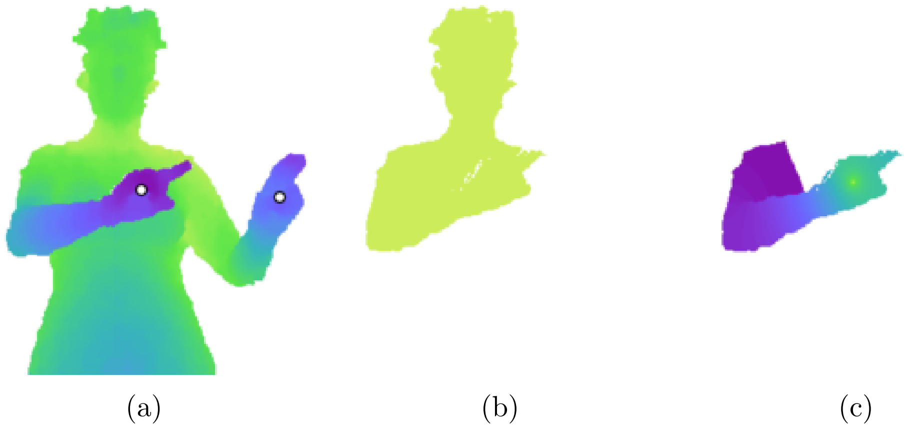


Figure 2.1: Effect of adding a dummy node. (a) Depth image. The white dots indicate the annotated hand pixels. (b) Probability values for the right hand, using the two hand pixels as seeds for the RW algorithm. The values are close to either zero or one. (c) Probability values after adding a dummy node to the graph. There is now a greater variance in the values.

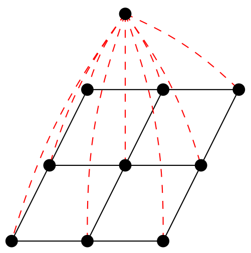


Figure 2.2: Diagram of a dummy node being added to the grid graph. Each node in the grid represents a pixel in the depth image. The dummy node is connected to each grid node by a single edge (dashed line).

probability values is clustered. After sorting the probabilities, n uniformly-spaced values are taken from the sorted list and clustered with mean shift, resulting in k clusters. The value of n can be tuned by the user for their specific dataset. The value k is determined automatically by mean shift.

A new image I_{seg} is the result of segmenting the probability matrix with mean shift. Each pixel in the foreground is assigned to the cluster with the closest centroid value. Figure 2.3 shows an example of this segmented image.

An iterative process is used to find the arm segment that minimizes a cost function,

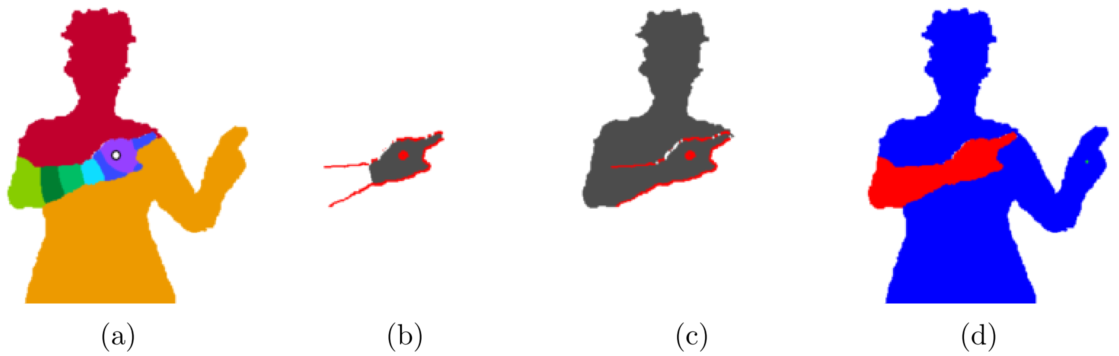


Figure 2.3: Segmenting the arms. (a) Segmented image resulting from clustering probability values of the right arm with mean shift. (b) Potential arm segment (grey) and strong gradient pixels (red). While this binary arm segment B_i does not cut across the gradient, its complement $\sim B_i$ does. This invokes a high cost. (c) Another potential arm segment. The segment B_i cuts across the gradient, also invoking a high cost. (d) Final arm segments (the left segment is minuscule). Only the right arm is valid.

as explained below. Beginning with the segment on I_{seg} corresponding to the highest cluster (i.e., the segment containing the hand pixel), segments are added in order of descending probability. At each iteration i, the full arm segment is the union of segments in I_{seg} from one to i.

The image gradient captures information about the local change in pixel values [17]. It returns an (x, y) vector at each pixel location in an image. I_{grad} is the magnitude of the image gradient of the probability matrix P (with all background pixels of P set to null). An example is shown in Figure 2.3. The highest values in I_{grad} occur at sharp differences in pixel probability, corresponding to the depth difference caused by the arm occluding the body. The cost function for the arm segment uses this image gradient.

The binary arm segment B_i is eroded with the structuring element for the four-neighbourhood.

$$\text{erosion}(B_i) = B_i \ominus \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{2.2}$$

A new binary image $B_{union,\ i}$ is the union of the eroded B_i and the erosion of its complement image, i.e., all image pixels not in B_i, including background pixels.

$$B_{union,\ i} = \text{erosion}(B_i) \cup \text{erosion}({\sim}B_i) \tag{2.3}$$

The cost of B_i is the mean of gradient values inside $B_{union,\ i}$.

$$\text{cost}(B_i) = \text{mean}(I_{grad}(B_{union,\ i})) \tag{2.4}$$

In essence, an arm segment B_i that is too small will cause ${\sim}B_i$ to cut across a strong gradient line, while a B_i that is too large will cut across a strong gradient line itself. This is evident in Figure 2.3. The segments are eroded to avoid the gradient values along their perimeters.

Algorithm 1 shows the pseudocode for the iterative process that segments the arm. It selects the arm segment that minimizes the cost defined in Equation (2.4). After the two arm segments have been found, there may be pixels that belong to both segments. Each of these pixels are assigned to the side with the higher RW probability.

2.3.4 Validation of Arm Segments

Once the arms have been segmented, they are evaluated for evidence that they are occluding the body. An occluding arm segment is referred to as valid. In the case

Algorithm 1 Arm Segmentation

```
1: Sort values of probability matrix $P$
2: Select $n$ evenly-spaced samples from sorted list
3: $I_{seg} \leftarrow$ Image segmented by applying mean shift to probability samples
4: $I_{grad} \leftarrow$ Image gradient of $P$

5: function ArmCost($B$, $I_{grad}$)
6:     $E_1 \leftarrow$ erosion of binary image $B$
7:     $E_2 \leftarrow$ erosion of complement image $\sim B$
8:     $E_{union} \leftarrow E_1 \cup E_2$
9:     return mean($I_{grad}(E_{union})$)

10: function SegmentArm($I_{seg}$, $I_{grad}$)
11:     $C_{min} \leftarrow$ infinity
12:     for $i$ in $k$ clusters do
13:         $B_i \leftarrow \bigcup I_{seg} == \{1, ..., i\}$
14:         $C \leftarrow$ ArmCost($B_i$, $I_{grad}$)
15:         if $C < C_{min}$ then
16:             $C_{min} \leftarrow C$
17:             $B_{arm} \leftarrow B_i$
18:     return $B_{arm}$
```

that an arm is not occluding the body, there can be an insignificant depth difference between the body and the arm.

B_{arm} is the arm segment determined from Section 2.3.3. F_{other} is the rest of the foreground F, including the other arm.

$$F_{other} = F \cap \sim B_{arm} \tag{2.5}$$

B_{inner} is the set of pixels in B_{arm} that are in the four-neighbourhood of F_{other}.

This set is found by dilating F_{other} and taking the intersection with B_{arm}.

$$B_{inner} = \left(F_{other} \oplus \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \right) \cap B_{arm} \tag{2.6}$$

Similarly, B_{outer} is the set of pixels in F_{other} that are in the four-neighbourhood of B_{arm}. Figure 2.4 shows the inner and outer pixels for the right arm segment.

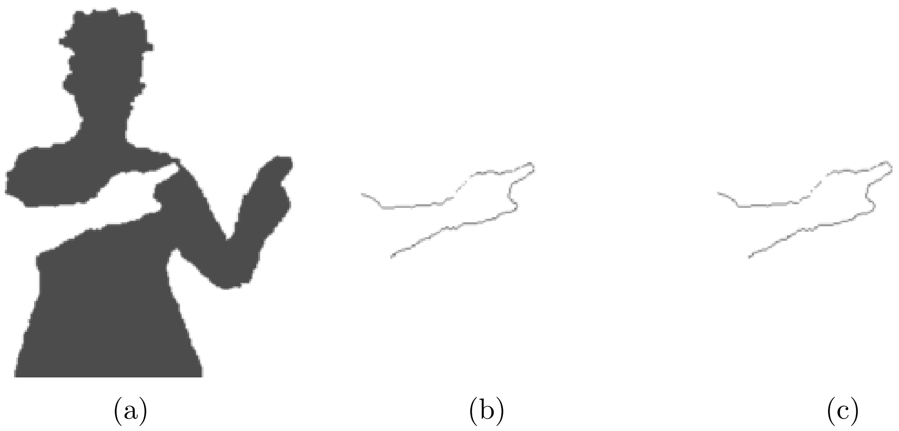


Figure 2.4: Relevant binary images. (a) F_{other} is the set of foreground pixels outside the arm segment (in this case, the right arm). (b) B_{inner} is the set of pixels inside the arm and neighbouring F_{other}. (c) B_{outer} is the set of pixels inside F_{other} and neighbouring the arm.

If the arm is occluding the body, there should be a significant difference between the depths of pixels in B_{inner} and B_{outer}, and the outer depths should be greater than the inner. There should also be a significant probability difference. Let D_{inner} be the set of depths of pixels in B_{inner} and P_{inner} be the set of probabilities. An arm segment is valid only if it is whole (i.e., only one connected component) and if it meets both of the following two conditions:

$$\text{median}(D_{outer}) - \text{median}(D_{inner}) \geq 1\,\text{cm} \tag{2.7}$$

$$\text{median}(P_{inner}) - \text{median}(P_{outer}) \geq 1\% \tag{2.8}$$

In the case shown in Figure 2.3, the right arm is valid and the left is not.

2.3.5 Layered Grid Graph

After the arms have been segmented, a new grid graph G is created with one to three 'layers': a base layer G_B, a right arm layer G_R and a left arm layer G_L. Each layer is a distinct set of nodes. A layer is created for an arm only if the segment was found to be valid (see Equation (2.7) and Equation (2.8)).

The purpose of this layered graph is to emulate a body with the arms outstretched, rather than occluding the mid-region. The base layer G_B is a grid graph with one node for every foreground pixel, including the pixels that are in the right or left arm segments. A pixel inside binary arm segment $B_{arm,\ X}$ is represented by a node in G_B and a node in G_X.

When all three layers are constructed, the graph G has n_G nodes.

$$n_G = n_F + n_R + n_L \tag{2.9}$$

where n_F is the number of foreground pixels, n_R is the number of pixels in arm segment $B_{arm,R}$ and n_L is the number of pixels in arm segment $B_{arm,L}$.

In the regions of the arms, nodes in G_B are given interpolated depth values, which are used to compute edge weights. This allows a segmentation algorithm to be unaffected by the sharp depth difference caused by occluding arms. To interpolate the depth values, every background (i.e., non-human) pixel is given the same depth, which is the median of all foreground depths. Then, the arm regions of the base layer are filled with inward interpolation. The edges of G are weighted using Equation (2.1).

2.3.6 Connecting Graph Layers

When the graph layers G_B, G_R and G_L are first constructed, there are no edges between layers. A set of nodes is selected that will connect G to G_R and another set to connect G to G_L.

$I_{grad,\ inner}$ is the set of gradient values on B_{inner} (from Equation (2.6)).

$$I_{grad,\ inner} = I_{grad}(B_{inner}) \tag{2.10}$$

Similar to the clustering in Section 2.3.3, the values of $I_{grad,\ inner}$ are clustered with mean shift. Along the inner pixels of the arm segment, the gradient is lowest where the arm connects to the body. $B_{connect}$ is the set of pixels in B_{inner} corresponding to the lowest value cluster found by mean shift. These connecting pixels are shown in Figure 2.5.

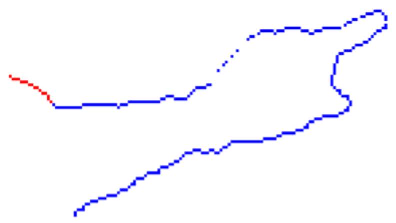


Figure 2.5: Pixels on the perimeter of an arm segment and adjacent to the rest of the foreground (binary image B_{inner}, also shown in Figure 2.4). The pixels used to connect graph layers (binary image $B_{connect}$) are shown in red.

Each pixel in $B_{connect,X}$ has two nodes associated with it: node u in the base graph G_B and node v in the arm graph G_X. An edge $u \leftrightarrow v$ is inserted with unit weight. This is repeated for each pixel $p \in B_{connect,X}$.

2.3.7 Assigning Body Parts to Layers

Every annotated part position must be assigned to its correct layer in the graph G. Each part is initially assumed to belong to the base layer of G. The forearm parts of arm X (e.g., hand and elbow) are assumed to be the only parts that can occlude the body; therefore, they are the only candidates for belonging to layer X. If the segmentation for arm X is valid, the hand of this arm must belong to layer X in the graph. The elbow is assigned to layer X only if its position is inside the arm segment $B_{arm,\ X}$.

2.3.8 Seed Nodes and Labels

In order to run an interactive segmentation algorithm on the graph, a subset of nodes must be specified as seeds. The provided body part positions are used as seed nodes, with labels one to n_{parts}.

More seed nodes can be added by drawing lines between adjacent body parts. Bresenham's algorithm [18] is used to find the pixels that constitute a line between two image positions A and B. This line L_{AB} is now split into L_A and L_B, i.e., pixels in L_A and L_B are given labels A and B, respectively. The user specifies a value r, which is the ratio of the length of L_A to the total length of L_{AB}. This allows the user to alter the size of the final part segment.

Each pixel in L_{AB} is assigned to a node in the layered graph. The layers for each body part position have been found by the process described in Section 2.3.7. Each seed pixel with label i is assigned to the layer of part i. Thus, seed pixels on the image are converted to seed nodes in the layered graph.

2.3.9 Final Segmentation

Once the seed nodes and labels have been determined, an interactive image segmentation algorithm can be run on the layered graph G. The RW algorithm is used for the final segmentation. Every node in the graph is given a label by the algorithm. The final output image has the same size as the original depth image, and each pixel in the foreground receives a label. Pixels in arm segment X are represented by a node in G_B and a node in G_X. The final label of these pixels is the label of the node in G_X. The label of the other node in G_B is ignored.

2.4 Experimental Results

2.4.1 Datasets

Our method is tested on two public datasets of human depth images with segmented parts: "Human Limbs from RGBD data" [4] (see Figure 2.7) and "Human Depth Images with Body Part Labels" [5] (see Figure 2.8). These are referred to as Datasets 1 and 2. In both datasets, only the foreground depth values are available.

Dataset 1 includes two actors in three video sessions, performing gestures in front of a Kinect v1 camera. Only the upper body is shown. The accompanying ground truth images have seven labels: torso, right/left upper arm, right/left lower arm and right/left hand. Each frame is a 640×480, 12-bit depth image. The three sessions have 236, 156 and 100 frames, respectively, for a total of 492 frames. Corresponding RGB images are included in the dataset, but our method is tested only on the depth images.

Our method is tested on one case from Dataset 2: sitting poses viewed from the front. This set contains nearly 5000 depth images of the full human body. Each frame is a 212×256 image, and 11 body parts are segmented.

2.4.2 Experiment Setup

The RW algorithm is performed using an open source MATLAB program written by the author of the algorithm [19]. In this implementation, the default value for the weighting parameter of Equation (2.1) is $\beta = 90$. Our method uses this value of β for the arm segmentation and final segmentation.

Each input body part position is annotated using the ground truth labels of the test dataset. For each part i, the annotated part position is the foreground pixel closest to the centroid of all pixels labelled i (the pixel is not required to have truth label i itself).

To show that our method offers an improvement over generic interactive image segmentation, the standard RW algorithm is run on all frames with the annotated part positions as seed pixels, using the same β value.

Dataset 1

Some additional pre-processing is used for this dataset. As an exception to the annotation rule defined above, the torso pixel is the mean position of the left and right upper arm pixels, whenever both of these arm parts exist. Some of the depth images include abnormally high depth values along the outline of the human figure, interfering with the RW algorithm. Therefore, the depth images are pre-processed to remove pixels with values greater than 2.5 metres. The same pixels are removed from the truth images.

Two versions of our method are tested. The first (referred to as 'our method, no line') uses only the seven part positions as seeds for the segmentation. The second (referred to as 'our method, with line') adds a line of seed pixels between the torso and the two upper arm positions, as per Section 2.3.8. The ratio r is set to $2/3$ for all frames. This is intended to limit the size of the upper arm segments. An example is shown in Figure 2.6.

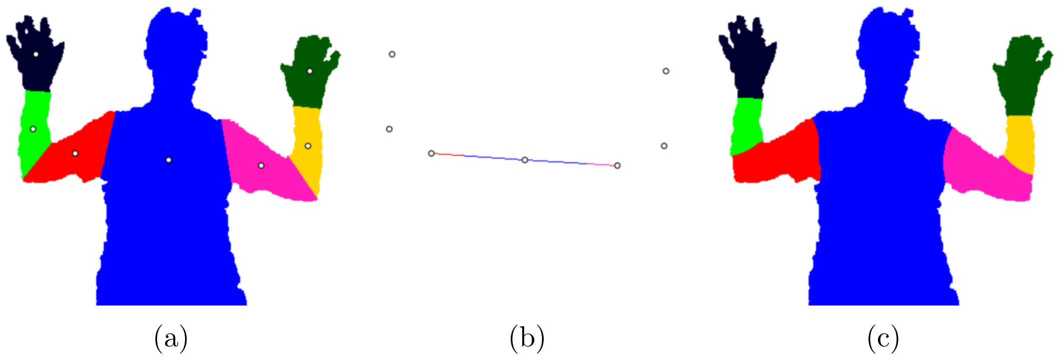


Figure 2.6: Segmenting with line seed pixels. (a) Ground truth image and part positions. (b) A line of seed pixels is drawn from the torso to each upper arm position. The first 2/3 of the line is labelled torso, and the remaining pixels are labelled L/R upper arm. Each other part i receives a single seed pixel. (c) Final segmentation.

The results of our method are also compared to the best performance of two algorithms tested on the same dataset [4, 10].

Dataset 2

Because of the relative positions of the ground truth labels for this dataset, it was decided that adding a line of seed pixels was unnecessary. Thus, only the simple version of our method is tested on this dataset, using the 11 annotated part positions as seed pixels.

2.4.3 Evaluation

Two metrics are used to evaluate our method: the per-class accuracy **PC** and the Jaccard index **JI**. These are calculated as described in [20]. The confusion matrix

$\mathbf{C}$ is computed using the ground truth label image and the predicted label image. $\mathbf{C}_{ij}$ is the number of pixels with truth label i and predicted label j. The matrix has dimensions $L \times L$, where L is the number of labels.

The per-class accuracy for label i is the number of pixels correctly labelled i over the total number of pixels labelled i. This metric has been previously used to report results on Dataset 1 [4, 10].

$$\mathbf{PC}_i = \frac{\mathbf{C}_{ii}}{\sum_j \mathbf{C}_{ij}} \tag{2.11}$$

The Jaccard index for label i is the number of pixels labelled i in both images over the number of pixels labelled i in either image, i.e., the intersection over union. This has been the standard evaluation metric for the PASCAL challenge since 2008 [20], which is a benchmark competition for object classification and detection [21].

$$\mathbf{JI}_i = \frac{\mathbf{C}_{ii}}{-\mathbf{C}_{ii} + \sum_i \mathbf{C}_{ij} + \sum_j \mathbf{C}_{ij}} \tag{2.12}$$

Pairwise Comparisons

To evaluate one method against another, the mean **PC** and **JI** values are calculated for each frame and by both methods. Let V_A and V_B be the mean metric values for a single frame segmented by methods A and B, respectively. A win for method A is recorded when $V_A > V_B$, a loss when $V_A < V_B$ and a tie when $V_A = V_B$. The differential is $V_A - V_B$.

2.4.4 Results

Dataset 1

Qualitative results are shown in Figure 2.7. The first column shows the depth images and annotated part positions, and the second shows the arm segmentations. The following columns display the three segmentation methods that were tested on Dataset 1.

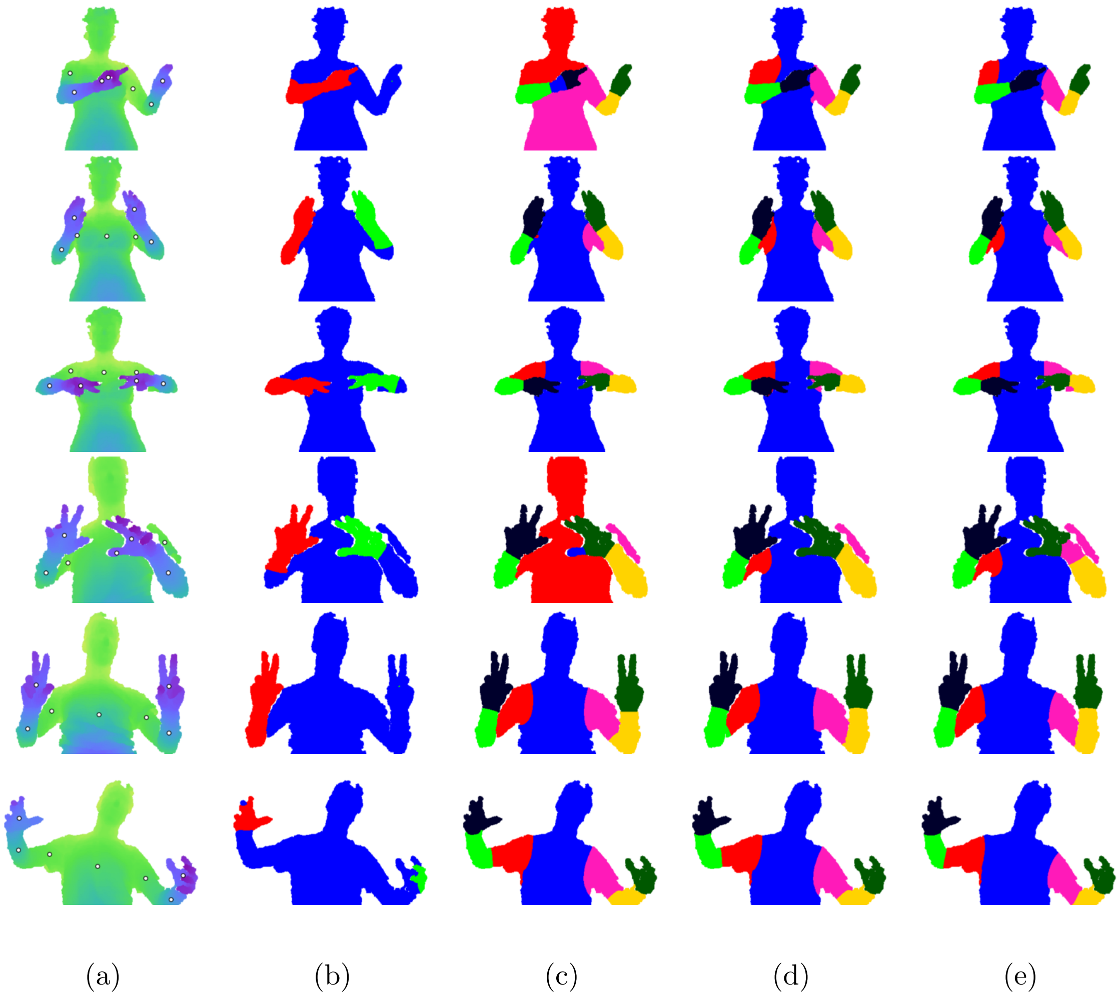


Figure 2.7: Dataset 1 [4]: Qualitative results. (a) Depth image with annotated part positions. (b) Arm segmentation. (c) Standard RW segmentation. (d) Our method without line seed pixels. (e) Our method with line seed pixels.

Table 2.1 shows the mean Jaccard index for each body part. We use the same abbreviations as [4]: L, left; R, right; U, upper; W, lower. Both versions of our method have higher overall averages (82.27% and 86.01%) than the standard RW algorithm (80.31%). Adding a line of seed pixels, as described in Section 2.4.2, results in a higher average, with the largest gains in the upper arm segments.

Table 2.1: Dataset 1 [4]: Mean Jaccard index in % over all frames. U, upper; W, lower.

	Torso	**LU arm**	**LW arm**	**L hand**	**RU arm**	**RW arm**	**R hand**	**Average**
RW	91.83	60.31	82.31	92.79	60.65	82.48	91.77	80.31
Our method, no line	95.81	65.04	81.85	92.69	66.62	81.63	92.25	82.27
Our method, with line	97.73	74.98	84.81	92.55	75.84	84.11	92.03	86.01

The mean per-class accuracy is shown in Table 2.2. In this case, adding line seed pixels causes a slight decrease in performance. In terms of the overall average, both versions of our method outperform the standard RW, as well as the two cited algorithms.

Table 2.2: Dataset 1 [4]: Mean per-class accuracy in % over all frames.

	Torso	**LU arm**	**LW arm**	**L Hand**	**RU arm**	**RW arm**	**R hand**	**Average**
Hernández-Vela et al. [4]	98.44	78.93	84.38	88.32	82.57	88.85	93.86	87.91
Liang et al. [10]	94.10	93.57	90.43	87.31	93.13	87.84	85.79	90.31
RW	92.29	85.06	95.71	96.69	85.65	95.46	95.75	92.37
Our method, no line	96.13	87.46	95.66	96.02	88.59	95.45	95.53	93.55
Our method, with line	98.65	87.67	92.84	95.92	91.40	91.47	95.48	93.35

Tables 2.3 and 2.4 show the results of pairwise comparisons. The total differential is the sum of differentials over all frames, i.e., a positive value indicates a net advantage of using method A over method B. For both metrics, the two versions of our method outperform the standard RW algorithm. Examining the total differentials, adding line seeds causes a slight decrease in per-class accuracy (-0.99%), but a large increase in the Jaccard index (18.40%).

Table 2.3: Dataset 1 [4]: Pairwise comparison of methods with the Jaccard index. A win indicates that method A outperformed method B on one frame.

Method A	**Method B**	**Wins**	**Losses**	**Ties**	**Total Differential (%)**
Our method, no line	RW	290	165	37	9.67
Our method, with line	RW	432	60	0	28.06
Our method, with line	Our method, no line	436	55	1	18.40

Table 2.4: Dataset 1 [4]: Pairwise comparison of methods with per-class accuracy.

Method A	**Method B**	**Wins**	**Losses**	**Ties**	**Total Differential (%)**
Our method, no line	RW	249	207	36	5.85
Our method, with line	RW	229	263	0	4.86
Our method, with line	Our method, no line	215	276	1	−0.99

Dataset 2

Qualitative results for Dataset 2 are shown in Figure 2.8. Similar to Figure 2.7, the first column shows depth images, and the second shows arm segmentations. These are followed by the results of RW and our method.

Table 2.5 shows the Jaccard index for our method and the standard RW algorithm on Dataset 2. There is a slight increase in the overall average, from 77.31% to 78.45%. The per-class results in Table 2.6 also show a slight increase in the overall average, from 90.31% to 90.60%. The pairwise comparison in Table 2.7 shows that our method outperforms RW on a majority of frames (3002 wins). While the comparison shows more losses than wins for the per-class metric, the total differential is still positive, indicating that the wins were by a greater margin than the losses.

Table 2.5: Dataset 2 [5]: Mean Jaccard index in % over all frames.

	Torso	**Head**	**LU arm**	**RU arm**	**Hip**	**LW arm**	**RW arm**	**LU leg**	**RU leg**	**LW leg**	**RW leg**	**Avg**
RW	60.19	89.58	69.50	65.21	56.82	88.71	89.95	79.36	75.56	88.68	86.87	77.31
Our method	64.74	90.98	71.80	67.32	57.06	89.81	90.54	79.28	75.09	89.15	87.20	78.45

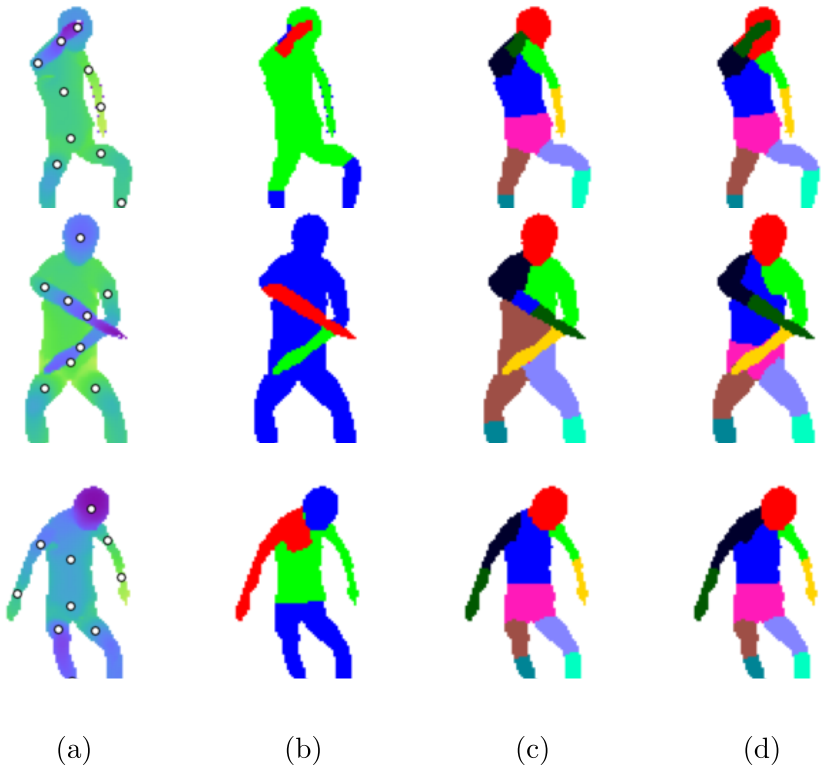


Figure 2.8: Dataset 2 [5]: Qualitative results. (a) Depth image with annotated part positions. (b) Arm segmentation. (c) Standard RW segmentation. (d) Our method.

Table 2.6: Dataset 2 [5]: Mean per-class accuracy in % over all frames.

	Torso	**Head**	**LU arm**	**RU arm**	**Hip**	**LW arm**	**RW arm**	**LU leg**	**RU leg**	**LW leg**	**RW leg**	**Avg**
RW	60.71	99.42	97.85	96.86	94.35	89.69	90.99	94.79	92.37	89.13	87.20	90.31
Our method	65.59	98.23	97.77	95.79	94.71	90.85	91.82	93.73	91.10	89.53	87.47	90.60

Analysis

As seen in the first and fourth row of Figure 2.7, the RW algorithm greatly fails when the torso position is located on a hand pixel. This problem is averted by our method, because the torso seed node is located on the lower graph layer, underneath the arm. Most of the rows demonstrate that adding a line of seeds reduces the size of the upper arm segments.

The first two rows of Figure 2.8 are examples of our method outperforming RW,

Table 2.7: Dataset 2 [5]: Pairwise comparison of methods.

Metric	**Method A**	**Method B**	**Wins**	**Losses**	**Ties**	**Total Differential (%)**
Jaccard	Our method	RW	3002	1839	83	53.49
Per-class	Our method	RW	2306	2528	90	12.63

while the third is an example of similar results. The first row performs well because the right arm is made into a graph layer, separating it from the head. In the RW figure, the right forearm is incorrectly labelled as the head. In the third row, the seed pixel for the hip is cut off from the rest of the midsection by the left arm. Our method allows the segment to spread further because the left arm is made into a graph layer.

Connecting graph layers (Section 2.3.6) allows the final segmentation algorithm to propagate a label across different layers. The first row of Figure 2.7 shows a right arm segment that is differently shaped than the final right arm parts. By connecting the graph layers, the upper arm label (red) can spread into the arm segment, reducing the size of the lower arm part (green).

Between the two versions of our method tested on Dataset 1, there is a trade-off between the two evaluation metrics. We hypothesize that adding line seed pixels greatly increases the Jaccard index because it limits the size of the upper arm segments. The left and right upper arm values increase from 65.04% and 66.62% to 74.98% and 75.84%, respectively, when the line seeds are added (Table 2.1). Since the per-class accuracy measures the ratio of pixels correctly labelled i to the total number of pixels labelled i, an overly large arm segment will still have a high accuracy. If the torso pixels are misclassified as upper arm, the per-class accuracy does not greatly reduce because the torso is so large by comparison. For this reason, we consider the Jaccard index to be a more discriminative metric and conclude that adding a line of seed pixels is overall advantageous to our method.

The results indicate that our method offers superior performance to the standard

RW algorithm. By creating a layered graph structure, there is an increase in both the Jaccard index and the per-class accuracy. Our method also outperforms two state of the art algorithms [4, 10] on the same dataset. However, it should be noted that these other algorithms addressed the more difficult problem of segmentation without annotated part positions.

2.5 Conclusions

We have presented a graph-based approach to segmenting human parts from depth images, given the image positions of each part. We propose a layered graph structure that can handle self-occlusion by the arms. The provided part positions are used to determine seed nodes and labels for a standard interactive segmentation algorithm. This work is intended to facilitate the labelling of human segmentation datasets, which can be used for training and testing new algorithms.

Future work in this area could aim to optimize the two β values for segmentation. While our method only used the default value of $\beta = 90$, two different β values can be used for segmenting the arms and final parts. Other interactive segmentation algorithms can be tested on the same layered graph structure, and the full method can be tested on other types of images, such as RGB or RGB-D. There is also room for improving the method of segmenting the arms and testing for signs of occlusion.

The main conclusion from this study is that a layered graph representation of an image can improve the performance of an interactive segmentation algorithm, when segmenting body parts with a small number of seed pixels. This has been demonstrated on two depth datasets of humans from a frontal perspective. To encourage further research on this topic, our implementation will be made publicly available at `https://github.com/ajhynes7`.

References

[1] J. Charles and M. Everingham, “Learning shape models for monocular human pose estimation from the microsoft xbox kinect,” in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on.* IEEE, 2011, pp. 1202–1208.

[2] S. Chandra, S. Tsogkas, and I. Kokkinos, “Accurate human-limb segmentation in rgb-d images for intelligent mobility assistance robots,” *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pp. 436–442, 2015.

[3] M. Roccetti, G. Marfia, and A. Semeraro, “Playing into the wild: A gesture-based interface for gaming in public spaces,” *Journal of Visual Communication and Image Representation*, vol. 23, no. 3, pp. 426–440, 2012.

[4] A. Hernández-Vela, N. Zlateva, A. Marinov, M. Reyes, P. Radeva, D. Dimov, and S. Escalera, “Graph cuts optimization for multi-limb human segmentation in depth maps,” in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on.* IEEE, 2012, pp. 726–732.

[5] K. Nishi and J. Miura, “Generation of human depth images with body part labels for complex human pose recognition,” *Pattern Recognition*, vol. 71, pp. 402–413, 2017.

[6] D. Sánchez, J. C. Ortega, M. Á. Bautista, and S. Escalera, “Human body segmentation with multi-limb error-correcting output codes detection and graph cuts optimization,” in *Iberian Conference on Pattern Recognition and Image Analysis.* Springer, 2013, pp. 50–58.

[7] F. Xia, P. Wang, L.-C. Chen, and A. L. Yuille, "Zoom better to see clearer: Human part segmentation with auto zoom net," *CoRR*, vol. abs/1511.06881, 2015.

[8] A. Hernández-Vela, M. Reyes, V. Ponce-López, and S. Escalera, "Grabcut-based human segmentation in video sequences," in *Sensors*, 2012.

[9] M. Madadi, S. Escalera, J. Gonzalez, F. X. Roca, and F. Lumbreras, "Multi-part body segmentation based on depth maps for soft biometry analysis," *Pattern Recognition Letters*, vol. 56, pp. 14–21, 2015.

[10] H. Liang, J. Yuan, and D. Thalmann, "Parsing the hand in depth images," *IEEE Transactions on Multimedia*, vol. 16, no. 5, pp. 1241–1253, 2014.

[11] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. Ieee, 2011, pp. 1297–1304.

[12] A. Hernández-Vela, C. Primo, and S. Escalera, "Automatic user interaction correction via multi-label graph cuts," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1276–1281.

[13] Y. Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in nd images," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 1. IEEE, 2001, pp. 105–112.

[14] L. Grady, “Random walks for image segmentation,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 11, pp. 1768–1783, 2006.

[15] V. Vezhnevets and V. Konouchine, “Growcut: Interactive multi-label nd image segmentation by cellular automata,” in *proc. of Graphicon*, vol. 1, 2005, pp. 150–156.

[16] D. Comaniciu and P. Meer, “Mean shift: A robust approach toward feature space analysis,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[17] R. C. Gonzalez and R. E. Woods, “Digital image processing,” 2012.

[18] J. E. Bresenham, “Algorithm for computer control of a digital plotter,” *IBM Systems journal*, vol. 4, no. 1, pp. 25–30, 1965.

[19] L. Grady, “Software,” available online: http://leogrady.net/software/ (accessed on 14 December 2017).

[20] G. Csurka, D. Larlus, F. Perronnin, and F. Meylan, “What is a good evaluation measure for semantic segmentation?.” in *BMVC*, vol. 27. Citeseer, 2013, p. 2013.

[21] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

Chapter 3

Combinatorial Optimization for Human Body Tracking

Andrew Hynes and Stephen Czarnuch

Abstract

We present a method of improving the accuracy of a 3D human motion tracker. Beginning with confidence-weighted estimates for the positions of body parts, we solve the shortest path problem to identify combinations of positions that fit the rigid lengths of the body. We choose from multiple sets of these combinations by predicting current positions with kinematics. We also refine this choice by using the geometry of the optional positions. Our method was tested on a dataset from an existing motion tracking system, resulting in an overall increase in the sensitivity and precision of tracking. Notably, the average sensitivity of the feet rose from 52.6% to 84.8%. When implemented on a 2.9 GHz processor, the system required an average of 3.5 milliseconds per video frame.

3.1 Introduction

Previous work on computer vision based 3D human motion capture commonly uses probabilistic methods [1–7] that select body part locations from multiple hypotheses. These methods predict locations of tracking points in three-dimensional space, with associated confidence estimates for each prediction. The direct outputs of these motion capture methods are subject to noise and uncertainty as a result of the probabilistic nature of the hypothesis estimation and the underlying stochasticity of the methodologies. Methods of raw motion capture data processing generally focus on two objectives: 1) pose estimation and task identification [8]; and 2) noise reduction and prediction improvement [9–13]. Our objective is to develop a novel method of noise reduction and prediction improvement for human tracking.

Methods of noise reduction and prediction improvement generally aim to improve the underlying raw motion tracking data independent of any tasks or activities beyond fundamental human motion. Methods such as Kalman filters and wavelet transforms do not require training data, and incorporate temporal and kinematic information on each tracked point [9]. However, these methods generally consider each point independently and do not take into account the physical and kinematic relationship between the tracked human joints [10]. Training data have also been used to improve tracking data with particular focus on restoring lost tracking points caused by occlusion or missing markers (in the case of marker-based systems) [11–13]. These approaches develop motion dictionaries to remove noise and fill in incomplete data, but again do not utilize the physical and kinematic relationship between points [10]. Recently, these data-driven approaches have begun to incorporate human kinematic information, but this is still an active and new area of research [10].

We build upon a stochastic, probabilistic body part predictor originally developed to track the upper body and hands of persons with dementia [1], and recently extended to track the full-body motion of persons with multiple sclerosis while they walk [14]. The computer vision based predictor provides multiple confidence-weighted estimates for the 3D location of 11 body parts, for each new frame of video data. The tracked body parts are: head, hips, thighs, knees, calves and feet. Data for the system are captured from a single view with a depth sensor. Each possible position for each tracked body part is associated with a confidence value. The position with the highest confidence is not always the correct position of the body part. We hypothesize that the correct part location is usually available in the set of possible locations. If the correct position is not available, restoration of the lost point is required.

We show that the true part positions can be more accurately selected by combinatorial and kinematic techniques, rather than choosing the positions with the highest confidence. We treat stochastic motion capture estimates as vertices in a graph, and we solve the shortest path problem to identify valid combinations of these estimates. Finding the shortest path has been previously used for body tracking in 2D images [15] to distinguish body parts from the background. We rely on raw tracking data, in this case from our stochastic motion capture predictor [14], to identify multiple body part position estimates, and propose a new implementation of the shortest path algorithm to optimally select from these estimates.

3.2 Methodology

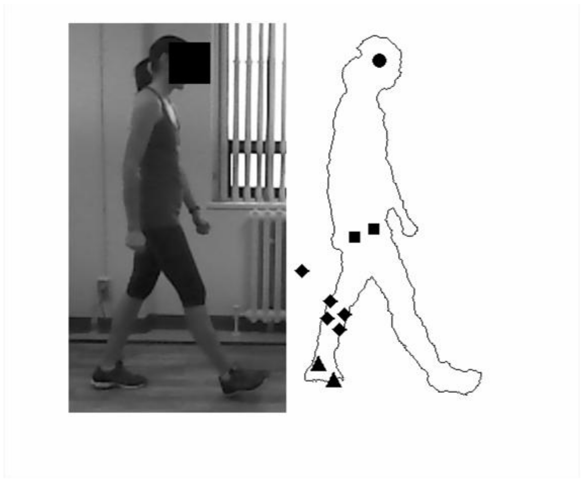
We used a single depth camera to capture 640×480 images at ~30 frames per second of a person walking across the view of the camera four times (twice to the right, twice to the left). Each frame of captured data is processed by the body part predictor [14], which was trained to provide multiple confidence-weighted estimates of the locations of the 11 body parts during walking. We propose representing these estimates as two graphs, one for each side of the body. We then solve the shortest path problem for each graph to select body positions that closely match the expected lengths of the body (the estimation of these lengths is explained in section 3.2.2). The result is a set of shortest paths through each graph. The best path is chosen from these shortest paths using kinematic data from previous frames. Under certain conditions, we revise these selections by choosing paths that maximize the area spanned by the path positions.

3.2.1 Graph representation of the Human Body

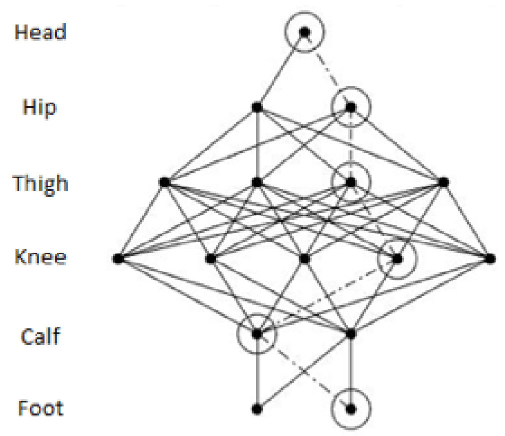
In discrete mathematics, a graph is a collection of vertices, with edges that connect them. The graph is weighted when the edges have an associated value or cost meaningful to the application. The total weight of a path between two vertices is then the sum of all edge weights along this path.

Due to the high reliability of the original predictions for the head [1], we always select the head position with the highest confidence for both left and right graphs. The confidence values of the other part positions are disregarded. We index the set of body parts (head, hip, thigh, knee, calf, foot) as i, with each part having a set of vertices j representing the estimated 3D positions for that body part. We denote an estimated position as $part_{i,j}, i \in \{1, ..., 6\}, j \in \{0, 1, 2, ...\}$. We define a weighted edge as a connection between adjacent body parts. The vertices of $part_i$ form a complete bipartite graph with the vertices of $part_{i+1}$, i.e., every vertex in a row is connected to every vertex in adjacent rows (Fig. 3.1).

The shortest path problem seeks to minimize the total path weight between two vertices. In our case, the path begins at the head and ends at the foot. Learned estimates are computed for the expected length from $part_i$ to $part_{i+1}$ (outlined in section 3.2.2). We define the edge weight w for a pair of connected vertices as the square of the difference between the expected length, $\hat{l}$, and the actual 3D length associated with the pair of vertices, l.



(a)



(b)

Figure 3.1: Graph representation of part predictions. (a) Sample image frame and subsequent body part predictions. Predictions are shown for the head and left hip, knee, and foot (b) Corresponding graph, comprised of the body parts i (rows) and possible 3D positions j of each part (vertices in a row). The shortest path from the head to the second foot vertex is shown as the circled vertices and the dashed line.

$$w = (l - \hat{l})^2 \tag{3.1}$$

Using this definition of edge weight, a shortest path is a combination of position estimates that closely fits the expected lengths of the body.

3.2.2 Shortest Path Algorithm

We aim to move along each graph from head to foot, forming a shortest path to each foot vertex. One vertex is chosen for each body part, when all six parts are present in the frame. Since each $part_i$ is only connected to $part_{i+1}$, with no cycles, our graph representation is a directed acyclic graph, which is topologically sorted. This allows for the implementation of a shortest path algorithm that runs in linear time [16] (the pseudocode is shown in Algorithm 2 of Chapter 5).

The total distance of a path is the sum of the weights along the path. The distance to the head vertex is set to zero, since each path begins at the head. The total distance to every other vertex is initially assumed to be infinite. The shortest path algorithm for a topologically sorted, directed acyclic graph can now be executed. For each part beginning with the head, the distance is calculated between each vertex $part_{i,j}$ and adjoining vertices $part_{i+1,k}$. If the current distance to $part_{i+1,k}$ is greater than the distance to $part_{i,j}$ plus the weight w of the edge from $part_{i,j}$ to $part_{i+1,k}$, the distance to $part_{i+1,k}$ is revised to the lower value, and the vertex $part_{i,j}$ is recorded as being along the shortest path to vertex $part_{i+1,k}$.

On each frame, when the shortest path algorithm is completed for each side, every foot vertex has a unique shortest path leading to it. Each path can have a different total weight, depending on which edges constitute the path.

Learning Body Lengths

In an uninitialized state, we first assume that the length from every $part_i$ to $part_{i+1}$ is zero. We execute the shortest path algorithm for both graphs in a video frame, finding vertices with the lowest overall path distance. From these selections, we record the lengths between $part_i$ and $part_{i+1}$. This process is repeated over multiple frames of data, resulting in a population of lengths for each pair of adjacent body parts. For a given pair, any lengths outside of the median $\pm$ median absolute deviation are removed. The median of the remaining data replaces the initial estimate for the true length of the pair. The body length learning process is repeated over this same set of frames until each pair length converges to a stable value within a tolerance of ± 0.1 cm. These become the expected lengths of the person being tracked, which are used in Equation 3.1. The process of learning body lengths can be computed prior to, or concurrently with the selection of position estimates.

3.2.3 Shortest Path Selection

The result of the shortest path algorithm for each graph is a shortest path to each foot vertex. Out of the multiple shortest paths for each foot, the path with the smallest total weight (the minimum shortest path) may not necessarily be the best choice for tracking the body, because some position estimates for the right side of the body are actually for the left side, and vice versa. The shortest path algorithm alone cannot distinguish between left and right sides of the body. To counter this, we employ additional methods to select the optimal shortest path.

Path Selection with Kinematics

We compare the positions of each vertex associated with these paths to positions predicted by kinematics. At frame f, we use the positions from the previous three frames to obtain the velocity and acceleration at frame $f-1$. This is repeated for each $part_i$. The velocity and acceleration at frame $f-1$ are used to predict the part position at f.

We now choose the shortest path that most closely fits these kinematic predictions. We compute the square of the Euclidean distance between a position on a path and its corresponding kinematic prediction. We then define the total kinematic cost C_{kin} of a path as the sum of these squares.

$$C_{kin} = \sum_{i=1}^{6} \left\| \mathbf{p}_{kin,\ i} - \mathbf{p}_{path,\ i} \right\|^2 \tag{3.2}$$

In Equation 3.2, $\mathbf{p}_{kin,\ i}$ is the position of $part_i$ predicted by kinematics, and $\mathbf{p}_{path,\ i}$ is the position of $part_i$ from the path being considered. Out of the shortest paths available for each side of the body, we first exclude paths whose total weights are too large to be valid. For each side of each new frame, we compute the absolute relative difference Δ_{path} between the total weight of a given shortest path W_{path} and the total weight of the minimum shortest path W_{min}.

$$\Delta_{path} = \frac{|W_{path} - W_{min}|}{W_{min}} \tag{3.3}$$

The maximum allowed relative difference is a learned value of 600%. This was optimized to our ground truth data. Any path with a higher relative difference is ignored. After excluding invalid paths, the shortest path with the lowest C_{kin} is selected.

Path Selection Refinement

In some cases, when the legs cross one another, the combined shortest path algorithm and kinematic path selection still fails to select the optimal body part positions. Accordingly, a path selection refinement is executed when the two selected feet are within 10 cm, considering all possible combinations of the left and right side shortest paths. For a given pair of left and right paths, there is a triangle formed by the left foot, right foot, and head. We select the pair of shortest paths that maximizes the area of this triangle. Similar to the kinematic path selection process, we exclude paths which have a total weight that differs too greatly from that of the minimum shortest path. In this case, the maximum allowed relative difference is a learned value of 60%, for both sides of the body.

3.2.4 Classification Accuracy

We use a confusion matrix to assess the performance of our body tracking. For a given part, we consider the Euclidean distance from the chosen position estimate to the corresponding ground truth position. If the truth position is closer to the given part estimate than to all the other part estimates, then the part is correctly classified. If the chosen position estimate of a different body part is closer to the truth, the part has been misclassified. From the confusion matrix, we calculate the sensitivity and precision for each body part.

3.3 Results

A total of 332 frames of motion data were captured. Of these, there were 282 frames with at least one shortest path computed, as the shortest path algorithm is only applied when all six body parts are present in the graph. The system was implemented in MATLAB on a 2.9 GHz Intel Core i7 processor. The combined shortest path algorithm, kinematic selection and selection refinement required an average time of 3.5 milliseconds per frame.

3.3.1 Graph Representation of the Human Body

Table 3.1 displays statistics on the graphs constructed over the full image set. The mean number of vertices is equivalent to the mean number of optional estimates given for that part position. These values were calculated after excluding frames with no options present.

3.3.2 Shortest Path Algorithm

There are 282 frames with at least one completed path and 269 frames with at least one completed path for both sides of the body. The remaining frames were missing a vertex for at least one body part. A total of 1259 shortest paths are calculated for

Table 3.1: Mean and maximum number of vertices from the full set of video frames.

Body part	Mean		Maximum	
	Left	Right	Left	Right
Head	1.27	1.27	3	3
Hip	2.05	2.26	9	8
Thigh	1.87	1.77	7	10
Knee	2.37	3.95	10	14
Calf	2.62	2.51	10	8
Foot	2.18	2.03	7	7

this dataset, giving an average of 4.46 paths per frame. The total weight of each path ranges from $1.73 \cdot 10^0$ to $3.37 \cdot 10^4$ cm^2 with an average path weight of 656.3 cm^2.

Learning Body Lengths

We show the convergence of the learned body lengths to the ground truth measures by calculating the absolute relative errors that result by using the first 30, 60, 90, and 120 frames, and using all frames. The averages of these errors over all body parts are 11%, 11%, 8%, 7%, and 6%, respectively.

3.3.3 Shortest Path Selection

Path Selection with Kinematics

This path selection process was used on each frame, for each side of the body. A shortest path other than the minimum was chosen on 89 frames for the left side, and 86 frames for the right.

Path Selection Refinement

The path refinement process occurred on 107 frames. On 49 of the 107 frames, the paths selected by this refinement further reduced the difference between the chosen position estimates and the ground truth positions. The average absolute error from the refinement process over the entire dataset is shown in Table 3.2 and compared to the error of the shortest path algorithm alone and to the kinematic selection. The absolute error of a body part on a video frame is the Euclidean distance between the position estimate and its corresponding ground truth position. A visual example of the refinement process is shown in Fig. 3.2.

Table 3.2: Average absolute error (cm) over the full dataset. This is defined as the distance between the chosen position estimate and the corresponding ground truth position.

Body part	Shortest path only	Kinematic selection	Selection refinement
Head	3.28	3.28	3.28
Hip	8.43	8.45	8.46
Thigh	7.37	7.27	7.32
Knee	7.58	7.44	7.15
Calf	9.1	9.42	6.85
Foot	11.24	10.72	6.76
Overall average	**7.83**	**7.76**	**6.64**

3.3.4 Classification Accuracy

The average sensitivity percentages are shown in Table 3.3. The precision percentages were similar to the sensitivity, with overall averages of 73.6, 78.2, 78.6, and 83.2 for the original estimates, shortest path only, kinematic selection, and selection refinement, respectively.

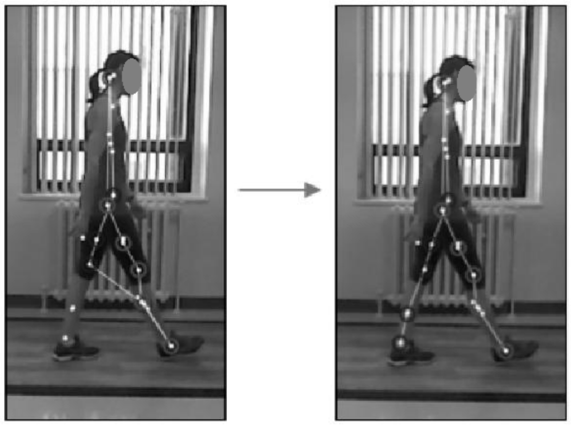


Figure 3.2: Refining the path selection by maximizing area. The white dots show all position estimates. The white lines represent the chosen positions.

Table 3.3: Average sensitivity percentages over the full dataset. The original estimates were determined by selecting the positions with the highest confidence.

Body part	Original estimates	Shortest path only	Kinematic selection	Selection refinement
Head	99.4	99.6	99.6	98.3
Hip	73.7	75.9	75.5	72.6
Thigh	74.9	76.5	77.0	76.7
Knee	83.5	79.0	80.2	84.2
Calf	55.1	69.7	68.5	82.5
Foot	52.6	68.6	70.6	84.8
Overall average	**73.2**	**78.2**	**78.6**	**83.2**

3.4 Discussion

We have proposed a method of optimally selecting positions from the set of estimates generated by a human motion tracking system. Two or three position estimates are given per body part, on average, with as many as 12-14 in some cases. The estimate with the highest confidence is often not the best estimate, allowing our method to improve the final prediction. This is evinced by the improvement in the overall sensitivity and precision of our motion tracker, which increased from 73.2% to 83.2%

and from 73.6% to 83.2%, respectively.

The use of the shortest path algorithm alone resulted in marked improvement in part locations. However, the shortest path algorithm cannot directly distinguish between the left and right legs since its function is to optimize predictions rather than to make its own. In many (if not all) cases, the predictions for the left body parts often include the true position of the right parts, and vice versa. This can result in the shortest path algorithm selecting a highly similar set of points for both legs. For example, a data frame may correctly identify the legs far apart, as the person is in mid-stride. In the following frame, the shortest path algorithm may show one leg abruptly shifted to match the other, when the legs are still actually apart. The kinematic selection process finds the shortest path which most likely follows from the previous frame, given the velocity and acceleration of each part at that frame. On its own, this method reduces the risk of a leg moving too rapidly. However, this process often results in the legs incorrectly adhering together after they first cross, giving the impression that the person is not walking, but gliding along with both legs pointed forwards. The path selection refinement successfully ameliorated this issue in all frames where this condition was present (as shown in the example in Fig. 3.2). Critical to this methodology is the automated determination of expected body lengths on a per-user basis. Our results show that the lengths of the body rapidly converge to the ground truth values with just a small number of data frames. The error between the actual and estimated lengths decreases with an increase in the number of data frames used, at a small cost of processing time. This suggests that the system could periodically evaluate the learned lengths with new data.

There are two main limitations for this approach to body tracking optimization. First, being a proof-of-concept study, our sample size is relatively small. Future work will look to include more motion images from a wider, more diverse range of participants and motions. The second is that our dataset is currently restricted to walking data. An implicit assumption for this application of the shortest path algorithm is that the head to hip is a rigid link with a constant length. This assumption is useful for an upright walking pose, but it becomes problematic when the person is bending their spine, changing the link into a curve. Our data suggests that the shortest path problem can still be implemented in this case, but more body points would be needed to maintain accuracy, like the chest, stomach, and shoulders. Our part predictor currently estimates these and other upper body parts. Future work will look to implement the proposed algorithm on the upper body as well as the lower body and head, providing an optimization of the full body tracking data.

3.5 Conclusion

We have presented results suggesting that our method of body tracking optimization provides an improvement for a stochastic motion tracker. We first represent the set of left and right body part predictions as two directed acyclic graphs. We then optimize part selections via error calculations for learned and expected body lengths, and revise these selections using kinematics and the geometry of the body. Our results have demonstrated that this method chooses accurate body part positions from available options on a small dataset capturing real human motion.

References

[1] S. Czarnuch and A. Mihailidis, "Development and evaluation of a hand tracker using depth images captured from an overhead perspective," *Disability and Rehabilitation: Assistive Technology*, vol. 11, no. 2, pp. 150–157, 2016.

[2] B. J. Southwell and G. Fang, "Human Object Recognition Using Colour and Depth Information from an RGB-D Kinect Sensor," *International Journal of Advanced Robotic Systems*, vol. 10, 2013.

[3] I. Oikonomidis, N. Kyriazis, and A. Argyros, "Efficient Model-based 3D Tracking of Hand Articulations using Kinect," *22nd British Machine Vision Conference*, pp. 1–11, 2011.

[4] A. Hernandez-Vela, N. Zlateva, A. Marinov, M. Reyes, P. Radeva, D. Dimov, and S. Escalera, "Graph cuts optimization for multi-limb human segmentation in depth maps," in *2012 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE, 2012, pp. 726–732.

[5] B. Holt and R. Bowden, "Static Pose Estimation from Depth Images using Random Regression Forests and Hough Voting," in *Proceedings of the 7th International Conference on Computer Vision Theory and Applications*, 2012, pp. 557–564. [Online]. Available: http://eprints.pascal-network.org/archive/00009014/

[6] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth

images," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on.* Ieee, 2011, pp. 1297–1304.

[7] C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in *IEEE International Conference on Computer Vision Workshops*, 2011, pp. 1228–1234. [Online]. Available: http://link.springer.com/chapter/10.1007/978-1-4471-4640-7{_}7

[8] D. Demirdjian, T. Ko, and T. Darrell, "Constraining human body tracking," in *Proceedings Ninth IEEE International Conference on Computer Vision.* IEEE, 2003, pp. 1071–1078 vol.2.

[9] K. Yamane and Y. Nakamura, "Dynamics filter - Concept and implementation of online motion generator for human figures," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 3, pp. 421–432, 2003.

[10] Z. Wang, Y. Feng, S. Liu, J. Xiao, X. Yang, and J. J. Zhang, "A 3D human motion refinement method based on sparse motion bases selection," in *Proceedings of the 29th International Conference on Computer Animation and Social Agents.* New York, New York, USA: ACM Press, 2016, pp. 53–60.

[11] H. Lou and J. Chai, "Example-based human motion denoising," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 5, pp. 870–879, 2010.

[12] D. Holden, J. Saito, T. Komura, and T. Joyce, "Learning motion manifolds with convolutional autoencoders," in *SIGGRAPH Asia Technical Briefs.* New York, New York, USA: ACM Press, 2015, pp. 1–4.

[13] Y. Feng, M. Ji, J. Xiao, X. Yang, J. J. Zhang, Y. Zhuang, and X. Li, "Mining Spatial-Temporal Patterns and Structural Sparsity for Human Motion Data Denoising," *IEEE Transactions on Cybernetics*, vol. 45, no. 12, pp. 2693–2706, 2015.

[14] S. M. Czarnuch and M. Ploughman, "Toward inexpensive, autonomous, and unobtrusive exercise therapy support for persons with MS," in *ACTRIMS*, New Orleans, 2016.

[15] X. Ren, A. Berg, and J. Malik, "Recovering human body configurations using pairwise constraints between parts," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 1. IEEE, 2005, pp. 824–831 Vol. 1.

[16] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms: Second Edition*. Cambridge: The MIT Press, 2001.

Chapter 4

Comparing the Gait Analysis of a Kinect System to the Zeno Walkway: Preliminary Results

Andrew Hynes, Megan C. Kirkland, Michelle Ploughman, and Stephen Czarnuch

Abstract

Pressure sensitive walkways are a commonly used measuring device for gait analysis. However, they can be prohibitively expensive for out-of-clinic measurements. An alternative approach to gait analysis is the use of a depth sensing camera (e.g., the Kinect). Our approach is to collect lower-body gait data using a single, inexpensive Kinect camera, with a line of sight perpendicular to the walking path. Participants with MS performed walking passes on a pressure sensitive walkway and in front of the camera. The following gait metrics were measured with both systems: step length, stride length, stride width, and stride velocity. We present the preliminary results of comparing gait metrics, showing Spearman correlations ranging from 0.857 to 0.976. These preliminary results suggest that inexpensive gait tracking may be a practical reality in non-clinical settings.

4.1 Introduction

Gait analysis is a common clinical practice for tracking disease progression and facilitating rehabilitation, for a variety of neurological diseases including Parkinson's [1], stroke [2], and multiple sclerosis [3–9]. The analysis is often performed by clinicians using observational tests, such as the Timed Up and Go [10], or with the aid of pressure sensitive walkways [11]. Because these tests need a certified clinician, they are less accessible to rural areas. Alternatively, gait analysis has been performed by computer vision systems, such as depth sensing cameras [7, 12, 13].

Pressure sensitive walkways measure important gait characteristics that a computer vision system is unable to directly evaluate, such as the force of the foot on the ground. However, both systems can measure spatiotemporal gait characteristics such as step length and stride velocity. A computer vision system can also supplement the measurements of the pressure walkway by tracking upper body parts and joint angles.

The purpose of this study is to compare our developed depth sensor tracking system to a validated pressure sensitive walkway, the Zeno Walkway, in conjunction with the ProtoKinetics Movement Analysis Software (PKMAS) [14, 15].

Participants with MS completed four walking passes on the walkway, while being simultaneously recorded by a Kinect camera from a side view. The native Kinect software development kit (SDK) is intended to track from a frontal view, as is common for its original purpose of gaming. Instead of using the SDK, we build upon our previous work [16–18], which developed an algorithm for tracking bodies from a side view. This allows for our method to be implemented on a generic depth sensing camera.

4.2 Related Work

Performing gait analysis with the Kinect camera is an active area of research [3–7, 9, 13, 19–28]. The Kinect has been used to analyze gait for a number of neurological disorders [29, 30], including multiple sclerosis [3, 5–7, 9]. A common technique is to first track the human skeleton, either using the native software development kit (SDK) [7, 24], or with novel algorithms [28]. However, gait analysis has been accomplished without skeleton tracking, by analyzing the motion of the body centre of mass [20]. The tracking abilities of the Kinect from a non-frontal view have also been examined, for general tracking [31], and for gait analysis [20, 23, 28, 31].

Gabel et al. [24] measured both stride metrics and arm kinematics. A model for walking was built using information from wearable sensors. The Kinect SDK was used to track a virtual skeleton, which was passed into this learned model. Gholami et al. [7] used the concept of dynamic time warping to develop novel gait metrics. Their study compared the gait of participants with MS to a healthy control group, and they developed a distance metric to compare dysfunctional gait to healthy gait.

Several studies have compared the gait analysis of Kinect to previously validated systems, including marker-based motion tracking [21, 26, 32, 33] and the GAITRite pressure mat [13, 20, 27]. Cippitelli et al. [23] tracked body joints from a side view, using a purpose-built algorithm. They obtained an objective score for the Get Up and Go test, and compared results to a marker-based system. Motiian et al. [27] focused on gait analysis for children, and compared results to the GAITRite pressure mat. The Kinect SDK was used to track the skeleton from a frontal view, accompanied by a side view Kinect for data visualization during the annotation phase. Dolatabadi et al. [13] tracked the walks of healthy participants with both a GAITRite mat and a frontal view Kinect using the SDK. They found strong agreement between the two systems for a number of spatiotemporal gait parameters.

To our knowledge, gait analysis with the Kinect has not yet been compared to a Zeno Walkway with the PKMAS software. However, these two systems have been used in conjunction to provide a non-immersive virtual reality for treadmill training [34]. The comparison with Kinect is valuable as it can provide an open-source alternative to the proprietary PKMAS software.

4.3 Methodology

Eight walking trials were completed by two participants with MS. Each trial consisted of four passes in front of the camera, two to the left and two to the right. Data collection occurred at the Recovery and Performance Laboratory, a part of the Faculty of Medicine at Memorial University. The Kinect tracked 11 separate body parts: the head, hips, thighs, knees, calves and feet.

Four gait metrics were measured by both the Zeno Walkway and the Kinect: step length, stride length, stride velocity, and stride width. For our Kinect system, only the head and foot positions are needed to calculate these gait metrics. However, tracking the full lower body is instrumental in correctly estimating the foot positions [17, 18].

4.3.1 Stride Detection

During the swing phase of a normal stride, one foot remains planted on the ground, while the other moves forward. These are the stance foot and swing foot, respectively.

Using the tracked body part positions, the distance between the two feet is recorded for each frame. An example of the foot distance data can be seen in Fig. 4.1. There are four main sections of data, showing the different passes in front of the camera. The peaks in the data indicate instances when the feet are furthest apart in a stride.

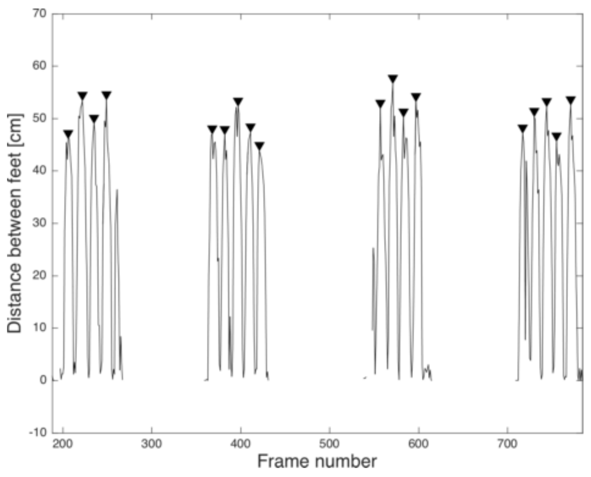


Figure 4.1: Foot to foot distance for each image frame in a walking trial, with detected peaks marked.

A stride is detected with the following steps:

1. Use a peak detection algorithm to locate the peaks in the foot distance data. The MATLAB `findpeaks` function [35] was used for this implementation. The minimum peak prominence was specified as 75% of the maximum foot distance, to avoid detecting false peaks.
2. Record the frame numbers of each detected peak.
3. Cluster the peak frame numbers, so that peaks are grouped by walking pass.
4. Examine each pair of consecutive peaks that both occur in the same pass. This represents a full walking stride. The pair of frame numbers F_i and F_f are later used to calculate stride velocity.

When a stride is detected, the two peak frames are analyzed to obtain gait metrics. The distance travelled by the left foot between the two frames is calculated, as well as for the right. Ideally, one foot will move a relatively long distance while the other remains in its place. The foot which travelled a greater distance is labelled as the swing foot, and the other as the stance foot.

4.3.2 Gait Metrics

Before the gait metrics are calculated, all peak foot positions are projected onto the same plane. The plane passes through the point $\begin{bmatrix} 0 & y_{min} & 0 \end{bmatrix}^T$, where y_{min} is the lowest y coordinate of the peak foot positions in a trial. The normal vector of the plane is $\begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$. This plane is intended to model the surface of the Zeno Walkway.

The gait metric calculations were designed to closely match the calculations by PKMAS, as described in [8]. A diagram of a full stride is shown in Fig. 4.2. The swing foot moves from its initial position $\mathbf{p}_{swing,\ i}$ to its final position $\mathbf{p}_{swing,\ f}$. The swing path $\mathbf{s}$ is defined as the displacement vector between these points.

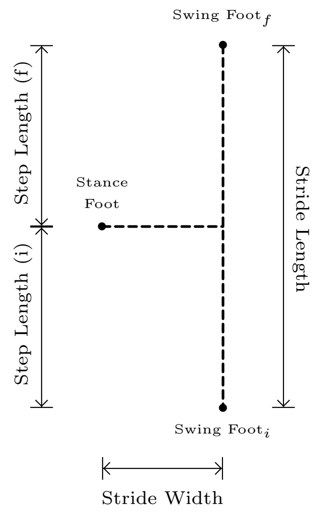


Figure 4.2: Diagram of step length, stride length, and stride width. During a stride, the stance foot stays stationary while the swing foot moves forward. The labels assume that the right foot is swinging.

$$\mathbf{s} = \mathbf{p}_{swing,\ f} - \mathbf{p}_{swing,\ i} \tag{4.1}$$

The stride length is the distance between the initial and final swing foot positions.

$$l_{stride} = ||\mathbf{s}|| \tag{4.2}$$

The stance foot position, $\mathbf{p}_{stance}$, is projected onto the line between the two swing foot positions.

$$\mathbf{p}_{proj} = \mathrm{proj}_{\mathbf{s}}\, \mathbf{p}_{stance} \tag{4.3}$$

This projected point is used to calculate step length and stride width. A full stride consists of two step lengths. The first step length is the distance from $\mathbf{p}_{swing,\ i}$ to $\mathbf{p}_{proj}$, and the second from $\mathbf{p}_{proj}$ to $\mathbf{p}_{swing,\ f}$.

$$\begin{aligned} l_{step,\ i} &= ||\mathbf{p}_{swing,\ i} - \mathbf{p}_{proj}|| \\ l_{step,\ f} &= ||\mathbf{p}_{swing,\ f} - \mathbf{p}_{proj}|| \end{aligned} \tag{4.4}$$

The stride width is the distance from the stance foot to its projection along the swing path.

$$w_{stride} = ||\mathbf{p}_{stance} - \mathbf{p}_{proj}|| \tag{4.5}$$

Finally, the stride velocity is calculated using the positions of the head. $\mathbf{p}_{head,\ i}$ and $\mathbf{p}_{head,\ f}$ are the head positions at frames F_i and F_f, respectively. Since the frame rate of the Kinect camera is 30 frames per second, the difference of frame numbers is divided by 30 to obtain a stride time in seconds. Thus, the stride velocity is

$$v_{stride} = \frac{d_{head}}{(F_f - F_i)/30} \tag{4.6}$$

where d_{head} is the distance from $\mathbf{p}_{head,\ i}$ to $\mathbf{p}_{head,\ f}$.

After gait metrics have been calculated for every detected stride in a trial, outliers are removed from the dataset of each gait metric. Outliers are defined as values outside of the median $\pm\, 2 \cdot \text{MAD}$, where MAD is the median absolute deviation [36].

4.4 Results

4.4.1 Relative Error

The mean of the gait metric measurements was calculated for each walking trial. Table 4.1 shows data from both the Kinect and Zeno systems, as well as the relative error. The Kinect measurements for step length and stride length were consistently under the Zeno measurements, resulting in negative relative errors. In general, the stride velocity has the lowest relative error magnitudes, ranging from 0 %–6 %. There is a mixture of negative and positive errors. The stride width has the highest overall relative errors, ranging from 2 %–47 %. For this metric, the Kinect measurements are consistently above the Zeno measurements.

Table 4.1: Mean gait metrics and relative error for each trial.

	Step Length [cm]			Stride Length [cm]			Stride Velocity [cm/s]			Stride Width [cm]		
Trial	Kinect	Zeno	Rel. error	Kinect	Zeno	Rel. error	Kinect	Zeno	Rel. error	Kinect	Zeno	Rel. error
1	49.8	55.6	-11%	99.9	112.6	-11%	121.1	120.6	0%	13.6	13.4	2%
2	47.9	53.2	-10%	96.8	108.0	-10%	102.0	103.4	-1%	13.1	11.4	15%
3	44.5	48.9	-9%	89.4	96.9	-8%	94.9	90.9	4%	14.6	11.2	30%
4	48.6	54.2	-10%	97.4	110.1	-12%	107.2	110.8	-3%	12.6	11.1	13%
5	45.3	50.2	-10%	90.5	100.6	-10%	98.1	98.9	-1%	12.1	9.9	23%
6	42.0	46.9	-10%	83.5	94.3	-12%	88.7	94.6	-6%	12.4	9.7	28%
7	56.0	64.1	-13%	114.3	129.0	-11%	124.3	123.8	0%	12.3	8.4	47%
8	57.3	63.6	-10%	115.5	128.0	-10%	116.8	121.8	-4%	11.0	8.1	37%

4.4.2 Correlation

The Spearman correlation coefficient was used to measure the correlation of the two systems. The coefficient, also referred to as Spearman's rho, has been previously used for assessing gait analysis with Kinect [5, 30]. It does not require that the variables are normally distributed, and it is more robust to outliers than the Pearson coefficient [37].

Table 4.2 shows the Spearman coefficient for each gait metric. The Kinect measurements of step length, stride length and stride velocity are all strongly correlated with the Zeno measurements, having coefficients > 0.95.

Table 4.2: Spearman correlation between the two systems.

Gait metric	ρ
Step Length	0.9762
Stride Length	0.9762
Stride Velocity	0.9524
Stride Width	0.8571

4.4.3 Agreement

Bland-Altman analysis [38] is a common method in medical statistics for assessing the agreement between two systems of measurement. It has been used for concurrent validity studies with the Kinect [13, 25, 33].

In a Bland-Altman plot, the difference between two measurements is plotted against the mean of the two measurements. Bland and Altman recommended that 95% of the data should lie within the lower and upper limits of agreement, which are defined as ± 1.96 standard deviations from the mean difference. The differences can also be displayed as percentages of the mean values, so that they are proportional to the magnitude of the data [39]. This is useful for comparing limits of agreement between metrics with different magnitudes, such as stride velocity and stride width. Fig. 4.3 shows the Bland-Altman plot for stride velocity, with differences expressed as percentages.

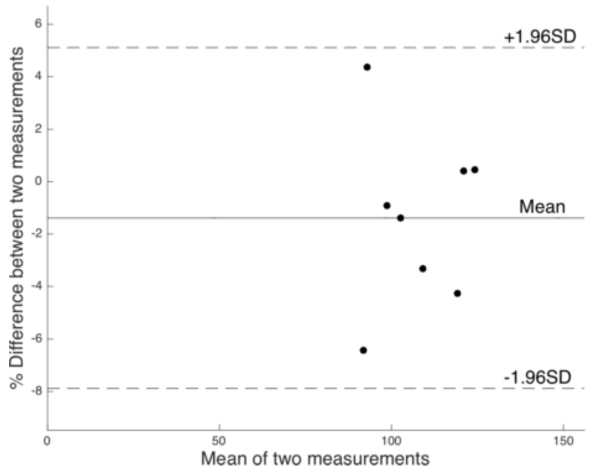


Figure 4.3: Bland-Altman plot for stride velocity.

Table 4.3 shows the results of Bland-Altman analysis for each gait metric. The bias is the mean of differences between measurements. This bias is visible in Fig. 4.4. The Kinect measurements of step length have a clear negative bias, while the stride velocity is essentially unbiased.

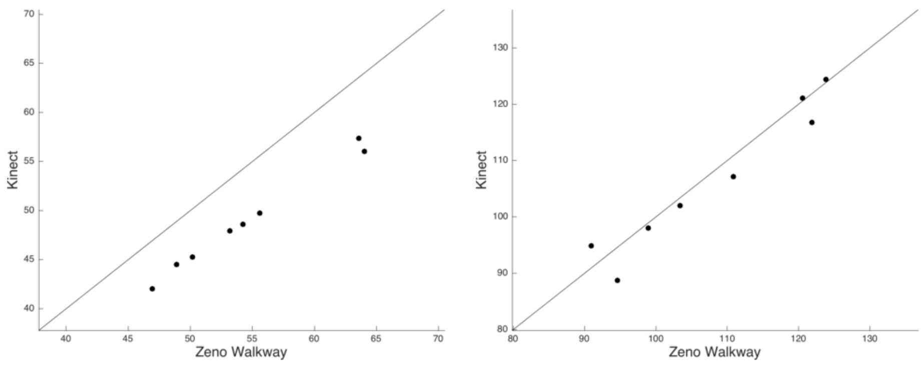
Although the stride velocity has the lowest absolute bias, the step and stride lengths have narrower limits of agreement. A narrow range between the limits indicates strong agreement.

4.5 Discussion

The results indicate that the Kinect measurements of stride velocity are highly similar to the Zeno Walkway measurements, with low relative error, low bias and a narrow limit of agreement. The step length and stride length have high correlations,

Table 4.3: Bland-Altman results as percentage.

		Limits of agreement (%)		
	Bias (%)	Lower	Upper	Range
Step Length [cm]	-10.9	-13.1	-8.6	4.4
Stride Length [cm]	-11.0	-13.8	-8.2	5.6
Stride Velocity [cm/s]	-1.4	-7.9	5.1	13.0
Stride Width [cm]	21.1	-1.3	43.4	44.7



(a) Step length (b) Stride velocity

Figure 4.4: Mean gait metrics of the Kinect plotted against the Zeno Walkway. The line of equality shows the ideal placement of the points.

but there is a significant negative bias. If the source of this bias is identified and corrected, the step and stride length could be in even stronger agreement than stride velocity. The stride width metric has the least agreement between systems.

The effectiveness of this approach to gait analysis relies on correctly detecting strides from peaks in the foot distance data. During a walking pass, some image frames may only contain noise. These are deleted by our system, making them blank. Because of this, the number of walking passes cannot be determined by simply counting the blocks of uninterrupted frames. Instead, the peak frame numbers are clustered, so that the peaks are correctly grouped by walking pass. If the number of walking passes is known beforehand, then k-means clustering is sufficient for this purpose, where k is the number of passes in front of the camera. If the number of passes is unknown or variable, the mean shift clustering algorithm is suitable, as it automatically determines the number of clusters.

4.6 Limitations and Future Work

As shown in the results, the Kinect camera system measures step and stride length with a negative bias. The cause of this bias will be addressed for future publications. Furthermore, the stride width calculation will be inspected and possibly revised to achieve a better agreement with the Zeno Walkway.

The Zeno Walkway measures gait metrics for the left and right sides, and for each individual stride. Future work could examine the agreement of the Kinect with these measurements.

The trials that were measured at the Recovery and Performance Laboratory by the Kinect and Zeno Walkway involved a variety of walking conditions. Specifically, participants either walked normally, or were asked to engage in a cognitively challenging task while walking (dual-tasking). These different types of walks will be analyzed separately in further work.

4.7 Conclusion

Participants with MS completed walking trials on a pressure sensitive walkway designed for gait analysis, the Zeno Walkway. They were simultaneously recorded by a Kinect camera from a side view. The PKMAS software was used to calculate gait metrics from the walkway measurements.

Four gait metrics were measured by the Kinect camera and the Zeno Walkway: step length, stride length, stride width, and stride velocity. The measurements from the first 8 walking trials have been presented, and the two systems have been compared. Strong agreement was found between the two systems with stride velocity, and medium to strong agreement with other gait metrics.

References

[1] O. Sofuwa, A. Nieuwboer, K. Desloovere, A.-M. Willems, F. Chavret, and I. Jonkers, “Quantitative gait analysis in parkinson’s disease: comparison with a healthy control group,” *Archives of physical medicine and rehabilitation*, vol. 86, no. 5, pp. 1007–1013, 2005.

[2] S. J. Olney, M. P. Griffin, and I. D. McBride, “Multivariate examination of data from gait analysis of persons with stroke,” *Physical Therapy*, vol. 78, no. 8, pp. 814–828, 1998.

[3] J. Behrens, C. Pfüller, S. Mansow-Model, K. Otte, F. Paul, and A. U. Brandt, “Using perceptive computing in multiple sclerosis-the short maximum speed walk test,” *Journal of neuroengineering and rehabilitation*, vol. 11, no. 1, p. 89, 2014.

[4] C. Morrison, M. D’Souza, K. Huckvale, J. F. Dorn, J. Burggraaff, C. P. Kamm, S. M. Steinheimer, P. Kontschieder, A. Criminisi, B. Uitdehaag *et al.*, “Usability and acceptability of assess ms: assessment of motor dysfunction in multiple sclerosis using depth-sensing computer vision,” *JMIR human factors*, vol. 2, no. 1, 2015.

[5] J. Behrens, K. Otte, S. Mansow-Model, A. Brandt, and F. Paul, “Kinect-based gait analysis in patients with multiple sclerosis,” *Neurology*, vol. 82, no. 10, p. P3, 2014.

[6] C. Pfueller, K. Otte, S. Mansow-Model, F. Paul, and A. Brandt, “Kinect-based analysis of posture, gait and coordination in multiple sclerosis patients (p04. 097),” *Neurology*, vol. 80, no. 7 Supplement, pp. P04–097, 2013.

[7] F. Gholami, D. A. Trojan, W. M. Haddad, B. Gholami *et al.*, "Gait assessment for multiple sclerosis patients using microsoft kinect," *arXiv preprint arXiv:1508.02405*, 2015.

[8] M. C. Kirkland, E. M. Wallack, S. N. Rancourt, and M. Ploughman, "Comparing three dual-task methods and the relationship to physical and cognitive impairment in people with multiple sclerosis and controls," *Multiple sclerosis international*, vol. 2015, 2015.

[9] C. Morrison, K. Huckvale, B. Corish, J. Dorn, P. Kontschieder, K. O'Hara, A. M. Team, A. Criminisi, and A. Sellen, "Assessing multiple sclerosis with kinect: designing computer vision systems for real-world use," *Human–Computer Interaction*, vol. 31, no. 3-4, pp. 191–226, 2016.

[10] D. Podsiadlo and S. Richardson, "The timed "up & go": a test of basic functional mobility for frail elderly persons," *Journal of the American geriatrics Society*, vol. 39, no. 2, pp. 142–148, 1991.

[11] H. B. Menz, M. D. Latt, A. Tiedemann, M. M. San Kwan, and S. R. Lord, "Reliability of the gaitrite® walkway system for the quantification of temporo-spatial parameters of gait in young and older people," *Gait & posture*, vol. 20, no. 1, pp. 20–25, 2004.

[12] S. Czarnuch and M. Ploughman, "Automated gait analysis in people with multiple sclerosis using two unreferenced depth imaging sensors: Preliminary steps," in *Proceedings of the 29th International Conference on Image and Vision Computing New Zealand, IVCNZ*, 2014, pp. 19–21.

[13] E. Dolatabadi, B. Taati, and A. Mihailidis, "Concurrent validity of the microsoft kinect for windows v2 for measuring spatiotemporal gait parameters," *Medical engineering & physics*, vol. 38, no. 9, pp. 952–958, 2016.

[14] T. Egerton, P. Thingstad, and J. L. Helbostad, "Comparison of programs for determining temporal-spatial gait variables from instrumented walkway data: Pkmas versus gaitrite," *BMC research notes*, vol. 7, no. 1, p. 542, 2014.

[15] R. C. Lynall, L. A. Zukowski, P. Plummer, and J. P. Mihalik, "Reliability and validity of the protokinetics movement analysis software in measuring center of pressure during walking," *Gait & posture*, vol. 52, pp. 308–311, 2017.

[16] S. M. Czarnuch and M. Ploughman, "Toward inexpensive, autonomous, and unobtrusive exercise therapy support for persons with ms," *Rehabilitation*, vol. 11, no. 2, pp. 150–157, 2016.

[17] A. Hynes and S. Czarnuch, "Combinatorial optimization for human body tracking," in *International Symposium on Visual Computing.* Springer, 2016, pp. 524–533.

[18] A. Hynes and S. Czarnuch, "Building a feature vector for assessing the gait of persons with multiple sclerosis," in *Newfoundland Electrical and Computer Engineering Conference*, 2016.

[19] C. Morrison, P. Culmer, H. Mentis, and T. Pincus, "Vision-based body tracking: turning kinect into a clinical tool," *Disability and Rehabilitation: Assistive Technology*, vol. 11, no. 6, pp. 516–520, 2016.

[20] G. Baldewijns, G. Verheyden, B. Vanrumste, and T. Croonenborghs, "Validation of the kinect for gait analysis using the gaitrite walkway," in *Engineering in*

Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE. IEEE, 2014, pp. 5920–5923.

[21] B. Müller, W. Ilg, M. A. Giese, and N. Ludolph, "Validation of enhanced kinect sensor based motion capturing for gait assessment," *PloS one*, vol. 12, no. 4, p. e0175813, 2017.

[22] D. J. Geerse, B. H. Coolen, and M. Roerdink, "Kinematic validation of a multi-kinect v2 instrumented 10-meter walkway for quantitative gait assessments," *PloS one*, vol. 10, no. 10, p. e0139913, 2015.

[23] E. Cippitelli, S. Gasparrini, S. Spinsante, and E. Gambi, "Kinect as a tool for gait analysis: validation of a real-time joint extraction algorithm working in side view," *Sensors*, vol. 15, no. 1, pp. 1417–1434, 2015.

[24] M. Gabel, R. Gilad-Bachrach, E. Renshaw, and A. Schuster, "Full body gait analysis with kinect," in *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE.* IEEE, 2012, pp. 1964–1967.

[25] R. A. Clark, K. J. Bower, B. F. Mentiplay, K. Paterson, and Y.-H. Pua, "Concurrent validity of the microsoft kinect for assessment of spatiotemporal gait variables," *Journal of biomechanics*, vol. 46, no. 15, pp. 2722–2725, 2013.

[26] A. Pfister, A. M. West, S. Bronner, and J. A. Noah, "Comparative abilities of microsoft kinect and vicon 3d motion capture for gait analysis," *Journal of medical engineering & technology*, vol. 38, no. 5, pp. 274–280, 2014.

[27] S. Motiian, P. Pergami, K. Guffey, C. A. Mancinelli, and G. Doretto, "Automated extraction and validation of children's gait parameters with the kinect," *Biomedical engineering online*, vol. 14, no. 1, p. 112, 2015.

[28] E. Cippitelli, S. Gasparrini, E. Gambi, and S. Spinsante, "A depth-based joints estimation algorithm for get up and go test using kinect," in *Consumer Electronics (ICCE), 2014 IEEE International Conference on.* IEEE, 2014, pp. 226–227.

[29] B. Galna, G. Barry, D. Jackson, D. Mhiripiri, P. Olivier, and L. Rochester, "Accuracy of the microsoft kinect sensor for measuring movement in people with parkinson's disease," *Gait & posture*, vol. 39, no. 4, pp. 1062–1068, 2014.

[30] R. A. Clark, S. Vernon, B. F. Mentiplay, K. J. Miller, J. L. McGinley, Y. H. Pua, K. Paterson, and K. J. Bower, "Instrumenting gait assessment using the kinect in people living with stroke: reliability and association with balance tests," *Journal of neuroengineering and rehabilitation*, vol. 12, no. 1, p. 15, 2015.

[31] T. Wei, B. Lee, Y. Qiao, A. Kitsikidis, K. Dimitropoulos, and N. Grammalidis, "Experimental study of skeleton tracking abilities from microsoft kinect non-frontal views," in *3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON), 2015.* IEEE, 2015, pp. 1–4.

[32] E. E. Stone and M. Skubic, "Passive in-home measurement of stride-to-stride gait variability comparing vision and kinect sensing," in *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE.* IEEE, 2011, pp. 6491–6494.

[33] R. A. Clark, Y.-H. Pua, K. Fortin, C. Ritchie, K. E. Webster, L. Denehy, and A. L. Bryant, "Validity of the microsoft kinect for assessment of postural control," *Gait & posture*, vol. 36, no. 3, pp. 372–377, 2012.

[34] A. Mirelman, L. Rochester, I. Maidan, S. Del Din, L. Alcock, F. Nieuwhof, M. O. Rikkert, B. R. Bloem, E. Pelosin, L. Avanzino *et al.*, "Addition of a non-immersive virtual reality component to treadmill training to reduce fall risk in

older adults (v-time): a randomised controlled trial," *The Lancet*, vol. 388, no. 10050, pp. 1170–1182, 2016.

[35] MATLAB. findpeaks - Find local maxima. [Online]. Available: https://www.mathworks.com/help/signal/ref/findpeaks.html

[36] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median," *Journal of Experimental Social Psychology*, vol. 49, no. 4, pp. 764–766, 2013.

[37] M. M. Mukaka, "A guide to appropriate use of correlation coefficient in medical research," *Malawi Medical Journal*, vol. 24, no. 3, pp. 69–71, 2012.

[38] J. M. Bland and D. Altman, "Statistical methods for assessing agreement between two methods of clinical measurement," *The lancet*, vol. 327, no. 8476, pp. 307–310, 1986.

[39] D. Giavarina, "Understanding bland altman analysis," *Biochemia medica: Biochemia medica*, vol. 25, no. 2, pp. 141–151, 2015.

Chapter 5

Gait Analysis with a Side-View Depth Sensor via Optimal Selection of Human Joint Proposals

Andrew Hynes, Stephen Czarnuch, Megan C. Kirkland, and Michelle Ploughman

Abstract

Objective: We propose a method for calculating standard gait parameters from individual joints with a side-view depth sensor. *Methods:* Clinical walking trials were measured concurrently by a side-view Kinect and a pressure-sensitive walkway, the Zeno Walkway. Multiple joint proposals were generated from depth images by a stochastic predictor based on the Kinect algorithm. The proposals are represented as vertices in a weighted graph, where the weights depend on the expected and measured lengths between body parts. A shortest path through the graph is a set of joints from head to foot. Accurate foot positions are selected by comparing pairs of shortest paths. Stance phases of the feet are detected by examining the motion of the feet over time. The stance phases are used to calculate five gait parameters: stride length, step length, stride width, stride velocity, and stance percentage. *Results:* Gait parameters from 52 trials were compared to the ground truth using Bland-Altman analysis and intraclass correlation coefficients. The large spatial parameters had the strongest agreement with the walkway (ICC(2, 1) = 0.991 and 0.985 for stride and step length, respectively). *Conclusion:* Clinical gait parameters can be accurately measured from individual foot positions without the need for frontal tracking with the Kinect. *Significance:* The presented system directly calculates gait parameters from individual foot positions while previous side-view systems rely on indirect measures.

5.1 Introduction

The analysis of human gait is an important component of treating walking disorders [1], which arise from neurological diseases including cerebral palsy [2] and multiple sclerosis (MS) [3–6]. Clinical gait analysis is commonly performed with timed walking tests [7, 8]. For a deeper analysis, quantitative gait measures can be obtained using pressure-sensitive walkways such as GAITRite [5] or the Zeno Walkway [9]. By recording the positions of feet over time, pressure walkways can be used to calculate spatial and temporal gait parameters. They can also measure kinetic properties such as the centre of pressure of the foot. However, walkways are unable to directly measure the kinematics of body parts other than the feet. Full-body gait analysis has been performed using sensors attached to the body [10–12], or by tracking markers on the body with a motion capture system [13].

Human pose estimation from depth sensors has seen large advances in recent years, notably with the release of the Microsoft Kinect [14]. A large volume of research has now investigated the Kinect as a device for gait analysis [13, 15–23]. The advantages of gait analysis with a depth sensor include the abilities for long-term monitoring in a home setting [16] and tracking the full body at a low cost without wearable sensors or markers.

Gait analysis with the Kinect is often conducted using the Kinect Software Development Kit (SDK) to process the depth images captured by the camera [15, 18–21, 23, 24]. The SDK outputs a skeleton model of the human body with 20 joints at 30 frames per second [15]. Behrens et al. introduced a computerized measure for gait analysis in persons with MS, the Short Maximum Speed Walk (SMSW) test [24]. The parameters of SMSW were calculated using the positions of the hip-centre joint extracted from the SDK. This new test was found to be correlated with established clinical measures including the Timed 25-Foot Walk. Gabel et al. derived a feature vector from multiple consecutive frames of skeleton data from the SDK [15]. A regression model used this vector to predict stride durations and arm angular velocities, which were validated against ground truth data from wearable sensors. Gait parameters have been measured concurrently with the Kinect SDK and the GAITRite mat in both children [21] and healthy adults [17, 23].

The Kinect SDK is intended to track the human skeleton from a frontal perspective [22], which can be inconvenient in a clinical setting. In [23], a Kinect camera was placed at each end of a GAITRite mat so the subject would be tracked from a frontal perspective while walking in either direction on the mat. Participants in [21] walked in only one direction on the GAITRite (towards a Kinect camera at the end of the mat). A second side-view camera captured depth images for assisting with manual annotation.

In response to the limitations of the SDK, gait analysis with a non-frontal Kinect has been explored. Cippitelli et al. presented an algorithm for a side-view Kinect that functions without machine learning [22]. However, a calibration step was required in which the subject faces the sensor with outstretched arms. The lengths between adjacent body joints were calculated from this calibration image. The system tracked six joints visible on one side of the body, in order to produce an objective score for the Get Up and Go Test (GUGT), which involves standing from an arm-less chair and beginning to walk. While the six joints were sufficient for GUGT, spatial gait parameters such as stride length require separate foot positions to be measured directly. Baldewijns et al. used the SDK to extract the binary image of the person from a side view, but not to track the full skeleton model [17]. Step length and step time were then calculated indirectly by analyzing the centre of mass of the binary image. Stone and Skubic [16] performed continuous and long term monitoring of older adults with an environmentally mounted Kinect in their apartments. A probabilistic model was used to estimate gait parameters rather than tracking a skeleton, limiting the applicability of this approach to clinical gait assessment.

The tracking ability of the Kinect SDK is rooted in a machine learning algorithm developed by Shotton et al. [25]. Given a single depth image, the trained system produces multiple proposals for the 3D positions of human joints. Each joint proposal is associated with a confidence value indicating the likelihood that the position is correct. The different human joints are identified independently (i.e., without information from other image frames or the kinematic constraints of the body). Unfortunately, the process from the joint proposals to the final smooth tracking of the human skeleton is a proprietary and unpublished algorithm [26].

We use a predictor first developed for overhead hand tracking [27] which is based on the algorithm of Shotton et al. The predictor has now been trained to output multiple joint proposals from side-view depth images of the human body. However, inaccurate proposals can be generated due to the stochastic nature of the predictor, by mistaking one body part for another, or by detecting noise in the background. Therefore, we first present a method to select accurate joints from the proposals. The problem of optimally selecting from multiple human joint proposals has also been applied to pose estimation in RGB videos [28] and multi-person pose estimation in RGB images [29].

We present three main contributions:

1. A method to select accurate head and foot positions from multiple joint proposals. After estimating the fixed lengths of links between body parts, the best head and foot positions are selected on a per-frame basis. The feet are then assigned to left and right sides based on the direction of walking motion.

2. A method to calculate standard spatiotemporal gait parameters from the selected positions.

3. A concurrent validation with the Zeno Walkway.

Our system is tested on a dataset of 52 walking trials recorded at the Recovery and Performance Laboratory of Memorial University. The study was approved by the Health Research Ethics Board of Newfoundland and Labrador. Participants with MS were measured concurrently by a Zeno Walkway and a Kinect v1 camera from a side view. Each trial consists of multiple passes across the walkway in both directions. Gait parameters were calculated from the Zeno Walkway data by the Protokinetics Movement Analysis Software (PKMAS), which uses calculations as defined by Huxham et al. [30].

The depth images of two extra walking trials have been manually labelled to provide ground truth positions of body parts. The first section of our method is tested by comparing the selected positions to the ground truth, and the second section is tested by comparing our gait parameters to those calculated by PKMAS.

The paper is organized as follows. Section 5.2 presents the method to select accurate head and foot positions from multiple proposals and to assign left/right sides to the feet. Section 5.3 presents the method to calculate spatiotemporal gait parameters from these positions. Section 5.4 reports the results of both sections of our method. We discuss various aspects of our approach in Section 5.5 and conclude in Section 5.6.

5.2 Pose Estimation

Given a single depth image, the predictor introduced in [27] outputs multiple proposals for various body parts. While separate proposals are generated for left and right parts, we group the proposals by part type, removing the left/right distinction. This is intended to avoid situations when the predictor only generates left proposals for a right part, or vice versa.

Let $\mathcal{P}$ be the set of all joint proposals on one frame captured by the depth sensor. These proposals are positions in 3D space. $\mathcal{P}$ is partitioned into subsets representing body part types. We utilize six part types: head, hip, thigh, knee, calf, and foot. Figure 5.1 shows a two-dimensional view of the positions in $\mathcal{P}$ labelled by part type.

Our method is based on the assumption that the links between consecutive pairs of parts (head to hip, hip to thigh, etc.) have fixed lengths [31]. These lengths are first estimated for the walking trial. Then, the joint proposals of each frame are represented as a weighted graph, with edge weights dependent on the difference between the expected lengths for the trial and the measured lengths on the frame. A shortest path from head to foot in this graph finds a set of body parts with lengths similar to the expected lengths.

5.2.1 Length Estimation

The part types are indexed in order from head to foot. For a given walking trial, a fixed length is estimated between each consecutive pair of part types. A frame in the trial is included in the calculation if at least one proposal exists for each part type.

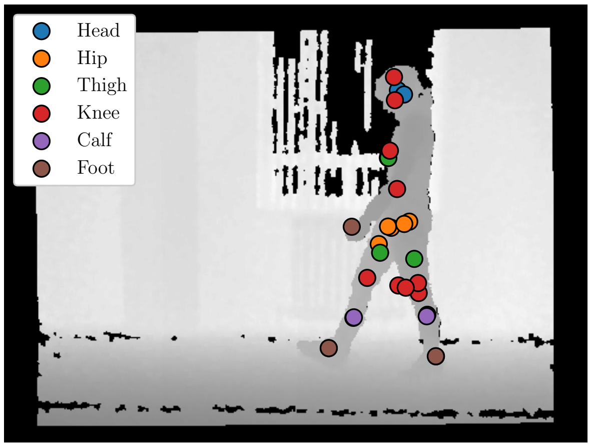


Figure 5.1: Example of joint proposals generated on a depth frame.

On each of these frames, the distance is measured between each proposal of type t and each proposal of type $t + 1$.

We assume that the measured distances between consecutive parts fall into two major categories: those between proposals on the same side of the body and those between proposals on opposite sides. The distances in the former category are expected to be shorter than those in the latter, especially if the legs are at full stride. Therefore, the length estimate on one frame is calculated as the median of the lower half of the measured distances, also known as the first quartile. However, this grouping does not apply to the distances from head to hip, so the length is estimated as the median of the distances rather than the first quartile.

Once a length has been estimated for each frame, the final estimate for the trial is the median of all frame estimates.

5.2.2 Graph Representation

The joint proposals of $\mathcal{P}$ are represented as the vertices of a weighted graph G. The vertices of part type t form a complete bipartite graph with the vertices of part type $t+1$ (i.e., there is an edge between each vertex of type t and each vertex of type $t+1$, and no edges between vertices of the same type [32]). The edges are directed from type t to type $t+1$. An example of graph G is shown in Figure 5.2.

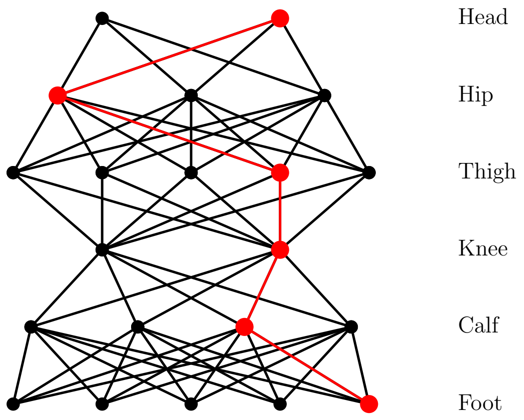


Figure 5.2: Graph representation of joint proposals. Each proposal is represented by one vertex in the graph. The red vertices and edges show a possible shortest path from head to foot. The path consists of one proposal for each part type.

Each proposal i has a 3D position $\mathbf{p}_i$ and a part type t_i. The measured length l_{ij} between two proposals i and j is

$$l_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\| \tag{5.1}$$

If $t_j = t_i + 1$, then proposals i and j are connected by a directed edge $i \to j$ in G, and there is an expected length $l_{t_i t_j}$ between parts of type t_i and parts of type t_j. As introduced in [31], the weight w_{ij} of the edge is

$$w_{ij} = \left(l_{ij} - l_{t_i t_j}\right)^2 \tag{5.2}$$

5.2.3 Shortest Paths

After the weighted graph G has been constructed, an algorithm is run to find the shortest path to each vertex representing a foot proposal. A shortest path between two vertices u and v is a path along the edges from u to v with the lowest possible sum of edge weights [33]. Figure 5.2 shows a possible shortest path in G from head to foot.

Since each edge in G is directed from a vertex of part type t to one of type $t+1$, there are no paths in the graph that can begin and end on the same vertex. Thus, G is classified as a directed acyclic graph. The shortest path algorithm for a single-source directed acyclic graph can be found in [33]. The vertices of the graph must be in topological order before the algorithm is run. A topological ordering is a sequence of vertices such that for each edge $u \rightarrow v$, u appears before v in the ordering. A topological ordering for G is obtained by listing the vertices of each part type in order from head to foot.

In a single-source shortest path problem, all paths must begin from the same vertex, the source. G can be represented as a single-source graph by adding one source vertex, connected to each head vertex by a directed edge with zero weight. Thus, the shortest path to each head vertex is zero.

Algorithm 2 summarizes the shortest path algorithm that is used on G. It returns two variables $prev$ and $dist$, each being an array with $|\mathcal{P}|$ elements. The element $prev[i]$ is the vertex previous to vertex i in the shortest path to i, and $dist[i]$ is the total distance (sum of edge weights) of the shortest path to i.

The algorithm begins by assuming $dist[i]$ is zero for the head vertices and infinite for the others. For each vertex u taken in topological order, each vertex v adjacent to u is considered. If $dist[u]$ plus the weight w_{uv} is less than $dist[v]$, then u is set as

Algorithm 2 Shortest Paths on the Directed Acyclic Graph G

```
Input:   G      Graph representation of joint proposals
Output:  dist   Total distances of the shortest paths
         prev   Previous vertices on the shortest paths
 1: function ShortestPaths($G$)
 2:     for each vertex $v \in G$ do
 3:         $prev[v] \leftarrow$ null
 4:         if $v$ represents a head proposal then
 5:             $dist[v] \leftarrow 0$
 6:         else
 7:             $dist[v] \leftarrow \infty$
 8:     for each vertex $u \in G$ do
 9:         for each vertex $v$ adjacent to $u$ do
10:             $alternative \leftarrow dist[u] + w_{uv}$
11:             if $alternative < dist[v]$ then
12:                 $dist[v] \leftarrow alternative$
13:                 $prev[v] \leftarrow u$
14:     return $dist$, $prev$
```

the previous vertex to v, and the distance to v is reduced to the lower value. After the algorithm terminates, the shortest path to each vertex can be found by tracing back the vertices in *prev*.

While Algorithm 2 finds a shortest path to every vertex in G, the term 'shortest path' will refer only to a path ending on a foot vertex for the remainder of the paper. The structure of G guarantees that such a path consists of exactly one vertex for each part type, as demonstrated in Figure 5.2.

5.2.4 Foot Selection

There are n_{foot} proposals for foot positions in a frame. From these, two must be selected as the best estimates for the actual feet of the walking person. This is achieved by comparing all possible pairs of shortest paths to the foot vertices. For n_{foot} paths, there are $\binom{n_{foot}}{2}$ pairs of paths.

A subset of proposals, $\mathcal{P}_{paths}$, is taken from the set of all joint proposals $\mathcal{P}$. A proposal is in $\mathcal{P}_{paths}$ if it is included in any of the shortest paths. Many noisy joint proposals are absent in $\mathcal{P}_{paths}$, as evident in Figure 5.3.

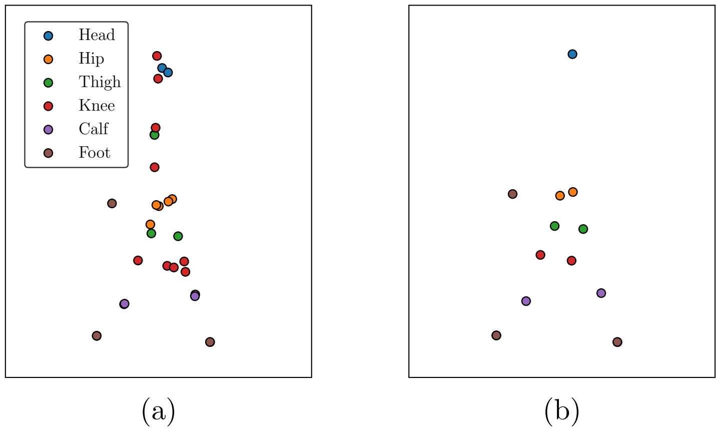


Figure 5.3: Removing noisy joint proposals. (a) $\mathcal{P}$, the set of all joint proposals on a frame. (b) $\mathcal{P}_{paths}$, a subset of $\mathcal{P}$. A proposal is in $\mathcal{P}_{paths}$ if it is included in any of the shortest paths to the feet.

Scores are now assigned to links between proposals in $\mathcal{P}_{paths}$. A score S_{ij} exists between two proposals i and j if both of the following conditions are true:

1. There is a fixed length between the part types t_i and t_j.
2. Proposals i and j are on the same shortest path.

The score is calculated with a simple quadratic function.

$$S_{ij} = -(x-1)^2 + 1 \tag{5.3}$$

where x is the ratio between the measured length l_{ij} and expected length $l_{t_it_j}$. The ratio is calculated by dividing the greater length by the lesser length, so that $x \geq 1$.

$$x = \frac{\max(l_{ij},\ l_{t_it_j})}{\min(l_{ij},\ l_{t_it_j})} \tag{5.4}$$

The links between consecutive body parts are the only links represented by edges in G. However, there are additional links of fixed length between the lower body parts: hip to knee, and knee to foot. The expected length from hip to knee is calculated as the sum of the expected lengths from hip to thigh and thigh to knee, since all three parts should lie in a straight line. The same applies to the knee, calf, and foot. Scores are assigned to the links between consecutive parts as well as these additional links.

Like the edge weight w_{ij} in G, the score S_{ij} is dependent on the expected and measured lengths between joint proposals i and j. While w_{ij} is restricted to non-negative values (a consequence of Equation (5.2)), S_{ij} can be positive or negative. The highest possible score is one, which occurs when the measured length is equal to the expected length. When the ratio of the lengths is greater than two, the score becomes negative. The quadratic function defined in Equation (5.3) is shown in Figure 5.4.

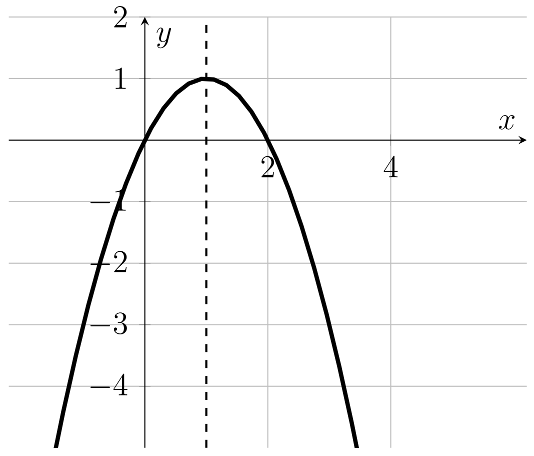


Figure 5.4: Plot of the score function defined in Equation (5.3). The scores are restricted to the right side of the dashed vertical line $x = 1$.

Once the scores have been assigned, all possible pairs of shortest paths are compared. Algorithm 3 summarizes the process to select the best pair of shortest paths. $\mathcal{P}_{pair}$ is the set of positions included in a pair of paths. A sphere of radius r is cen-

Algorithm 3 Foot Selection

```
Input:   $pairs$              All pairs of shortest paths
         $radii$              Array of radii for the spheres
         $\mathcal{P}_{paths}$  Set of proposals along the paths
         $S$                  Scores of links between proposals
Output:  $pair_{best}$        Best pair of shortest paths
1:  function SELECTBESTPAIR($pairs$, $radii$, $\mathcal{P}_{paths}$, $S$)
2:      $n_p \leftarrow$ number of pairs
3:      $votes \leftarrow$ array of $n_p$ zeros
4:      for $r \in radii$ do
5:          $scores \leftarrow$ array of $n_p$ zeros
6:          for $pair \in pairs$ do
7:              $\mathcal{P}_{pair} \leftarrow$ Set of positions in $pair$
8:              $V_{spheres} \leftarrow$ combined volume of spheres
                    centred on positions in $\mathcal{P}_{pair}$
9:              $s_{total} \leftarrow 0$
10:             for $\mathbf{p}_i \in \mathcal{P}_{paths}$ do
11:                 for $\mathbf{p}_j \in \mathcal{P}_{paths}$ do
12:                     if $\mathbf{p}_i$ and $\mathbf{p}_j$ are both in $V_{spheres}$ then
13:                         $s_{total} \leftarrow s_{total} + S_{ij}$
14:             $scores[pair] \leftarrow s_{total}$
15:         $winners_r \leftarrow$ all pairs with a score equal to
                $\max(scores)$ for radius $r$
16:         $votes(winners_r) \leftarrow votes(winners_r) + 1$
17:     $pair_{best} \leftarrow pairs[\text{argmax}(votes)]$
18:     return $pair_{best}$
```

tred on each position in $\mathcal{P}_{pair}$. If positions $\mathbf{p}_i$ and $\mathbf{p}_j$ from $\mathcal{P}_{paths}$ both lie inside the combined volume of spheres, the score S_{ij} is added to the total score for the pair of paths (note that $S_{ij} = 0$ unless $\mathbf{p}_i$ and $\mathbf{p}_j$ are on the same path). Figure 5.5 shows the spheres of one radius on different pairs of paths, and the links with non-zero scores that are included by these spheres.

After a total score has been calculated for each pair, a vote is given to the pair with the highest score. In the case of a tie, a vote is given to each pair tied for the top score. The process repeats with a new radius for the spheres, and the votes for

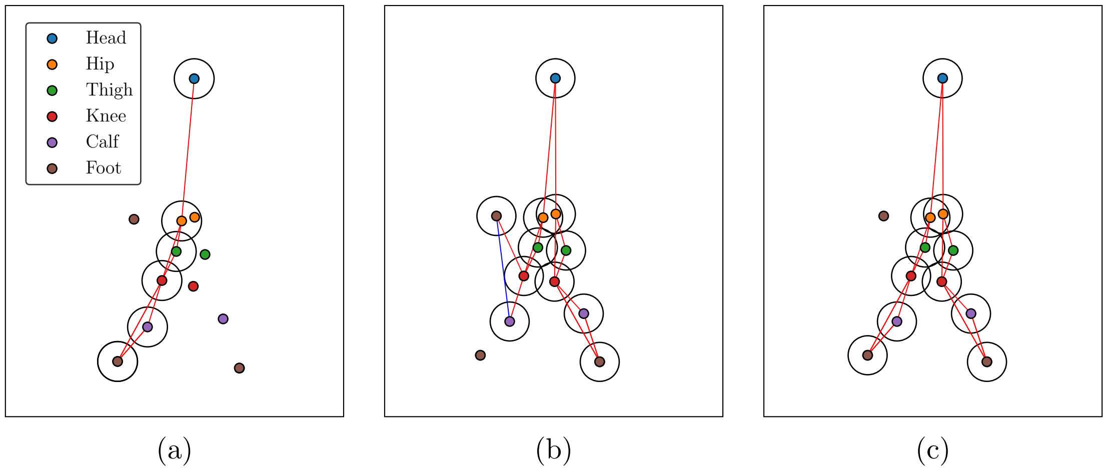


Figure 5.5: Comparing pairs of shortest paths. The visible points constitute $\mathcal{P}_{paths}$, a subset of $\mathcal{P}$ (see Figure 5.3). A sphere of radius r is centred on each joint proposal in the two paths. The lines between points indicate links with non-zero scores, ranging from negative (blue) to positive (red). (a) The two paths end on nearby foot positions. Several links with good scores (from the other leg) are missed by the spheres. (b) One path ends on an incorrect (noisy) foot position. The links to this foot have negative scores, reducing the total score of the pair. (c) The spheres include links with good scores from both sides of the body. This pair of paths has the highest total score for radius r.

the pairs are accumulated.

When the votes have been counted over a range of radii, the pair with the most number of votes is selected. The two foot positions from this pair are deemed to be the best estimates for the actual feet on the frame.

5.2.5 Head Selection

When a frame includes multiple proposals for the head position, the two shortest paths selected in Section 5.2.4 could include two different head proposals. When this occurs, the path with the lower total weight defines the selected head position.

5.2.6 Side Assignment

Two foot positions have now been selected on each frame in a walking trial: $\mathbf{p}_{foot,1}$ and $\mathbf{p}_{foot,2}$. These must now be correctly labelled as the left and right foot.

Walking Passes

The trials captured by the Kinect have a varying number of walking passes across the Zeno Walkway. For each pass, the participant would enter the field of view, walk across the walkway, and exit the field of view on the opposite side. This process ensured a number of empty frames between each pass. To identify the passes, the numbers of the non-empty frames in a trial are clustered with DBSCAN [34], which determines the number of clusters automatically. Each detected cluster of frame numbers is treated as one walking pass. DBSCAN also labels data points as noise if they are too far from the core clusters. Any frames identified as noise are excluded from the following calculations.

Dimension Reduction

The selected head and foot positions are now converted from 3D to 2D. The original position vectors have coordinates (x, y, z), where x is along the walkway, y is along the height of the person, and z is the depth. The positions are projected onto the xz plane, and are then treated as new 2D vectors with coordinates (x, y), where x is along the depth axis, and y is along the walkway.

Line of Best Fit

The general direction of the walking pass is used to determine the left and right sides of the body. This direction is estimated by calculating the line of best fit of the

2D head positions in the pass.

A line is defined by a position in space and a direction vector. The position defining the line of best fit is the centroid of the head positions, $\mathbf{p}_{centroid}$. The direction vector $\mathbf{v}_{forward}$ is obtained by applying principal component analysis on the set of head positions [35].

Finally, to ensure that $\mathbf{v}_{forward}$ is actually pointed in the forward walking direction, we compute its dot product with the vector from the initial head position in the pass, $\mathbf{p}_{head,i}$, to the final, $\mathbf{p}_{head,f}$.

$$p = (\mathbf{p}_{head,f} - \mathbf{p}_{head,i}) \cdot \mathbf{v}_{forward} \tag{5.5}$$

If the product p is negative, the direction of $\mathbf{v}_{forward}$ is reversed.

Side Evaluation

A value v_{side} is calculated for each frame in the walking pass. This value represents the amount that $\mathbf{p}_{foot,1}$ is left or right of $\mathbf{p}_{foot,2}$ given the current walking direction.

The cross product of two vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ is a vector perpendicular to both. If $\mathbf{v}_1$ and $\mathbf{v}_2$ lie in the xy plane, their cross product will be directed along the z axis. The direction of the cross product depends on the orientation of $\mathbf{v}_1$ relative to $\mathbf{v}_2$.

Therefore, the value v_{side} is calculated as follows:

$$\begin{aligned} \mathbf{v}_{2\to1} &= \mathbf{p}_{foot,1} - \mathbf{p}_{foot,2} \\ \mathbf{v}_{cross} &= \mathbf{v}_{2\to1} \times \mathbf{v}_{forward} \\ v_{side} &= \mathbf{v}_{cross}[z] \end{aligned} \tag{5.6}$$

If $v_{side} > 0$, then $\mathbf{p}_{foot,1}$ is to the right of $\mathbf{p}_{foot,2}$ relative to the direction vector

$\mathbf{v}_{forward}$. A foot further to the right will have a greater value of v_{side}.

A simple method to assign sides would be to label $\mathbf{p}_{foot,1}$ as the right foot on each frame that v_{side} is positive. However, if $\mathbf{v}_{2\to1}$ moved slightly to the left of $\mathbf{v}_{forward}$ on a frame, then $\mathbf{p}_{foot,1}$ would be incorrectly labelled as left, causing abrupt jumps in the motions of the feet.

Instead, $\mathbf{p}_{foot,1}$ and $\mathbf{p}_{foot,2}$ are labelled as foot A and foot B by establishing a motion correspondence over the whole walking pass. In essence, foot A refers to the same actual foot on each frame and foot B refers to the other foot. Afterwards, feet A and B are labelled as left and right.

Motion Correspondence

Let $\mathbf{P}_f$ be the matrix containing the two foot positions on frame f. A simple calculation is used to link $\mathbf{P}_f$ to $\mathbf{P}_{f+1}$. Let $\mathbf{D}$ be the 2×2 matrix of pairwise distances between $\mathbf{P}_f$ and $\mathbf{P}_{f+1}$. $\mathbf{D}_{ij}$ is the distance from $\mathbf{P}_f[i]$ to $\mathbf{P}_{f+1}[j]$.

The selected assignment α is the one that minimizes the total distance travelled by the feet from one frame to the next.

$$\alpha = \operatorname{argmin}\left(\mathbf{D}_{00} + \mathbf{D}_{11},\ \mathbf{D}_{01} + \mathbf{D}_{10}\right) \tag{5.7}$$

While α avoids sudden jumps in the foot positions, there is still a risk that the tracking will switch when the feet come together. Therefore, the walking pass is divided into contiguous sections of frames.

An approximately periodic signal is obtained by calculating the distance between the two feet on each frame in the pass. A peak detection algorithm is run on this signal to find the local minima. The signal and detected minima are shown in Figure 5.6. The sections of the walking pass are the ranges of frames between the minima (there are $n + 1$ sections for n detected minima). A section represents a time period when the feet are apart.

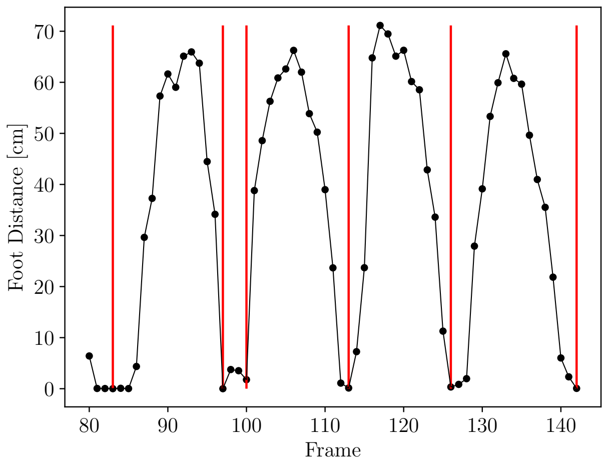


Figure 5.6: Distance between feet during a walking pass. The vertical lines indicate the local minima. The pass is split into the sections of frames between these minima.

For each section of the walking pass, foot A is initially assumed to be the left foot. Equation (5.6) is computed on each frame in the section and the resulting v_{side} values are summed. If the sum is positive, foot A is instead labelled as the right foot for the section.

5.3 Gait Analysis

The left and right feet have now been identified for each frame in the walking pass, allowing for the calculation of gait parameters for both sides of the body. The left and right foot positions are analyzed separately to determine their respective stance and swing phases. Standard gait parameters are then calculated using these phases.

5.3.1 Phase Detection

The stance and swing phases of a foot are identified by analyzing the motion of the foot over time. The displacement of the foot is expected to be significantly greater during the swing phases than during the stance phases.

The line of best fit from Section 5.2.6 is used to create a one-dimensional signal for detecting the phases. As described above, the line is defined by a position $\mathbf{p}_{centroid}$ and a vector $\mathbf{v}_{forward}$.

$\mathbf{V}_{foot}$ is the set of vectors from $\mathbf{p}_{centroid}$ to all positions of one foot in the pass, $\mathbf{P}_{foot}$.

$$\mathbf{V}_{foot} = \mathbf{P}_{foot} - \mathbf{p}_{centroid} \tag{5.8}$$

The signal Φ is found by taking the dot product of the direction vector $\mathbf{v}_{forward}$ with each vector in $\mathbf{V}_{foot}$. This transforms the vectors into one-dimensional values.

$$\Phi = \{\, \mathbf{v}_{forward} \cdot \mathbf{v} \mid \mathbf{v} \in \mathbf{V}_{foot} \,\} \tag{5.9}$$

A sliding window is centred on each frame in the walking pass. The variance of Φ values in the window is calculated for each frame. The variances are then clustered with k-means, where $k = 2$. The cluster with the smaller mean value corresponds to frames in the stance phase. The signal Φ and the result of the clustering are shown in Figure 5.7. It is clear that the stance phases correspond to flat sections of the signal.

Some frames in a walking pass can be missing foot data, causing a null value in the signal Φ. In these cases, the variance of neighbouring values in the sliding window is still calculated and the variance is included in the clustering algorithm. Thus, the blank frames can still be labelled as stance or swing.

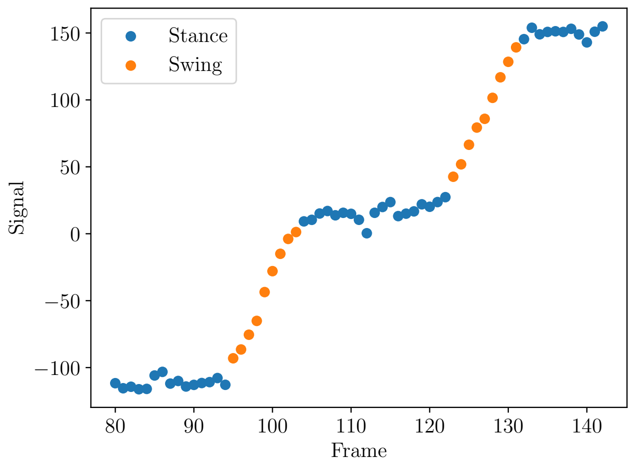


Figure 5.7: Signal Φ of one foot during a walking pass. The frames are labelled stance and swing by clustering the local variance using k-means.

Finally, the detected stance phases are filtered to avoid false positives. Any stance phase containing fewer than 10 non-blank frames is relabelled as a swing phase.

5.3.2 Gait Parameters

The PKMAS manual defines a stride as the first contact of one foot on the floor to the proceeding first contact of the same foot. The other foot is in the stance phase during this stride. Thus, stride i for foot a is defined by three positions: $\mathbf{p}_{a,i}$, $\mathbf{p}_{b,i}$, and $\mathbf{p}_{a,i+1}$. Figure 5.8 illustrates the foot positions of a stride and the corresponding spatial gait parameters. In this diagram, $\mathbf{p}_{a,i}$ is right foot 1, $\mathbf{p}_{b,i}$ is left foot 1, and $\mathbf{p}_{a,i+1}$ is right foot 2.

The detected stance phases from Section 5.3.1 are used to estimate the positions defining a stride. There is a signal Φ for the left foot and the right foot. The median frame and foot position are calculated for each stance phase from the left and right signals. The left and right stance phases are then grouped together and ordered by median frame. A toy example is shown in Table 5.1. Each group of three consecutive

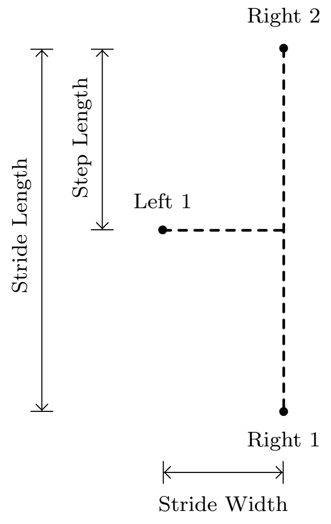


Figure 5.8: Spatial gait parameters for right foot 1.

stance phases constitute a stride, where the three median positions are $\mathbf{p}_{a,i}$, $\mathbf{p}_{b,i}$, and $\mathbf{p}_{a,i+1}$, respectively.

Table 5.1: Example of stance phases ordered by median frame (positions are not real samples).

Median Frame	Median Position	Stance No.	Side
88.0	$(20, 0)$	1	Right
95.5	$(10, 50)$	1	Left
113.0	$(20, 100)$	2	Right
127.5	$(10, 150)$	2	Left

The stride length is the distance from $\mathbf{p}_{a,i}$ to $\mathbf{p}_{a,i+1}$.

$$l_{stride,a,i} = \|\mathbf{p}_{a,i+1} - \mathbf{p}_{a,i}\| \tag{5.10}$$

The step length and stride width depend on $\mathbf{p}_{b,i,proj}$. This is the projection of $\mathbf{p}_{b,i}$ onto the line defined by $\mathbf{p}_{a,i}$ and $\mathbf{p}_{a,i+1}$.

$$l_{step,a,i} = \|\mathbf{p}_{a,i+1} - \mathbf{p}_{b,i,proj}\| \tag{5.11}$$

$$w_{stride,a,i} = \|\mathbf{p}_{b,i} - \mathbf{p}_{b,i,proj}\| \tag{5.12}$$

The first and last frames, F_{first} and F_{last}, are also recorded for each stance phase. These approximate the first and last instances that the foot contacts the walkway.

The stride time is the time from the first contact of one foot to the following first contact of the same foot. The difference of frames is divided by the frame rate in frames per second, fps, to obtain a time in seconds. We use a frame rate of 30 frames per second, the theoretical frame rate of the Kinect.

$$t_{stride,a,i} = \frac{F_{first,a,i+1} - F_{first,a,i}}{fps} \tag{5.13}$$

The stride velocity is the stride length divided by the stride time.

$$v_{stride,a,i} = \frac{l_{stride,a,i}}{t_{stride,a,i}} \tag{5.14}$$

The stance time is the time from the first contact to the last contact of the same foot.

$$t_{stance,a,i} = \frac{F_{last,a,i} - F_{first,a,i}}{fps} \tag{5.15}$$

The stance percentage is the stance time divided by the stride time.

$$p_{stance,a,i} = \frac{t_{stance,a,i}}{t_{stride,a,i}} \cdot 100 \tag{5.16}$$

These gait parameters are calculated for each stride in the walking pass and for each pass in the trial.

5.4 Experiments and Results

5.4.1 Datasets

The dataset used for gait analysis consists of 52 walking trials measured concurrently by a Zeno Walkway and Kinect v1 depth sensor. The trials were performed by four participants, who completed 6, 14, 14, and 18 trials, respectively. Joint proposals were generated for each depth frame by the predictor originally developed in [27]. Our method for selecting joint proposals (Section 5.2) was applied to the frames containing at least one proposal for each part type of interest (head, hip, thigh, knee, calf, and foot). A total of 18219 frames were processed from the 52 trials.

Two additional walking trials were captured only by the Kinect in the same environment. The depth images of these trials have been manually labelled to obtain ground truth part positions. Figure 5.9 shows a depth image and corresponding label image from one of these trials. The human form in the depth image is segmented into distinct body parts in the label image. Ground truth positions of body parts were obtained by computing the median position of each segment of pixels, then converting from image coordinates to real world coordinates. The two trials encompass 581 frames that have both joint proposals and ground truth positions for the head and two feet.

The method was implemented in Python on an Intel Core i5-8250U (1.60 GHz) quad-core processor.

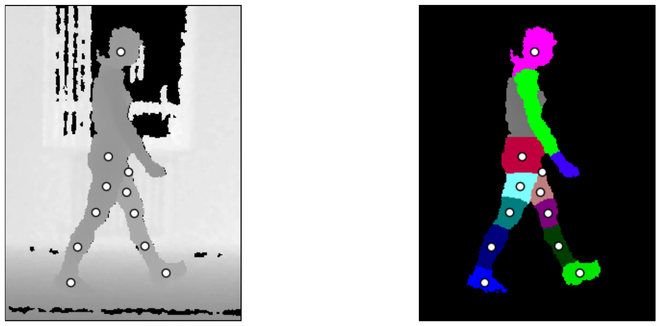

(a) Depth image (b) Label image

Figure 5.9: Depth image and corresponding label image. The white dots indicate the median positions of body part segments in the label image.

5.4.2 Pose Estimation

Length Estimation

Ground truth lengths were calculated from the two labelled trials by measuring the lengths of body links on each frame for the left and right sides. The median length of each body link was calculated over the whole trial (left and right values were grouped together), resulting in five final lengths. Table 5.2 compares the ground truth lengths to those estimated in Section 5.2.1. The relative error ranged from −14 % to 15 %.

In the case of the 52 trials without ground truth positions, the length estimation process can still be analyzed by grouping the trials by participant. Ideally, the estimated lengths should remain constant over different trials with the same participant. Table 5.3 shows the mean and standard deviation of the estimated lengths for each of the four participants. The greatest standard deviations were 0.79 cm for head to hip, 0.61 cm hip to thigh, 0.88 cm thigh to knee, 0.79 cm knee to calf, and 0.55 cm calf to foot.

Table 5.2: Comparison of estimated and ground truth lengths.

Link	Trial	Estimated	Ground Truth	Relative Error
Head → Hip	1	70.67	70.64	0.00
	2	70.52	70.64	0.00
Hip → Thigh	1	18.47	19.47	-0.05
	2	16.73	19.47	-0.14
Thigh → Knee	1	18.16	16.13	0.13
	2	15.26	16.13	-0.05
Knee → Calf	1	26.24	22.90	0.15
	2	25.66	22.90	0.12
Calf → Foot	1	24.48	23.58	0.04
	2	22.63	23.58	-0.04

Table 5.3: Mean and standard deviation of estimated lengths grouped by participant.

	Head → Hip		Hip → Thigh		Thigh → Knee		Knee → Calf		Calf → Foot	
	mean	std	mean	std	mean	std	mean	std	mean	std
Participant										
1	70.03	0.41	15.08	0.50	14.67	0.33	22.16	0.79	23.49	0.15
2	69.59	0.79	18.17	0.48	17.30	0.43	25.75	0.37	24.60	0.38
3	70.68	0.48	16.06	0.61	16.89	0.88	27.80	0.62	25.04	0.55
4	71.41	0.76	15.47	0.55	14.85	0.45	24.43	0.39	23.01	0.20

Head and Foot Selection

The head and two foot positions were selected on each frame before assigning left/right sides to the feet (Section 5.2.4 and Section 5.2.5). The process was found to run at approximately 200 frames per second.

We defined accuracy as the percentage of frames where the selected position is within a distance D of the ground truth position. Following the convention of [25], we set $D = 10\,\text{cm}$. The selected head positions achieved an accuracy of 98 %.

In order to compare the two selected feet to the ground truth, the selected positions were matched with the left and right truth positions by taking the pairing with the smaller total distance from matched to truth. Then, the left/right foot accuracy is the percentage of frames where the left/right matched position is within the distance D of the corresponding truth position. For a more challenging metric, we also found the percentage of frames where *both* of the matched foot positions were within the distance D of their corresponding truth positions.

The selected foot positions could only be as accurate as the available proposals on a given frame. There can be frames where none of the proposals are within the distance D of either truth position. For this reason, we also computed accuracies using a modified truth. The left/right modified truth position was set as the proposal closest to the left/right actual truth position.

The feet were selected using spheres of various radii (Section 5.2.4). Figure 5.10 shows the accuracy of both feet versus radii. Each radius r on the horizontal axis indicates the range of radii $\{0, 1, \ldots, r\}$. The accuracy improved significantly between using a radius of 0 to using radii of 0 and 1. Only minute improvements were observed afterwards.

The remaining results were calculated using radii of $\{0, 1, \ldots 5\}$. The left/right accuracies compared to ground truth were 80 % and 79 %, respectively, and the accuracy of both was 62 %. The accuracies compared to the modified truth were significantly higher, with 98 % for left and right, and 96 % for both (Table 5.4).

Table 5.4: Accuracy of selected feet matched with truth positions.

	Left	Right	Both
Truth	0.80	0.79	0.62
Modified	0.98	0.98	0.96

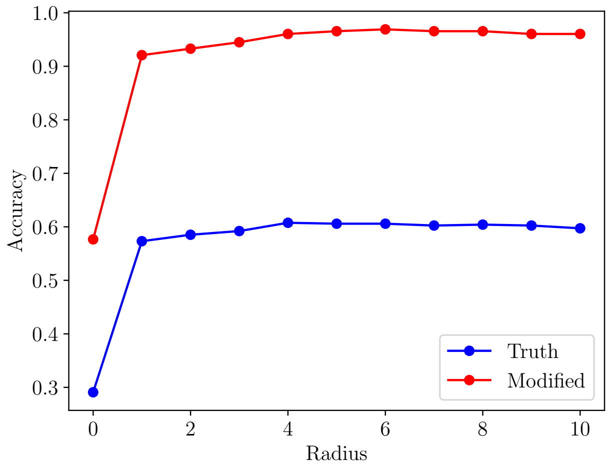


Figure 5.10: Accuracy of both selected foot positions compared to the radii used in Section 5.2.4. A radius r on the x axis indicates that the radii $\{0, 1, \ldots, r\}$ were used. The accuracy was calculated as the percentage of frames where both selected foot positions were within the distance D of the corresponding truth positions.

Side Assignment

The selected foot positions were converted to 2D before being assigned left/right labels (Section 5.2.6). The same conversion was applied to the ground truth and modified truth positions. The assigned feet were directly compared to the truth (no matching needed). The assigned feet achieved an accuracy of 76 % for both feet compared to ground truth and 96 % for both feet compared to modified truth (Table 5.5).

Table 5.5: Accuracy of feet after side assignment.

	Left	Right	Both
Truth	0.88	0.87	0.76
Modified	0.99	0.98	0.96

5.4.3 Gait Analysis

Bland-Altman

Bland-Altman analysis [36] is a common technique to quantify the agreement between two measurement devices. Given two sets of measurements X_A and X_B, the differences $X_A - X_B$ are computed. The bias of device A compared to device B is the mean of these differences. The limits of agreement are defined as the bias $\pm\, 1.96\sigma$, where σ is the standard deviation of the differences. Assuming that the differences are normally distributed, then 95 % of the differences are expected to lie between the limits of agreement [37]. Thus, a narrow range between the limits indicates a strong agreement.

In order to compare gait parameters with different magnitudes (e.g., stride length and width), relative differences were computed instead of actual differences. The relative difference between two measurements x_A and x_B was calculated as $(x_A - x_B)/\operatorname{mean}(x_A, x_B)$.

Table 5.6 displays the bias and limits of agreement of gait parameters calculated by our method when compared to the ground truth. Stride length had the lowest bias (0.3 %), followed by stance percentage (0.6 %), step length (1.4 %), stride velocity (17.5 %), and stride width (40.2 %). Furthermore, stride length and step length had the lowest range (8.0 %), followed by stance percentage (11.3 %), stride velocity (27.4 %), and stride width (146.2 %). The results are visualized in Figure 5.11.

Intraclass Correlation

Interclass correlation coefficients, such as Pearson's coefficient, quantify the correlation between variables of different classes. By contrast, intraclass correlation

Table 5.6: Bland-Altman analysis.

	Bias	Lower limit	Upper limit	Range
Stride Length	0.003	-0.037	0.043	0.080
Step Length	0.014	-0.026	0.054	0.080
Stride Width	0.402	-0.329	1.133	1.462
Stride Velocity	0.175	0.038	0.312	0.274
Stance Percentage	0.006	-0.051	0.062	0.113

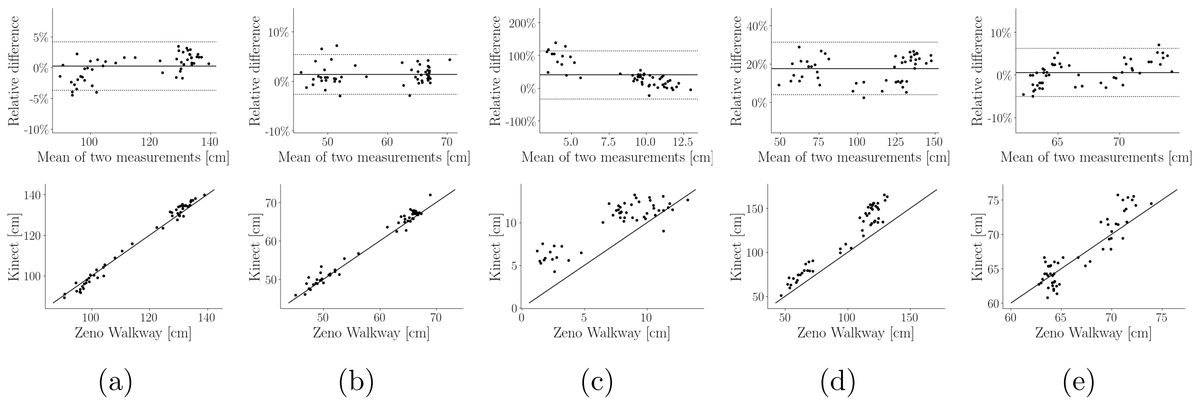


Figure 5.11: Comparison of gait parameters calculated by the Kinect and Zeno Walkway. The upper plots are Bland-Altman plots. The horizontal lines show the bias and limits of agreement. The lower plots are direct comparisons of the values. The diagonal line shows the ideal agreement. (a) Stride length. (b) Step length. (c) Stride width. (d) Stride velocity. (e) Stance percentage.

coefficients (ICCs) quantify both the correlation and agreement between variables of the same class [38]. We calculated ICCs of the form $ICC_{2,1}$ and $ICC_{3,1}$. The former quantifies the absolute agreement between raters (the Kinect and Zeno Walkway), and the latter quantifies consistency across the walking trials. The values can be interpreted as poor (<0.4), fair to good (0.4–0.74), and excellent (>0.75) [23].

The two forms of ICC are reported in Table 5.7 for the gait parameters. Stride length had the highest agreement ($ICC_{2,1} = 0.991$) followed by step length (0.985), stance percentage (0.878), stride velocity (0.802), and stride width (0.617). Stride and step length also had the highest consistencies across trials ($ICC_{3,1} = 0.991$ and

0.990, respectively). While stance percentage had a greater agreement than stride velocity, it had a lower consistency across trials ($0.881 < 0.946$). Stride width had the lowest consistency (0.841).

Table 5.7: Intraclass correlation coefficients.

	$ICC_{2,1}$	$ICC_{3,1}$
Stride Length	0.991	0.991
Step Length	0.985	0.990
Stride Width	0.617	0.841
Stride Velocity	0.802	0.946
Stance Percentage	0.878	0.881

5.5 Discussion

Our method calculates standard spatial and temporal gait parameters starting with multiple joint proposals, which are generated from side-view depth images of walking trials. We first estimate the lengths of rigid links between body parts by examining the set of frames in a walking trial. These estimated lengths are used to represent the joint proposals as a weighted graph. The shortest paths from head to foot find combinations of body parts with lengths similar to the estimated lengths. We employ a voting process to select the two shortest paths that best represent the actual two sides of the body, in turn providing the best head and feet. The feet are assigned to left/right sides using the direction of walking motion as defined by the head. By examining the motion of the feet over time, the stance phases (when the foot is contact with the floor) are detected. Gait parameters are calculated from the positions and frames of these stance phases.

While only the foot positions were used in the gait parameter equations (Section 5.3.2), the head positions were used to define the direction of walking motion

(Section 5.2.6) due to their high accuracy, which was demonstrated in our previous work [31] and again in Section 5.4.2. Future work could compare using the head as a proxy for certain gait parameters (such as stride velocity) to the standard calculations with the feet.

The method of finding shortest paths assumes that the lengths between parts remain constant over the trial. While the head to hip is not a rigid link like the calf to foot, we assume that the length does not greatly vary while walking upright.

The foot selection algorithm is designed to select a pair of paths that includes as many joint proposals as possible in the corresponding spheres, provided that the links between these proposals have positive scores. If the two feet were selected by simply choosing the two paths with the lowest total weights, the selected proposals could have both been generated from the same actual foot, while the other actual foot is ignored. Instead, the selection of two feet that are far apart results in a higher total score, provided that the links to these feet have overall positive scores.

The use of negative scores discourages the selection of a noisy foot proposal. Consider the scenario where the two correct foot proposals are close together while an incorrect proposal is far away. If the scores were restricted to positive values, the algorithm would select the noisy foot proposal as the link to the proposals would have a positive (albeit small) score, which still contributes to the total score. When the score is negative, the noisy proposal causes a net decrease in the total score.

We compared the lengths estimated over the walking trial to ground truth lengths from labelled trials and found close agreement. There was also little deviation in the lengths within trials by the same participant.

We found that few radii were needed for the foot selection algorithm to achieve a high accuracy. In fact, the addition of a single radius beyond zero caused the majority of the improvement. While fair to good accuracies were achieved for the feet compared to ground truth, the accuracies were much higher when comparing to a modified truth containing the best proposals available. The accuracy remained high after assigning left/right sides to the feet. We conclude that the bottleneck of the accuracy lies in the predictor generating joint proposals, not in the proposed method to select feet.

The calculated gait parameters were compared to ground truth parameters from the Zeno Walkway, a pressure-sensitive walkway used in clinical practice. We summarize the results by ranking the parameters by descending accuracy: stride length and step length (low bias, low variance), stance percentage (low bias, medium variance), stride velocity (medium bias, medium variance), and stride width (high bias, high variance). We hypothesize that the bias of stride velocity is mainly caused by using an inaccurate frame rate. The actual frame rate of our Kinect device may be slightly different than the theoretical rate of 30 frames per second. This would explain the low bias in the stance percentage, since the frame rate is cancelled by dividing the stance time by the stride time. The inaccuracy of stride width may be caused by the measuring error in the depth sensor, which becomes significant for small distances.

The proposed method is designed for clinical walking trials on a walkway, and relies on the consistency of these trials. For example, the feet can be projected onto the xz plane because of the consistent perpendicular view of the camera. A more general approach would be required for non-perpendicular views, such as estimating the plane that best fits the walkway.

5.6 Conclusion

We have presented a new system for clinical gait analysis with a depth sensor from a side view. The use of non-frontal depth sensors adds convenience to clinical trials and facilitates long term analysis. While researchers have previously investigated gait analysis with non-frontal depth sensors, our contribution is the direct calculation of standard gait parameters from individual foot positions.

We first select human joints from multiple proposals generated on depth images. The selected foot joints are further analyzed to detect stance phases, which are used to calculate five gait parameters (stride and step length, stride width, stride velocity, and stance percentage). The results demonstrate that accurate positions are selected from the available proposals. Using a pressure-sensitive walkway as ground truth, we find that the large spatial gait parameters (stride and step length) are the most reliable.

Possible extensions to our system include the use of other body parts to measure novel gait parameters that are inaccessible to a pressure-sensitive walkway. In order to track the upper body, our foot selection process could be applied to select the two best hand proposals, by finding the shortest paths from the head to the hands. The method could also be adapted into an online algorithm that continuously updates estimates of the body lengths and walking direction, rather than calculating them over the trial or walking pass.

In conclusion, we envision a vision-based system capable of measuring both standard and novel gait parameters from the full body, that can collect data conveniently and unobtrusively for clinical purposes.

References

[1] M. W. Whittle, "Clinical gait analysis: A review," *Human Movement Science*, vol. 15, no. 3, pp. 369–387, 1996.

[2] J. R. Gage, "Gait analysis. an essential tool in the treatment of cerebral palsy." *Clinical orthopaedics and related research*, no. 288, pp. 126–134, 1993.

[3] M. Benedetti, R. Piperno, L. Simoncini, P. Bonato, A. Tonini, and S. Giannini, "Gait abnormalities in minimally impaired multiple sclerosis patients," *Multiple Sclerosis Journal*, vol. 5, no. 5, pp. 363–368, 1999.

[4] C. L. Martin, B. Phillips, T. Kilpatrick, H. Butzkueven, N. Tubridy, E. McDonald, and M. Galea, "Gait and balance impairment in early multiple sclerosis in the absence of clinical disability," *Multiple Sclerosis Journal*, vol. 12, no. 5, pp. 620–628, 2006.

[5] U. Givon, G. Zeilig, and A. Achiron, "Gait analysis in multiple sclerosis: characterization of temporal–spatial parameters using gaitrite functional ambulation system," *Gait & posture*, vol. 29, no. 1, pp. 138–142, 2009.

[6] M. C. Kirkland, E. M. Wallack, S. N. Rancourt, and M. Ploughman, "Comparing three dual-task methods and the relationship to physical and cognitive impairment in people with multiple sclerosis and controls," *Multiple sclerosis international*, vol. 2015, 2015.

[7] D. Podsiadlo and S. Richardson, "The timed "up & go": a test of basic functional mobility for frail elderly persons," *Journal of the American geriatrics Society*, vol. 39, no. 2, pp. 142–148, 1991.

[8] R. Phan-Ba, A. Pace, P. Calay, P. Grodent, F. Douchamps, R. Hyde, C. Hotermans, V. Delvaux, I. Hansen, G. Moonen *et al.*, "Comparison of the timed 25-foot and the 100-meter walk as performance measures in multiple sclerosis," *Neurorehabilitation and neural repair*, vol. 25, no. 7, pp. 672–679, 2011.

[9] R. C. Lynall, L. A. Zukowski, P. Plummer, and J. P. Mihalik, "Reliability and validity of the protokinetics movement analysis software in measuring center of pressure during walking," *Gait & posture*, vol. 52, pp. 308–311, 2017.

[10] T. Cloete and C. Scheffer, "Benchmarking of a full-body inertial motion capture system for clinical gait analysis," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE.* IEEE, 2008, pp. 4579–4582.

[11] S. J. M. Bamberg, A. Y. Benbasat, D. M. Scarborough, D. E. Krebs, and J. A. Paradiso, "Gait analysis using a shoe-integrated wireless sensor system," *IEEE transactions on information technology in biomedicine*, vol. 12, no. 4, pp. 413–423, 2008.

[12] W. Tao, T. Liu, R. Zheng, and H. Feng, "Gait analysis using wearable sensors," *Sensors*, vol. 12, no. 2, pp. 2255–2283, 2012.

[13] A. Pfister, A. M. West, S. Bronner, and J. A. Noah, "Comparative abilities of microsoft kinect and vicon 3d motion capture for gait analysis," *Journal of medical engineering & technology*, vol. 38, no. 5, pp. 274–280, 2014.

[14] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE multimedia*, vol. 19, no. 2, pp. 4–10, 2012.

[15] M. Gabel, R. Gilad-Bachrach, E. Renshaw, and A. Schuster, "Full body gait analysis with kinect," in *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE.* IEEE, 2012, pp. 1964–1967.

[16] E. E. Stone and M. Skubic, "Unobtrusive, continuous, in-home gait measurement using the microsoft kinect," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 10, pp. 2925–2932, 2013.

[17] G. Baldewijns, G. Verheyden, B. Vanrumste, and T. Croonenborghs, "Validation of the kinect for gait analysis using the gaitrite walkway," in *Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE.* IEEE, 2014, pp. 5920–5923.

[18] F. Ahmed, P. Polash Paul, and M. L. Gavrilova, "Kinect-based gait recognition using sequences of the most relevant joint relative angles," 2015.

[19] A. A. Chaaraoui, J. R. Padilla-López, and F. Flórez-Revuelta, "Abnormal gait detection with rgb-d devices using joint motion history features," in *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, vol. 7. IEEE, 2015, pp. 1–6.

[20] X. Xu, R. W. McGorry, L.-S. Chou, J.-h. Lin, and C.-c. Chang, "Accuracy of the microsoft kinect™ for measuring gait parameters during treadmill walking," *Gait & posture*, vol. 42, no. 2, pp. 145–151, 2015.

[21] S. Motiian, P. Pergami, K. Guffey, C. A. Mancinelli, and G. Doretto, "Automated extraction and validation of children's gait parameters with the kinect," *Biomedical engineering online*, vol. 14, no. 1, p. 112, 2015.

[22] E. Cippitelli, S. Gasparrini, S. Spinsante, and E. Gambi, “Kinect as a tool for gait analysis: validation of a real-time joint extraction algorithm working in side view,” *Sensors*, vol. 15, no. 1, pp. 1417–1434, 2015.

[23] E. Dolatabadi, B. Taati, and A. Mihailidis, “Concurrent validity of the microsoft kinect for windows v2 for measuring spatiotemporal gait parameters,” *Medical engineering & physics*, vol. 38, no. 9, pp. 952–958, 2016.

[24] J. Behrens, C. Pfüller, S. Mansow-Model, K. Otte, F. Paul, and A. U. Brandt, “Using perceptive computing in multiple sclerosis-the short maximum speed walk test,” *Journal of neuroengineering and rehabilitation*, vol. 11, no. 1, p. 89, 2014.

[25] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, “Real-time human pose recognition in parts from single depth images,” in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. Ieee, 2011, pp. 1297–1304.

[26] J. Han, L. Shao, D. Xu, and J. Shotton, “Enhanced computer vision with microsoft kinect sensor: A review,” *IEEE transactions on cybernetics*, vol. 43, no. 5, pp. 1318–1334, 2013.

[27] S. Czarnuch and A. Mihailidis, “Development and evaluation of a hand tracker using depth images captured from an overhead perspective,” *Disability and Rehabilitation: Assistive Technology*, vol. 11, no. 2, pp. 150–157, 2016.

[28] D. Zhang and M. Shah, “Human pose estimation in videos,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2012–2020.

[29] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P. V. Gehler, and B. Schiele, “Deepcut: Joint subset partition and labeling for multi person

pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4929–4937.

[30] F. Huxham, J. Gong, R. Baker, M. Morris, and R. Iansek, "Defining spatial parameters for non-linear walking," *Gait & posture*, vol. 23, no. 2, pp. 159–163, 2006.

[31] A. Hynes and S. Czarnuch, "Combinatorial optimization for human body tracking," in *International Symposium on Visual Computing.* Springer, 2016, pp. 524–533.

[32] E. W. Weisstein, "Complete bipartite graph," 2002.

[33] T. H. Cormen, *Introduction to algorithms.* MIT press, 2009.

[34] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[35] J. Shlens, "A tutorial on principal component analysis," *arXiv preprint arXiv:1404.1100*, 2014.

[36] J. M. Bland and D. Altman, "Statistical methods for assessing agreement between two methods of clinical measurement," *The lancet*, vol. 327, no. 8476, pp. 307–310, 1986.

[37] D. Giavarina, "Understanding bland altman analysis," *Biochemia medica: Biochemia medica*, vol. 25, no. 2, pp. 141–151, 2015.

[38] T. K. Koo and M. Y. Li, "A guideline of selecting and reporting intraclass correlation coefficients for reliability research," *Journal of chiropractic medicine*, vol. 15, no. 2, pp. 155–163, 2016.

Chapter 6

Conclusion

6.1 Summary

As stated in Section 1.2, this thesis makes four main contributions:

1. *A novel graph representation of a human depth image that improves the supervised segmentation of body parts.*

   This is described in Chapter 2.

2. *A method of selecting accurate human joints from multiple proposals.*

   The method is introduced in Chapter 3 and finalized in Chapter 5.

3. *A method to calculate standard gait parameters from individual body parts with a side-view depth sensor.*

   The method is introduced in Chapter 4 and finalized in Chapter 5.

4. *Concurrent validation of our gait analysis compared to the Zeno Walkway.*

   Chapter 4 presents preliminary results on 8 walking trials, and Chapter 5 presents results on the full dataset of 52 trials.

The objective of Chapter 2 is to improve the supervised segmentation of human body parts in a depth image. While there are several established algorithms for supervised image segmentation, the task becomes more difficult when there is self-occlusion (i.e., some body parts occluding others). Standard algorithms represent the image as a graph with vertices for pixels and edges between adjacent pixels. An occlusive limb can cause a boundary that is represented as large weights in the graph. This means that more seed pixels are needed for proper segmentation, rather than one seed per body part. Our solution is to create separate layers of the graph for the arms, which are connected to the base graph only near the shoulder. Then, a standard segmentation algorithm is run on this layered graph. We demonstrate that our approach achieves better results when compared to a normal graph representation.

In Chapter 3, we track a walking human from depth images. We propose a method to select accurate human joints from the multiple proposals generated by a trained predictor. Joint proposals are represented as two graphs – one for left parts and one for right. Edges are connected between consecutive pairs of parts from head to foot. The shortest path is selected based on kinematic predictions from previous frames. Our method achieves a better accuracy than what is obtained by selecting proposals with the highest predictor confidence.

Chapter 4 builds upon the tracking in Chapter 3 to perform gait analysis. Four gait parameters are calculated: stride length, step length, stride width, and stride velocity. The relevant foot positions are found by detecting peaks in the foot-to-foot distance signal. Gait parameters are calculated without regard for left and right sides. Walking trials were captured concurrently by a Zeno Walkway and a Kinect. It was assumed that each trial contained four walking passes, so the passes were identified by clustering frames with k-means, where $k = 4$. Our gait parameters were compared to the ground truth from the Zeno Walkway. Stride velocity had the best agreement between the two devices. While the stride and step length had a high correlation, they also had a negative bias compared to the ground truth. The stride width had the lowest agreement.

Chapter 5 improves upon the methods of both Chapter 3 and Chapter 4. In this implementation, the initial left/right distinction between joint proposals is disregarded (removing the possibility of only left proposals being available for an actual right joint, and vice versa). The lengths of the body are estimated by measuring lengths between proposals across the whole walking trial, and by making assumptions on the expected distribution of these lengths. Afterwards, the best head and foot positions are selected on each frame without the use of previous frames.

The selected head positions in a walking pass define a general direction of motion for the pass, which is used to assign left/right sides to the feet. The assumption that each walking trial contains four passes is no longer used, as some walking trials were found to contain a different number of passes. Therefore, the passes are clustered with DBSCAN, which determines the number of clusters automatically, rather than k-means from Chapter 3.

The motions of the selected feet are analyzed to identify stance and swing phases. Stance percentage is calculated along with the four parameters from Chapter 4, and the parameters are now specific to a left and right side (e.g., right stride length). Stride and step length were found to have the strongest agreement with the ground truth. While not mentioned in the paper, the cause of the negative bias in Chapter 4 was identified as a camera calibration issue.

The Spearman correlation from Chapter 4 was replaced by ICCs in Chapter 5. While Spearman is only a measure of correlation, the ICCs have the added benefit of quantifying both correlation and agreement between measuring devices, and reliability across trials.

6.2 Publications

The following is a list of publications produced during this degree:

- Conference papers

- A. Hynes and S. Czarnuch, “Combinatorial optimization for human body tracking,” in Lecture Notes in Computer Science, Proceedings of the International Symposium on Visual Computing, Las Vegas, NV, 2016.
- A. Hynes and S. Czarnuch, “Building a feature vector for assessing the gait of persons with multiple sclerosis,” presented at the Newfoundland Electrical and Computer Engineering Conference, IEEE, Newfoundland and Labrador Section, St. John’s, NL, 2016.
- A. Hynes, M. C. Kirkland, M. Ploughman, and S. Czarnuch, “Comparing the gait analysis of a Kinect system to the Zeno Walkway: Preliminary results,” presented at the Newfoundland Electrical and Computer Engineering Conference, IEEE, Newfoundland and Labrador Section, St. John’s, NL, 2017.

- Conference posters
  - A. Hynes and S. Czarnuch, “Assessing the gait of people with multiple sclerosis using 3D motion tracking: toward objective outcome measures,” in Americas Committee for Treatment and Research in Multiple Sclerosis (ACTRIMS), Orlando, FL, 2017.
  - S. Czarnuch, A. Hynes, and H. Crichton, “The state of the art of technologies for the assessment of gait in multiple sclerosis: Filling the gap,” presented at the American Academy of Neurology (AAN) Annual Meeting, Boston, MA, 2017.
- Journal articles
  - A. Hynes and S. Czarnuch, “Human part segmentation in depth images with annotated part positions,” Sensors, vol. 18, p. 15, 2018.

– A. Hynes, S. Czarnuch, M. C. Kirkland, and M. Ploughman, "Gait analysis with a side-view depth sensor via optimal selection of human joint proposals," IEEE Transactions on Biomedical Engineering, 2019 (to be submitted).

6.3 Future Works

Chapter 2 presents a method for labelling human body parts using only one seed pixel per part. However, because there were already two walking trials that were fully labelled, this labelling method was not used on the walking trials. Future work should investigate the use of our segmentation system on side-view depth images of the full human body. Then, more training images can be quickly obtained.

The overall pipeline could include these steps:

1. Capture raw depth images of a walking person
2. Train a predictor on images labelled by our supervised segmentation technique
3. Run the predictor on new depth images to generate joint proposals
4. Employ the system presented in Chapter 5 to select accurate joints and calculate gait parameters

By capturing depth images from multiple views, the predictor could be trained to generate joint proposals from any view of the human body. Our gait analysis system can also be improved to function from any viewpoint. Furthermore, additional gait parameters can be calculated from the full human body, such as arm velocity and angles between limbs. A future system could calculate accurate gait parameters directly from individual joints measured with an in-home, environmentally-mounted depth sensor.