8-2021

Motion and emotion estimation for robotic autism intervention.

Jacob M Berdichevsky
University of Louisville

MOTION AND EMOTION ESTIMATION FOR ROBOTIC AUTISM INTERVENTION

By

Jacob M. Berdichevsky

B.S. University of Louisville, 2020

A Thesis

submitted to the Faculty of the

J. B. Speed School of Engineering of the University of Louisville

in Partial Fulfilment of the Requirements

for the Degree of

Master of Science in Electrical Engineering

Department of Electrical & Computer Engineering

University of Louisville

Louisville, Kentucky

August 2021

MOTION AND EMOTION ESTIMATION
FOR ROBOTIC AUTISM INTERVENTION

By

Jacob M. Berdichevsky

B.S. University of Louisville, 2020

A Thesis Approved on

Aug 4, 2021

by the following Thesis Committee:

_____

Dr. Dan O. Popa

_____

Dr. Karla C. Welch

_____

Dr. Michael L. McIntyre

_____

Dr. Nazita Taghavi

_____

Dr. Sumit K. Das

DEDICATION

This thesis is dedicated to my parents (Dmitry and Emma), siblings (Ben and Rachel), and grandparents (Valentina and Arkadiy Gurevich, and Anatoly Berdichevsky) who have taught me the meaning of hard work, dedication, and laughter.

ACKNOWLEDGMENTS

I would like to thank Dr. Dan O. Popa for his guidance and teachings that enabled me to complete this work. Without his belief in me and his willingness to help push me forward, I would not have completed this degree. I would also like to thank Dr. Sumit K. Das and Dr. Nazita Taghavi for their help and advice through this process. I would also like to thank the other committee members, Dr. Karla C. Welch and Dr. Michael L. McIntyre, for their comments and for serving on my committee.

Of course, I must thank Carl D. Yoder, Chris P. Tran, Lauren E. Cline, Rohit A. Narayanan, and Sai Vanaparthy. Without their hard work, this thesis would not have been possible to complete.

My thanks also go out to all my friends at the Louisville Automation & Robotics Research Institute. Without all your support and help, I would not be the same person that I am today. I am grateful that I was able to meet every one of you and learn about you all—I hope we will stay in contact throughout the years.

I would like to thank the National Science Foundation (NSF) for their support through the following projects: NSF #1838808 SCH:INT Adaptive Partnership for Robotic Treatment of Autism [4] and NSF #1849213 Kentucky Advanced Manufacturing Partnership for Enhanced Robotics and Structures [4].

Finally, I would like to thank my family for encouraging me and pushing me to better myself every day. Without them, my life would be much different—this thesis is dedicated to them with my entire heart.

ABSTRACT

MOTION AND EMOTION ESTIMATION

FOR ROBOTIC AUTISM INTERVENTION

Jacob M. Berdichevsky

August 4, 2021

Robots have recently emerged as a novel approach to treating autism spectrum disorder (ASD). A robot can be programmed to interact with children with ASD in order to reinforce positive social skills in a non-threatening environment. In prior work, robots were employed in interaction sessions with ASD children, but their sensory and learning abilities were limited, while a human therapist was heavily involved in "puppeteering" the robot.

The objective of this work is to create the next-generation autism robot that includes several new interactive and decision-making capabilities that are not found in prior technology. Two of the main features that this robot would need to have is the ability to quantitatively estimate the patient's motion performance and to correctly classify their emotions. This would allow for the potential diagnosis of autism and the ability to help autistic patients practice their skills. Therefore, in this thesis, we engineered components for a human-robot interaction system and confirmed them in

experiments with the robots Baxter and Zeno, the sensors Empatica E4 and Kinect, and, finally, the open-source pose estimation software OpenPose

The Empatica E4 wristband is a wearable device that collects physiological measurements in real time from a test subject. Measurements were collected from ASD patients during human-robot interaction activities. Using this data and labels of attentiveness from a trained coder, a classifier was developed that provides a prediction of the patient's level of engagement. The classifier outputs this prediction to a robot or supervising adult, allowing for decisions during intervention activities to keep the attention of the patient with autism.

The CMU Perceptual Computing Lab's OpenPose software package enables body, face, and hand tracking using an RGB camera (e.g., web camera) or an RGB-D camera (e.g., Microsoft Kinect). Integrating OpenPose with a robot allows the robot to collect information on user motion intent and perform motion imitation. In this work, we developed such a teleoperation interface with the Baxter robot.

Finally, a novel algorithm, called Segment-based Online Dynamic Time Warping (SoDTW), and metric are proposed to help in the diagnosis of ASD. Social Robot Zeno, a childlike robot developed by Hanson Robotics, was used to test this algorithm and metric. Using the proposed algorithm, it is possible to classify a subject's motion into different speeds or to use the resulting SoDTW score to evaluate the subject's abilities.

TABLE OF CONTENTS

ACKNOWLEDGMENTS iv
ABSTRACT vi
LIST OF TABLES x
LIST OF FIGURES xi
CHAPTER 1 INTRODUCTION 1
1.1 Motivation 1
1.2 Thesis Contributions 5
1.3 Thesis Organization 8
CHAPTER 2 BACKGROUND 9
2.1 Robots for Autism 10
2.2 Human-Robot Interaction 13
2.3 Physiological Sensors and Emotion Classification 16
2.4 Motion Quality Studies 20
CHAPTER 3 EMPATICA E4 AND ITS APPLICATIONS 23
3.1 Gaining Access to the Empatica E4 Data 24
3.1.1 System 25
3.1.1.1 Data Collection 25
3.1.1.2 Data Processing 30
3.1.1.3 Data Visualization 32
3.2 Controlling Zeno's Arm with the Empatica E4 34
3.3 Classifying Attentiveness 35
3.3.1 Gathering and Labeling Data 36
3.3.2 Feature Extraction 37
3.3.3 Deep Learning 38
3.3.4 Models Experimentation 39
3.3.5 Results 43
3.3.6 Future Work 47

CHAPTER 4 BAXTER TELEOPERATION USING OPENPOSE 49
4.1 System Architecture 50
4.2 Description of Imaging Hardware and Software 53
4.3 Kinematic Model of Baxter's Right Arm 55
4.3.1 URDF Models 57
4.4 Inverse Kinematics 58
4.5 Finding Joint Offsets 63
4.6 Controller and Gain Tuning 64
4.7 Results 66
4.7.1 Simulated Results 66
4.7.2 Experimental Results 69
CHAPTER 5 SEGMENT-BASED ONLINE DYNAMIC TIME WARPING 79
5.1 Segment-based Online Dynamic Time-Warping Algorithm 80
5.2 Validation with Human Subjects 83
5.2.1 Experimental Setup 83
5.2.2 Offline Validation 85
5.2.3 Online Validation 87
5.3 Adaptive Imitation 87
CHAPTER 6 CONCLUSION AND FUTURE WORK 89
6.1 Conclusion 89
6.2 Future Work 91
REFERENCES 93
CURRICULUM VITA 99

LIST OF TABLES

Table 1. Empatica Streaming Server Codes to Sensor 27
Table 2. Layer Size Experiments 41
Table 3. Number of Epochs Experiment 41
Table 4. Loss Function Experiments 42
Table 5. Testing Optimizers 42
Table 6. Activation Functions 43
Table 7. DH Table for Baxter 55
Table 8 SODTW Parameter Initialization 85

LIST OF FIGURES

Figure 1. Diagram of Smart and Connected Health (SCH) Project #1838808 from the US National Science Foundation (NSF) 3
Figure 2. Robota robot, left is the obscured robot, middle shows the robotic parts, and right is the dressed robot [10] 11
Figure 3. KASPAR robot [11] 12
Figure 4. Experimental setup of the study on ZECA and the ASD children [12] 12
Figure 5. Milo robot [14] 13
Figure 6. Bandit [15] 14
Figure 7. Facial expression from a user and android Phillip K. Dick [17]. 16
Figure 8. Shimmer 3 sensors [18] 17
Figure 9. Sensor layout from AlZoubi et al [19] 18
Figure 10. The EmotiGo [20] 19
Figure 11. Wearable designed by Jiang et al. [21] 20
Figure 12. Samples from Video taken during data collection [22] 21
Figure 13. Stroke Rehabilitation system with two subjects [24] 22
Figure 14. Sample E4 Data 24
Figure 15. Data Flow between the Empatica E4 wearable and the Zeno Robot in our lab 25
Figure 16. Connect_2_E4.vi 26
Figure 17. SendData2E4.vi 27
Figure 18. MainLoop.vi 28
Figure 19. Get_e4_Data_Over_Period.vi 29
Figure 20. Wireshark Snippet 30
Figure 21. ParsingString.vi 30
Figure 22. GetArrays.vi 32
Figure 23. ExtractFeatures.vi 33
Figure 24. An Example of Real Time Data from E4 34
Figure 25. e4_Zeno_Move.vi's binary switch 35
Figure 26 Nao Robot 36
Figure 27. Number of Datapoints per Subject 38
Figure 28. Shallow Network 40
Figure 29. Two Layer Network 40
Figure 30. Three Layer Network 40
Figure 31. Five Layer Network 41
Figure 32. Layer Size Experiment Results 44
Figure 33. Epoch Number Experiment Results 45

Figure 34. Loss Function Experiment Results 45
Figure 35. Optimizer Experiment Results 46
Figure 36. Activation Layers Experiment Results 47
Figure 37. Baxter Teleoperation System Architecture 50
Figure 38. Joint Description [35] 53
Figure 39. Kinematic model generated using Peter Corke's Robotics toolbox 56
Figure 40. Deriving the DH Parameters 56
Figure 41. RVIZ Kinematic Arm models (Zeno left, Baxter Right) 57
Figure 42. Zeno Kinematic Angles [49] 58
Figure 43. Origin and Axes of the OpenPose 59
Figure 44. Azure Kinect vs Subject Location. Also depicted in our lab from left to right are robots PKD [17], Milo [13], Zeno [23][50], and Baxter 59
Figure 45. The Azimuth (φ) and Elevation angles (ψ) 60
Figure 46. Normal to the plane formed by $Vse \times Veh$ 62
Figure 47. Baxter's Joint Position Controller [52] 64
Figure 48. Baxter's Raw Joint Position Controller [53] 65
Figure 49. X Axis Actual and Error 67
Figure 50. Y Axis Actual and Error 68
Figure 51. Z Axis Actual and Error 68
Figure 52. Left and Right S0 Joint Angles (OpenPose desired vs Baxter Position) for Joint Position Controller 70
Figure 53. Baxter Untucked Position 99[55] 70
Figure 54. Left and Right S0 Joint Angles (OpenPose desired vs Baxter Position) for Raw Position Controller 71
Figure 55. Left and Right S1 Joint Angles (OpenPose desired vs Baxter Position) 72
Figure 56. Left and Right S1 Joint Angles (OpenPose desired vs Baxter Position) for Raw Position Controller 73
Figure 57. Left and Right E0 Joint Angles (OpenPose desired vs Baxter Position) 74
Figure 58. Left and Right E0 Joint Angles (OpenPose desired vs Baxter Position) for Raw Position Controller 74
Figure 59. Left and Right E1 Joint Angles (OpenPose desired vs Baxter Position) 75
Figure 60. Left and Right E1 Joint Angles (OpenPose desired vs Baxter Position) for Raw Position Controller 76
Figure 61. Left and Right Wrist Joint Angles (Openpose desired vs Baxter Position) 77
Figure 62. Left and Right Wrist Joint Angles (OpenPose desired vs Baxter Position) for Raw Position Controller 78
Figure 63. SODTW applied to a Cyclic Sequence 81
Figure 64. Experimental Setup 83
Figure 65. Sample Output of the Kinesthesia Toolkit 84
Figure 66. SODTW cost for 55 subjects with no weight 86
Figure 67. Sequence from a subject performing the motion with (right) and without (left) a weight 86
Figure 68. The performance of all 13 subjects during the Adaptation Experiment 88

CHAPTER 1

INTRODUCTION

1.1 Motivation

Autism spectrum disorder (ASD) affects approximately 1 in 54 US children from all racial, ethnic, and socioeconomic groups, as reported by the Centers for Disease Control and Prevention (CDC) in 2016 [1]. Individuals with ASD have difficulties communicating and interacting with others, restricted interests, and repetitive behaviors and have symptoms that impede their ability to function socially, as reported by the National Institute of Mental Health [2].

The diagnosis of this disorder is subjective, due to the difficulty of quantifying such things as cognitive level and language skills. The Autism Diagnostic Observation Schedule (ADOS) is a “semi-structured” assessment used to provide a standardized way to diagnose ASD [3]. The assessment consists of topics such as social behavior, and use of speech and gestures in social situations, among other topics [3]. Lord et al. suggest that the ADOS should be administered by professionals who have significant experience with ASD patients and the clinical assessment of the disorder [3]. Therefore, to diagnose a subject with autism is no small task.

Treating individuals affected with ASD can be complicated due to their restricted interests and impaired language abilities. According to the CDC, some of the typical

treatments include Discrete Trial Training, where a lesson is split into simple parts and positive reinforcement is used to reward correct answers, Social Skills Training, which teaches the children the skills necessary to interact with others, or Speech Therapy, which seeks to improve verbal communication skills [6].

The recently awarded National Science Foundation project, SCH:INT Adaptive Partnership for Robotic Treatment of Autism [4], aims to tackle some of these challenges in diagnosis and therapy to help individuals with this disorder using robotic technology. As depicted in Figure 1, the project has several goals, such as creating a quantitative severity scale for ASD, creating a robot that can adapt to the human in therapy sessions, and, finally, conducting human-robot interaction (HRI) studies between robots and children with ASD in a group therapy setting. To help reach the project goals, this thesis, focuses on creating subsystems that will enable a robot to be adaptive in a therapy scenario by interpreting physiological signals and motion data from ASD subjects. In addition, our work also proposed and validated an algorithm that can be used to identify the motion quality of a subject during imitation of the robot. This thesis contributes to research contributions one, two and four from Figure 1.

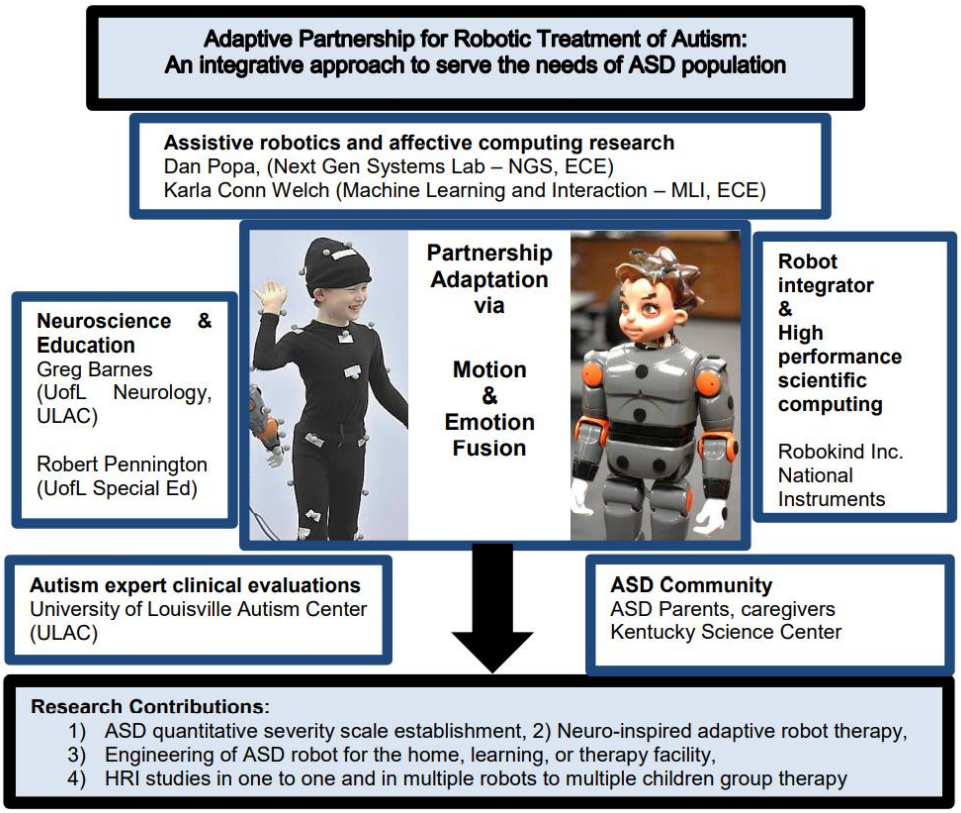

Figure 1. Diagram of Smart and Connected Health (SCH) Project #1838808 from the US National Science Foundation (NSF)

Research contribution one of the SCH project involves the creation of an ASD scale based on the assessment of motion performance and emotional engagement of a subject. There are several challenges that need to be considered when developing such a scale. Prior to being able to understand the motion performance of ASD subjects, it is necessary to acquire and stream sensor data and implement an on-line algorithm to quantify the performed motion. The algorithm must be thoroughly tested to validate that it can accurately assess the motion quality of a subject. Likewise, classifying the emotional engagement of a subject is no easy task. It is necessary to somehow capture the physiological and/or psychological signals of a subject, then classify them into different

emotions. This involves creating a dataset of physiological signals and labeling them appropriately using deep learning classification algorithms.

In a similar vein, research contribution two focuses on using a robot that can adapt during a therapy session. This necessitates that a robot can quantify and understand the needs of the user to maximize the effectiveness of the therapy. Previous work in this field necessitated heavy involvement of a human to essentially puppeteer (or teleoperate) the robot, such as the FACE, AuRoRa, Kaspar, and Keepon robotic systems [7]. By using the same algorithms created in research contribution 1, it is possible to enable a robot to adapt to the motion quality and emotional engagement of a subject. This will enable the creation of a "closed-loop" system for ASD intervention.

Creating a "closed-loop" system for ASD intervention allows the robot to compensate for users' behavioral variations in each instance of interaction. In the eyes of a robot, humans can be difficult to interpret and even behave illogically, which is the basis for the field of research in human-robot interaction. By creating a system that can understand the motions and emotions of a subject, we are able to strive for a more autonomous approach to therapy and diagnosis. This behavior enables a therapist to be solely focused on the subject rather than a robot, generates more natural responses from the robot (by reducing delay created by puppeteering), and can also help foster a peer-like relationship between the subject and the robot. Interpreting the emotional states of a subject can enhance the abilities of a robot to ensure the maximum ability of the therapy session. Using this information, the robot can be programmed to respond to certain emotional states on the fly. For instance, if inattentiveness is detected, the robot can respond by prompting the subject to pay attention. Or if the subject becomes frustrated,

the robot could alert a therapist while also beginning to follow a “calming” routine. These are only a few ways that enhancing a robot with the emotional state of a subject can beneficial.

Emotion classification is not the only function of an autism robot that can be improved to help ASD patients. A second task is to enable a robot to understand the motion quality of the subject. In past works, it has been found that motion quality can be indicative of ASD [8]. By enabling a robot to quantify the proficiency of a subject during a motion, it could be possible to diagnose a subject before they can even speak. Motion-quality assessment can also be used to quantify the improvement of a subject during a training session. This can be done by viewing the subject’s motion-quality score over time as they are trained during a therapy session. Therefore, motion-quality estimation would enable the next-generation autism robot to quantifiably track the improvement of a subject.

1.2 Thesis Contributions

The contribution of this research is to create algorithms and subsystems that can be used for motion and emotion estimation to enhance the next-generation autism robot. In particular, we utilized a wearable wristband (the Empatica E4) that collects physiological measurements in real time from a test subject. Measurements were collected from ASD patients during a social skills camp at the University of Louisville Autism Center (ULAC). Using this data, we explored different neural network structures and hyperparameters to see if an effective classifier can be developed for identifying a subject’s attentiveness. The classifier enables the performance of intervention activities by a robot or supervising adult to keep attention of the autism patient.

The Empatica E4 wristband is a wearable device that collects physiological measurements in real time from a test subject. It is equipped with a PPG sensor, GSR sensor, three axis accelerometer, infrared thermopile, event mark button, and an internal real time clock. This allows the researcher to measure blood volume pulse, heart rate, electrical properties of the skin, and skin temperature. In addition, the clock and event mark button allow researchers to save snapshots of important events (such as when a subject has begun or completed a certain exercise).

The work with the physiological wristband tackles the problem of creating an algorithm for the emotional understanding of a subject. By conducting a study, we are creating a dataset that can be used for classification of the subject's affective state. In this dataset, we explored the attentiveness of two subjects and the ability to classify it using neural networks. Some of the models developed as part of this work were able to classify the attentiveness of a subject at 68% accuracy. In the future, the experimental knowledge can be used to generate new studies and classifiers to capture other emotions. This will enable the next-generation autism robot to become more sensitive to the needs of the subject.

Using the CMU Perceptual Computing Lab's OpenPose software package, which enables body, face, and hand tracking using a camera (such as a Microsoft Azure Kinect), we were able to teleoperate a robot so that it could imitate the upper body motion of a user. This enhances our previous work in motion quality estimation with more data about the subject and provides a way to record and generate trajectories from a human subject who is performing a motion.

Skeletal tracking will be a necessary function for the next-generation autism robot to examine a subject's motion quality. Combining OpenPose's tracking abilities with a robot enables us to record and analyze more complicated motions involving features such as fingers and facial points for use in a motion quality study or gaze tracking.

The Baxter robot developed by Rethink Robotics is a collaborative robot that is meant to accomplish assembly line tasks. This robot features two arms with seven degrees of freedom (DOF) each, an ultrasonic sensor grid, three cameras (one on each arm near the end-effector, and one on the main display), accelerometers, and sensor pads for robot manipulation by a human. The pads enable a human to teach Baxter by showing examples of a task/motion. This robot was used to implement a teleoperation experiment with OpenPose.

Finally, we proposed a novel algorithm (called SODTW - Segment-based Online Dynamic Time Warping) to help in the diagnosis of ASD by quantifying the motion-imitation performance of human subjects. Social Robot Zeno, a childlike robot developed by Hanson Robotics, was used to implement and experimentally test this algorithm. Using the proposed algorithm, it is possible to adapt a robot to the subject's motion. This algorithm also enables the evaluation of motion quality to see how the subject is performing.

To test this algorithm, we used Social Robot Zeno and the Open Kinesthesia toolkit developed for LabVIEW by the University of Leeds, UK. Zeno features an emotive face with seven facial muscle motors, two eye motors, two neck motors, and a blink motor. It is powered by a LabVIEW NI myRio, therefore, all programming will be done in LabVIEW. The Open Kinesthesia toolkit is a software add-on that enables the use of a

Microsoft Kinect for Windows sensor. The toolkit allows for hand, feet, and basic head tracking, however, it doesn't provide tracking of fingers and facial features.

By using a study, Zeno was used to test whether it was possible to adapt to three different speeds of the subject and evaluate their motion quality. During the motion-quality evaluation, the subjects would mimic the movement of the robot with or without a 15lb kettlebell. This provided an impairment model that showed the SODTW algorithm can differentiate between an impaired motion and unimpaired motion. In addition, Zeno was able to identify the closest speed the subject was performing the motion (fast, normal, or slow), and adapt to their speed.

1.3 Thesis Organization

This thesis is organized in the following way:

Chapter 2 is a background survey of the topics: robots for autism, human-robot interaction, physiological sensors and emotion classification, and, finally, motion-quality estimation. Chapter 3 describes our experimentation in creating a neural network for classifying data from a wearable that collects physiological signals such as blood volume pulse (BVP), galvanic skin response (GSR), and temperature data. Chapter 4 introduces our work in teleoperating a robot using pose estimation software and inverse kinematics. Chapter 5 discusses our proposed algorithm, called Segment-based Online Dynamic Time Warping and the results of a study conducted on healthy subjects that are above the age of 18. Finally, Chapter 6 concludes by summarizing the thesis and discusses the future work that can be used to enhance the techniques described above.

CHAPTER 2

BACKGROUND

The use of robotics with autism is a field that is expanding in order to help the individuals affected by this disorder. A robot built for this task must be able to take in different sensor data and adapt to the subject. In this chapter, we explored scientific literature related to the topics of robots for autism, human-robot interaction (HRI), the use of physiological sensors in emotion classification, and motion-quality studies.

There are several examples of robots for autism, as well as studies that are used to analyze their therapeutic effects. In general, it has been found that individuals with autism are fascinated by and prefer them to humans. These robots are programmed to help subjects with social and communication skills or movement. By ensuring the subjects are engaged it is possible to maximize the effectiveness of therapy.

Several robots, such as Nao, Zeca, and Kaspar show promising results in helping the children with ASD in becoming more engaged. By keeping these subjects engaged, it is possible for them to receive the full benefit of therapy, which is an example of human-robot interaction (HRI). HRI is a field of study which includes analyzing how individuals feel about, interact with, and safely cooperate with robots. Reviewing the literature enables us to create novel ways to interact with the robot while designing our experiments to ensure that the subjects feel safe.

Analyzing the emotion of a subject ensures that it is possible for investigators and robots to understand the affective state of the subject. However, to understand this, it is necessary to somehow collect data for analysis. In the literature, it has been reported that physiological signals can be used to classify the affective state of a subject. By reviewing the types of sensors and experiment setups that are used in the literature, we can create a classifier would enable a robot to understand the emotions of its user. This information will allow "closed-loop" control of the robot, enabling on-the-fly decision-making by the robot.

It has been observed by Todorova et al. that children with ASD struggle with mimicking specific motions [9]. A motion, such as a hand wave, that would be relatively for a neurotypical child can be difficult for ASD patients to perform. Therefore, motion quality is also important to understand the capabilities of a subject. By reviewing the literature for algorithms to classify motion quality, it is possible to create algorithms for grading a subject's capabilities.

2.1 Robots for Autism

Several research groups have designed and studied robots meant for helping children with autism. These include such robots such as Softbank Robotics' Nao robot, Zeca, Kaspar, and many others. Two studies conducted by Robins et al. revealed that ASD subjects tend to prefer robots with simpler features over humans [1]. In the first study, a mime artist was dressed in either a robot costume (that obscured a face) or ordinary clothes. The mime artist was asked to act like a robot, perform the same movements, and avoid interaction with the children. It was observed that the ASD subject

interacted with the mime longer when wearing the robot costume than the normal clothes costume.

In the second study conducted by Robins et al., the children interacted with a Robota Robot in two different modes of dress. The robot would either be puppeteered by an investigator or dance to music. The trial would last as long as the children were comfortable, and the study showed that, over a long period of time, the children interacted better with the obscured robot [1]. These studies conducted by Robins et al. show that ASD subjects are more likely to interact with robots than humans and served as the inspiration for the robot KASPAR.

Figure 2. Robota robot, left is the obscured robot, middle shows the robotic parts, and right is the dressed robot [1]

Building upon the work performed on the Robota platform, a new robot called KASPAR was developed. Robins et al. introduced KASPAR to three children with ASD (ranging in age from 6 to 16) to gauge the effectiveness of the design of the robot. These children all were diagnosed with severe autism, would avoid looking at the faces of others, and had difficulties understanding social interaction. By the end of the therapy session with KASPAR, all three subjects were able to gaze at others who were present at

the same time [11]. Their results show that KASPAR can be used to facilitate rehabilitation in autistic subjects.

Figure 3. KASPAR robot [11]

In Silva et al., the ZECA robot (or, Zeno Engaging Children with Autism), which was based on the Zeno R50 Robokind platform, was used to help children with autism to learn how to produce various facial expressions. The researchers asked ASD children to mimic preprogrammed facial expression of the Zeca robot. The purpose of this study was meant to serve as a way to help the children learn how to create those facial expressions. The researchers found that, over time, the children "had a positive evolution over the sessions" with the subjects maintaining their interest during the activity [12].

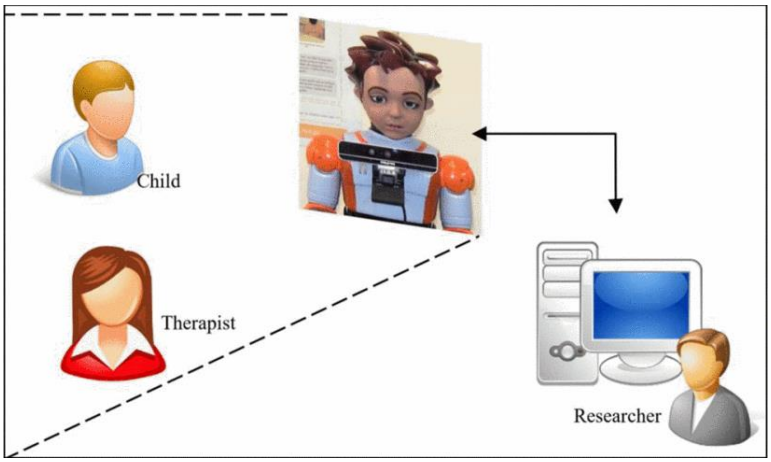

Figure 4. Experimental setup of the study on ZECA and the ASD children [12]

Robokind also produces a Milo robot (Figure 5), a production unit targeted towards getting ASD individuals access to robots. The robot is designed to enable subjects to practice communication and social skills through a preprogrammed curriculum [13]. Kroiss et al. investigate the engagement of children (between the ages of 5 and 14) with ASD when engaging with Milo versus engaging with the therapist. The subjects were either verbal or minimally verbal, and it was found that the children with ASD were more engaged with Milo than the therapist. In addition, once the minimally verbal were able to become comfortable with the robot, their performance was close to that of the fluent individuals [14].

Figure 5. Milo robot [14]

2.2 Human-Robot Interaction

The experiences of autistic children with robots falls into the domain of human-robot interaction (HRI). This field of study investigates how humans feel, control, cooperate, and interact with their robotic partner. In Fasola et al., a robot was used to analyze how subjects interact with and react to a robot during an HRI session.

The study used the Bandit robot, which features 19 degrees of freedom (with each arm having six DOF) and a USB camera. In the first study, the robot acts as an exercise coach for the elderly subjects. The subjects each participate in two sessions (10 minutes each), in one of which, the robot is a coach that gives the user praise or reassurances when appropriate, and, in the second, the robot only provides instructional feedback, such as user score and redemonstrating the exercise. After each session, the subject was given a survey to see which of the two robot modes they preferred. The researchers found that, on average, the subjects preferred the robot that would give praise or reassurances [15].

Figure 6. Bandit [15]

In a second study, the researchers evaluated whether giving the user a choice of which game to play with the robot would affect the associated score of the activity. The researchers recruited 24 participants between the ages of 68 and 89. The subjects were split into two groups, altering whether the robot would prompt the subject for the game to

play first, or not. Each subject participated in two sessions, where the second session would be the opposite of the initial choice. Once the activity was completed the subjects were given a survey to rank which session they preferred, their perception of the robot's helpfulness and intelligence, and their mood. It was found that there was no clear preference to either choice condition, however, the robot was rated highly by the subjects in the perception questions [15].

Other studies investigate the method of control for robots, such as brain-computer interfaces or physiological signals. In Bian et al., the researchers investigate the use of electroencephalogram (EEG), electromyography (EMG), and voice-command to control a robot for use in space. Ten subjects participated in a study where they would use their voice to confirm the start of the study by saying a phrase. The system would use the signals from the EEG to decide which task the subject wanted to be performed. If the system began the correct task, then the subject would utter the phrase "confirm," otherwise, the system would choose a different task. Once the task was confirmed, the EMG data would be used to manipulate virtual dials and controls such as moving a joystick *n* amount. The researchers concluded that their system was 86.7% to 97.6% accurate in choosing a control [16]. The research conducted by Bian et al. shows how different control methods can be used to control a robot.

Finally, human-robot interaction also touches upon how to ensure that a user is not unnerved by a robot. In Habib et al., the android Philip K Dick is used to understand how to generate natural facial features [17]. By utilizing a neural network and genetic algorithm, the researchers were able to take recorded human expressions and transfer this to the android. The software Faceshift ® was used to gather facial features of the human

and android. These two datasets for a given expression were compared and the genetic algorithm would explore ways to minimize the difference in expression [17].

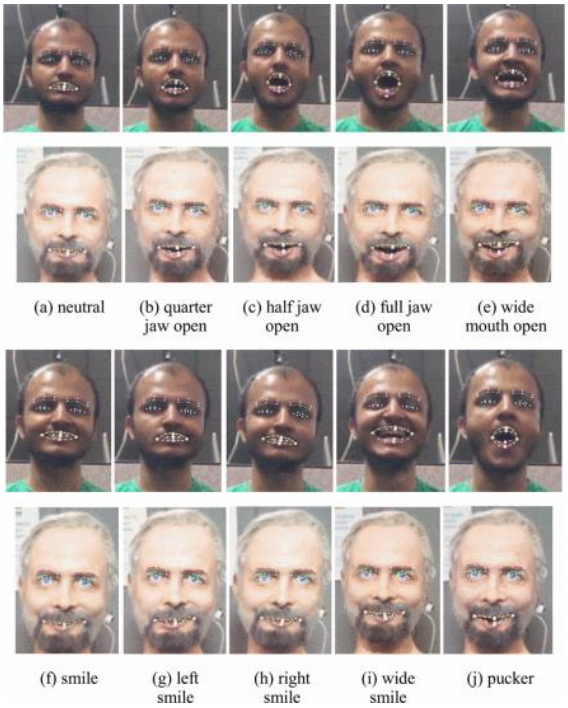

Figure 7. Facial expression from a user and android Phillip K. Dick [17].

2.3 Physiological Sensors and Emotion Classification

Using physiological sensors for emotion classification has been explored by several research groups. Some used units that were created in-house, while others used pre-purchased physiological sensors such as the Shimmer 3, Biopac, EEG headsets, or the Empatica E4. In addition, there is a variety of physiological signals that can be used to understand the emotional state of a subject. Therefore, it is necessary to review the literature and understand what sorts of sensors, signals, and classification methodologies have been used for this task.

In Rahim et al., a series of Shimmer 3 units are used to extract electrocardiogram (ECG) and galvanic skin response (GSR) data for emotion charting. This data is converted into a 2-D frequency spectrum and used in a convolutional neural network (CNN) to classify the emotion. To validate the usage of a scalogram, the researchers used the open-source AMIGOS dataset, which provides long-term and short-term physiological recordings of 40 people. The data is split into two types of social situations, either in a group or individual setting. The researchers found that the AMIGOS dataset yielded an accuracy of 92.7% using GSR and 91.5% using ECG, and, when both signals were combined the accuracy was 93%.

Figure 8. Shimmer 3 sensors [18]

In addition, real-time data was collected from 10 different subjects using the Shimmer 3 units. Different emotions were induced by having the subject watch a video that would invoke one of the seven emotions for classification (anger, disgust, fear, happy, neutral, sad, or surprised). The results found that the model produced an accuracy of 68% on only GSR, 64.2% using only ECG, and, finally, 68.5% when both signals were combined [18].

A second example of using an off-the-shelf physiological sensor can be found in AlZoubi et al. In this research, a BioPac AC MP150 system was used to record the ECG,

EMG, and GSR physiological signals while video was recorded of the learner's face and the computer screen. The study's participants were asked to complete a 45-minute learning session of AutoTutor, during which time, the signals were recorded. The subjects were then asked to rate their emotional state in 20-second intervals based on their facial expressions during the session. The subjects had a choice of eight emotions, which were boredom, confusion, curiosity, delight, engagement, frustration, surprise, and neutral. This data was used as the labels for the emotion classification that the investigators developed. Using several machine learning techniques, such as SVM, K-nearest neighbor, and multinomial logistic regression (among several others), the investigators combined all the subjects' data for the training and test sets. AlZoubi et al. concluded that it was possible to detect affect from physiological data [19].

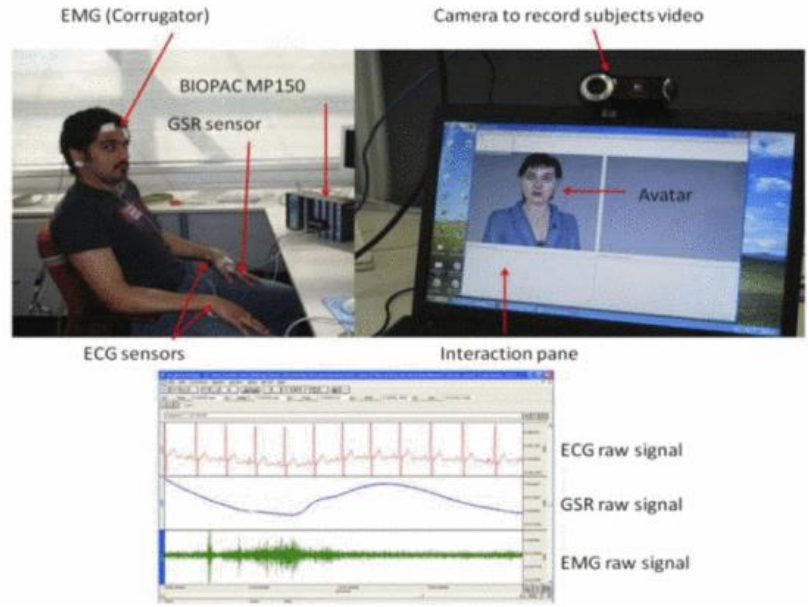

Figure 9. Sensor layout from AlZoubi et al [19]

Other researchers have approached the field of using physiological sensors for emotional classification differently. In Saadatzi et al., a wearable sensor, the EmotiGo,

was developed in-house that was then used for emotion classification. This wearable collected the tonic (slow changing) and phasic (rapidly changes) portions of GSR, photoplethysmography (PPG), and skin temperature. The sensor was integrated into a set of glasses frames that used Bluetooth Low Energy (BLE) to communicate with a computer. A Biopac MP150 was used to enable the comparison of the EmotiGo's performance versus an off-the-shelf device. Three subjects were asked to wear the EmotiGo and were connected to the Biopac at the same time. Data was collected while the subjects performed a series of tests, such as riding a stationary bike, taking a startle test, and completing a Stroop word-color matching test. The researchers found that the EmotiGo can be used for emotion classification because its measurements highly agree with that of the Biopac (which has been used for emotion classification in the past) [20].

Figure 10. The EmotiGo [20]

In Jiang et al., a wristband was created to monitor the emotional health of a subject. The researchers collected pulse, skin temperature, and GSR to recognize the emotions of a subject. The device was able to classify joy, anger, and sadness with around 67% accuracy after 10 trials using a K-nearest neighbor model. The creation of this device was for augmenting home health care for elderly or ill patients [21]. There are many other wristbands and sensors that have been used for the classification of emotions. However,

these have shown that the field of physiological sensors takes many forms and is constantly growing.

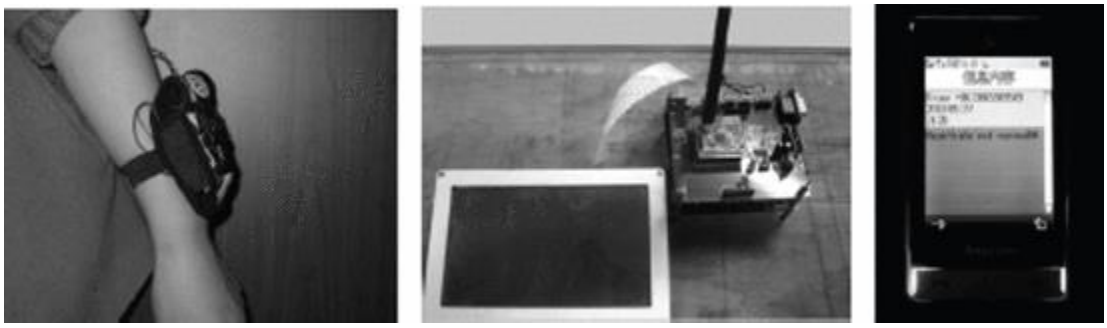

Figure 11. Wearable designed by Jiang et al. [21]

2.4 Motion Quality Studies

Todorova et al. conducted a literature review of 52 papers and concluded that children with ASD struggle with mimicking motions [9]. By being able to assess the motion quality of a subject, it may be possible to identify ASD at an earlier age and rehabilitate these individuals. Therefore, it is necessary to analyze the literature for different ways motion quality assessment has been applied towards analyzing motion and rehabilitation.

One example of motion quality being used to diagnose autism can be found in Zunino et al. In this study, the investigators asked 20 ASD and 20 typically developed children to grasp a bottle and perform four actions with the bottle. The interactions of the subjects with the bottles were recorded and the resulting video data was cut to the instant when the subject grasps the bottle until the subject finishes the experiment. The resulting video data was fed into a model that combined a Long Short-Term Memory Network

(LSTM) and a Convolutional Neural Network (CNN). The resulting network was able to classify the two types of subjects with good accuracy [22].

Figure 12. Samples from Video taken during data collection [22]

A second example of using motion quality to evaluate the imitation deficits in ASD subjects can be found in Wijayasinghe et al. In this study, the researchers used a Microsoft Kinect for Windows to record the motion sequence of the subject while imitating the Zeno robot, this was done for six different motions. The motion sequence was then compared offline to the sequence that the robot performed using Dynamic Time Warping (DTW). The study examined the motion performance of 13 ASD children and 54 neurotypical adult subjects (that were impaired with weights) to prove the validity of using this algorithm. The DTW scores for all four joints were combined into a total DTW score that was used for analysis. The researchers found that using the DTW scores enabled them to identify ASD subjects from others [23].

Motion quality assessment has also been used to help rehabilitate subjects. In Yang et al., the researchers utilized a Microsoft Kinect for Windows camera to help with stroke rehabilitation. These researchers focused on skeletal tracking, the analysis of rehabilitation progress, and the health of the subject. The developed application enabled patients to perform hand and leg training at home, and then provide the performance results for physicians. To aid with tracking the performance of the patient, the researchers simply counted the repetitions for a training scenario [24]. Therefore, analyzing the

motion of the patient enables the rehabilitation of an individual who is recovering from a stroke.

Figure 13. Stroke Rehabilitation system with two subjects [24]

In Wei et al., a rehabilitation program for Parkinson's disease was developed in order to improve the balance and mobility of the subjects. The created application allowed the subjects to perform exercises at home, while ensuring that they do not perform an incorrect motion. The researchers developed a machine learning task recommendation model that examined the actions of the patient. Each motion was divided into several sub-actions that the motion system would grade and identify errors using the angles between predefined points. By identifying errors, the recommendation model would decide the next exercise that the patient should perform (just like a physical therapist) [25]. In this example, assessing motion quality is of the utmost importance, due to the possibility of a subject hurting themselves during an exercise and impeding their recovery.

CHAPTER 3

EMPATICA E4 AND ITS APPLICATIONS

Emotion estimation is a necessary function of the next-generation autism robot. This will enable the robot to monitor the subject's affective states which give useful information to a therapist. In addition, affective information can maximize therapeutic effectiveness by ensuring subjects are attentive and not frustrated.

We propose using a physiological sensor wristband to augment the next-generation autism robot with these capabilities. The work presented here focuses on three main goals. First is to gain access to the capabilities of the sensor, second is to use a live data stream from the wearable to control the Social Robot Zeno as a proof of concept, and the final is to attempt classification of the attentiveness of a patient with ASD. The first two goals will be accomplished using NI's LabVIEW software, and the final classification will use Google's TensorFlow and Python 3.

The Empatica E4 wristband is a wearable device that collects physiological measurements in real time from a test subject. The device provides four raw signals from four sensors: acceleration (ACC), skin temperature (TMP), Galvanic Skin Response (GSR), and Blood Volume Pulse (BVP). Empatica also provides two calculated data streams from the BVP signal: heart rate (HR) and inter-beat interval (IBI).

3.1 Gaining Access to the Empatica E4 Data

To be able to use the E4, we implemented a data collection method for the device. A local server written by the Empatica team running on a Windows machine can be used to communicate with the E4. A telnet connection to the "Empatica Streaming Server" allows us to subscribe and store the data being transmitted by the wristband. Figure 14 provides a sample output of real-time data from the wearable that must be transformed into a form more suitable for research use. We place each unique line into a comma separated string that can be visualized and analyzed.

```
E4_Acc 1580837654.10826 3 64 4
E4_Acc 1580837654.13951 3 64 4
E4_Acc 1580837654.17076 3 64 4
E4_Gsr 1580837652.738 0
E4_Gsr 1580837652.988 0
E4_Gsr 1580837653.238 0
```

Figure 14. Sample E4 Data

Using the gathered data within a closed-loop system, we can identify patterns of changes within the data and communicate to the robot which actions to choose based on those changes. To achieve the above, we must split our system into several parts:

1. Accessing the Empatica E4 server, using a method (or SubVI in terms of LabVIEW) that connects using a TCP connection.
2. A way to read data and stores the data into a LabVIEW object.
3. A data parser that converts the data into a comma separated value format.
4. A LABVIEW SubVI to visualize the data coming from the E4.

These SubVIs will enable data capture, data parsing, and the display of the results and sensor readings.

3.1.1 System

As seen in Figure 15 (a), the wristband uses a Bluetooth to USB dongle to connect to a streaming server (b) provided by Empatica. This server exposes a TCP connection that anyone can connect to, allowing LabVIEW (c) to utilize a TCP Connection to receive data from the E4.

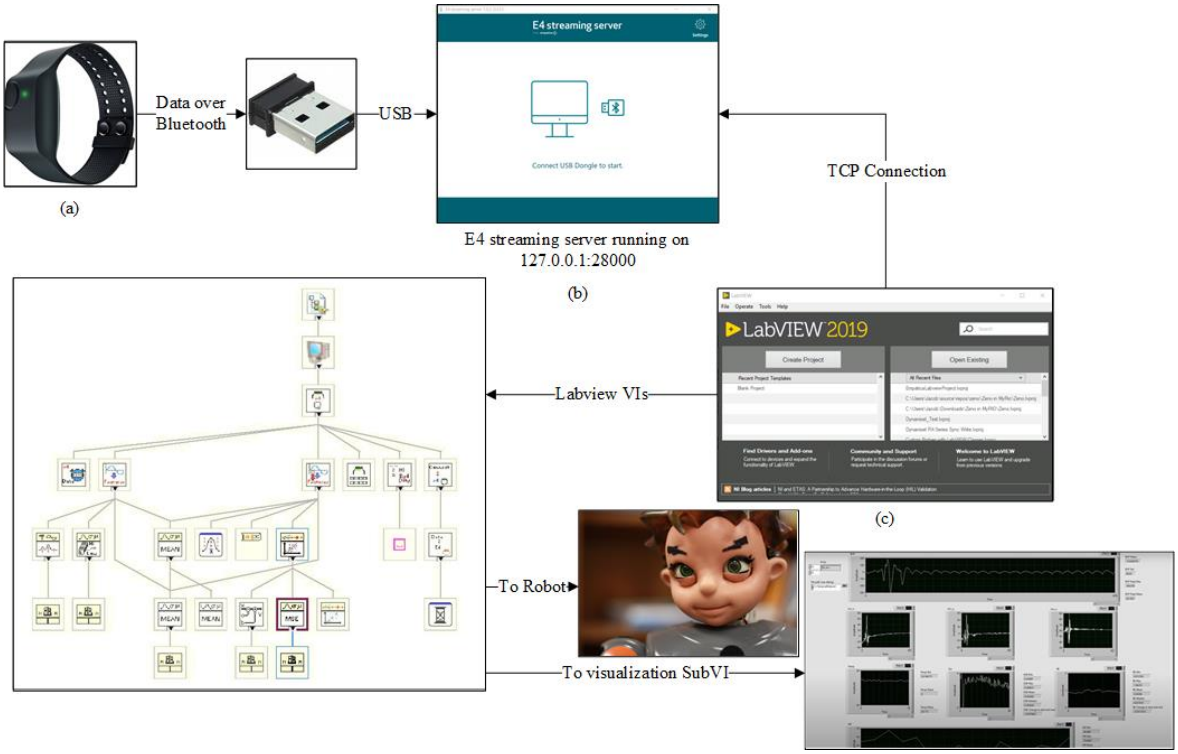

Figure 15. Data Flow between the Empatica E4 wearable and the Zeno Robot in our lab

3.1.1.1 Data Collection

To streamline the above behavior, a SubVI (the LabVIEW equivalent to a method in Object Oriented Programming) was created that is dedicated to connecting to the Empatica E4.

Figure 16 shows the SubVI for this task, which begins by using the “TCP Open Connection” function to open a TCP connection to the Empatica Streaming Server. Note, in Figure 16 we are providing the address 127.0.0.1 (also known as localhost) and a port of 28000, which is the default streaming server port. This does not necessarily mandate that this LabVIEW project must run on the same computer as the Streaming Server. If desired, it would be possible to open a firewall port to allow traffic from a different machine.

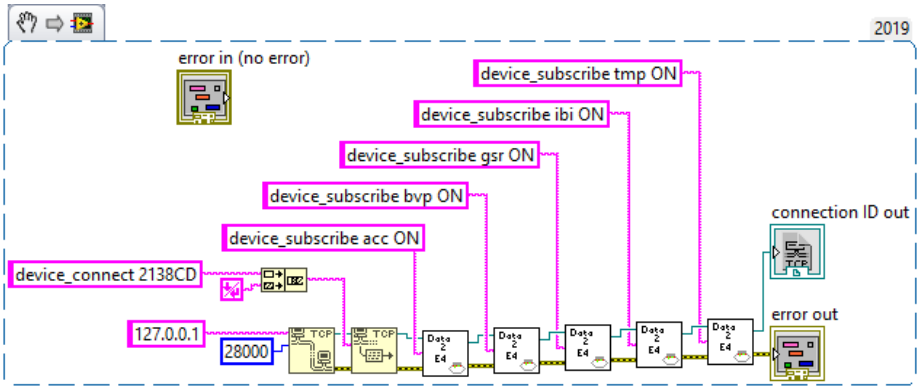

Figure 16. Connect_2_E4.vi

Once we have established a connection to the Streaming Server, we must specify an E4. The Streaming Server allows for multiple E4s to be connected at once. Therefore, to connect we must use a wristband’s specific unique identifier. This identifier can be found by using “device_list” when connected to the Streaming Server. It is also important to note that an end of line (EOL) constant must be sent so that the streaming server registers the end of a command.

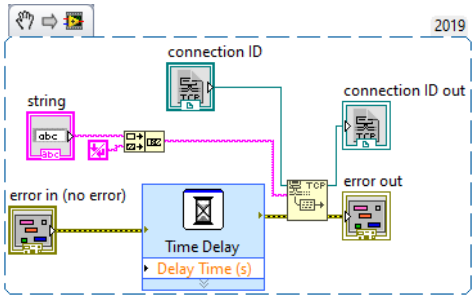

Figure 17. SendData2E4.vi

To simplify future commands sent to the streaming server, a secondary SubVI is created that appends the EOL constant and adds a delay of one second (using the "Time Delay" Express VI) to ensure that a command has been received and processed. Once an E4 wristband has been connected, all the data streams of interest must be subscribed to. See Table 1 for a list of data stream codes to sensor. After subscribing to all the data streams, it is now possible to begin receiving and parsing the data.

Table 1. Empatica Streaming Server Codes to Sensor

Code	Sensor
acc	Accelerometer X, Y, Z
bvp	Blood Volume Pulse (BVP) Raw Signal
gsr	Galvanic Skin Response (GSR) Raw Signal
ibi	Inter-Beat Interval (IBI) Calculated Signal
tmp	Temperature Raw Signal

Figure 18 shows the "Main Loop" of the Empatica project, using the Producer/ Consumer Design Pattern. This design pattern is based upon the Computer Science theory of the same name. One thread, or in this case "loop," is dedicated to creating work for a second "consumer" thread. This enables parallel execution of a process while not sacrificing data capture. The Producer Loop calls the SubVI

“Get_e4_Data_Over_Period” which will produce data for the consumer. Once the data is produced it will add an element to a queue which will then be passed to the Consumer loop.

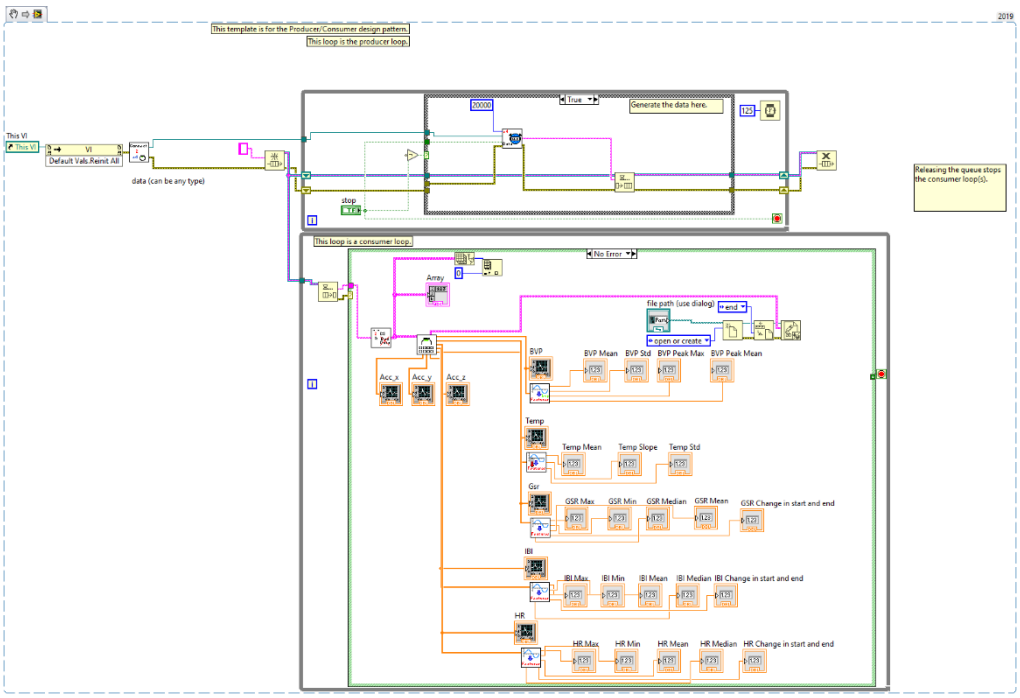

Figure 18. MainLoop.vi

Figure 19 shows the process for data collection when the “Get_e4_Data_Over_Period” SubVI is called. This SubVI receives an unsigned integer as an argument that specifies the amount of time to collect data (by default 20000 milliseconds or 20 seconds). In the “Get_e4_Data_Over_Period” SubVI the existing open TCP Connection is used to read data that is being transmitted from the Streaming Server (with the help of the “TCP Read” function). One of the required arguments of the “TCP Read” function is the number of bytes to read.

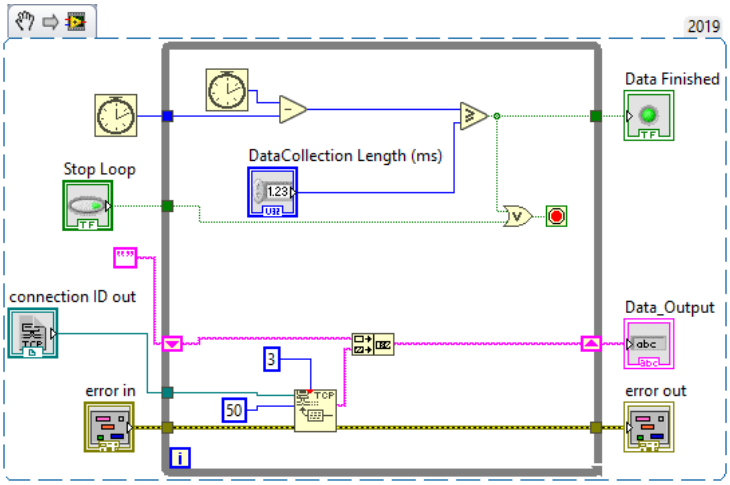

Figure 19. Get_e4_Data_Over_Period.vi

To determine this, we analyzed the traffic that is being streamed from the server. We use Wireshark [26] (see Figure 20) to see the TCP header; the length observed is 34 bytes. The packet length is rounded up to account for bigger packets, which is why a size of 50 bytes is used. After reading the data, it is appended to a string using a shift register, which will then output into the “Data_Output” control. To be able to capture data over a time period, it is necessary to iterate until the specified time has elapsed. This is gauged using the “Tick Count” function which takes the millisecond count. The SubVI captures the time at the beginning and the current count. The two values are then subtracted and compared to the “DataCollection Length (ms)” control. If the result is greater than or equal to the threshold, then the SubVI will finish execution. The enqueued data is now ready for the next step.

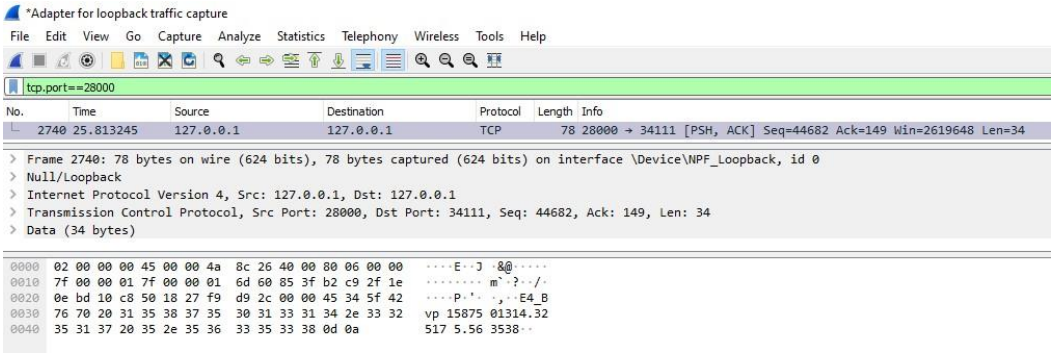

Figure 20. Wireshark Snippet

3.1.1.2 Data Processing

Once the data has been produced, by definition, it must be consumed. To do this the Main Loop dequeues the data into the consumer loop. This data is then passed to the “ParseString” SubVI which is responsible for data cleaning and parsing (seen in Figure 21).

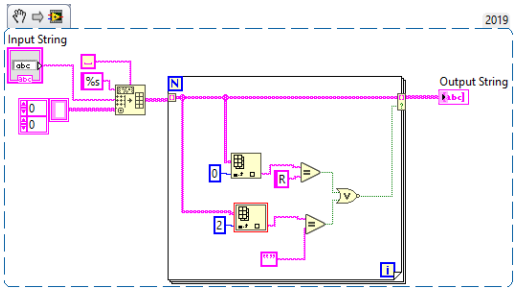

Figure 21. ParsingString.vi

The input data is parsed into a 2d array by the “Spreadsheet String to Array” function. Using the space constant and the format string of “%s” every whitespace character is considered a delimiter (e.g., a comma in a CSV). The data has now been parsed and is ready for cleaning. The 2d array must be cleaned of bad data. To do this it is necessary to

remove streaming server status responses and partial data (i.e., status responses are identified by the prefix R and partial data is identified if the third [data] argument is missing). The clean data is now ready for visualization and further processing.

To be able to visualize and extract features, the data must be split into respective groups per sensor. This is done by the "GetArrays" SubVI (illustrated in Figure 22). Using the "Index Arrays" function, the first column is used to specify which array the data column belongs to. Using a Conditional Tunnel, we can use a Boolean to sort the data into the appropriate array. However, if the data is from the accelerometer, then it is necessary to sort the data into the X, Y, and Z axes. To be able to split, the columns for each axis must be explicitly specified. This SubVI will also convert the data to a CSV format, using the upper-case statement which will go to a format of "%s,%s,%s" by default. If the data is from the accelerometer, then it must be split by the axes. Once all the above operations have concluded the data is ready for visualization and basic feature extraction.

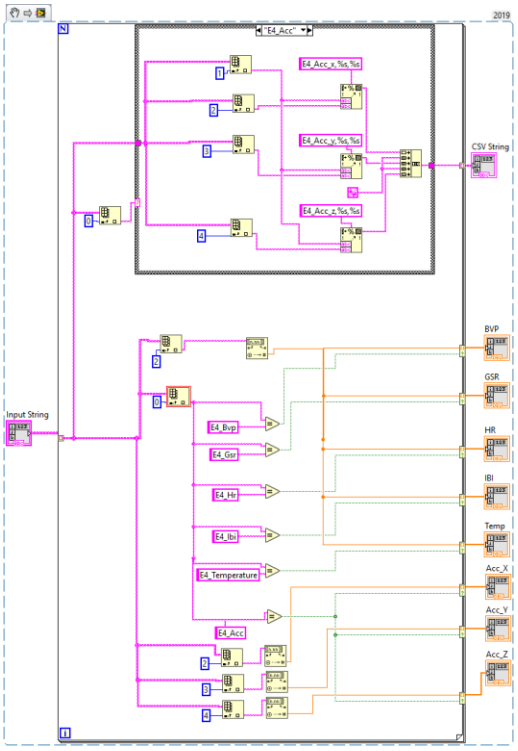

Figure 22. GetArrays.vi

3.1.1.3 Data Visualization

To visualize the data, the arrays are first input into a series of waveform graphs. These allow the user to see the data that has been produced after collection. Next, the created CSV is appended to a file. This file will be used for further data analysis (e.g., model training or to diagnose issues). Finally, the features are extracted using the techniques described by Welch et al. [27].

The “ExtractFeatures” SubVI is used as the default feature extraction function. This is meant to be a placeholder for exploring some basic features of the signals. This work served as a precursor to the attentiveness classification study at the University of Louisville Autism Center. Figure 23 illustrates the general feature extraction SubVI. The

features that it extracts are: Mean, Median, Max, Min, Peak Locations, Peak Amplitudes, and the Change between the start and end of the signal. This is done using the built in LabVIEW functions. Since this SubVI is run in parallel, it is necessary to enable the "Preallocated clone reentrant execution" option.

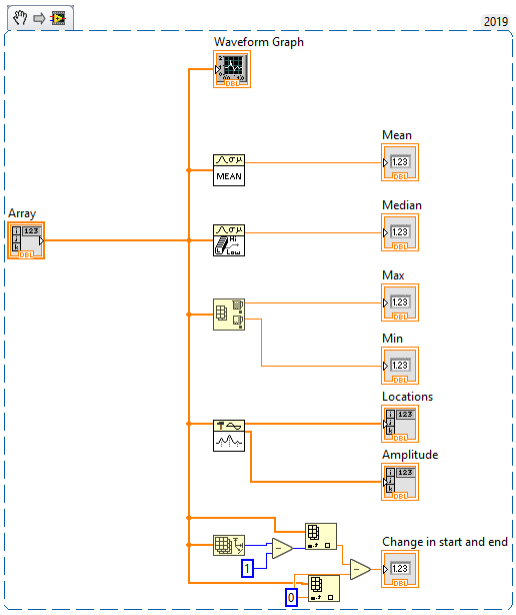

Figure 23. ExtractFeatures.vi

Now that the waveform graphs have been displayed to the user, the consumer loop finishes its iteration. The consumer loop will wait for new data in its queue (that is produced by the producer loop) and will begin another iteration. This continues until the stop button has been hit, or the VI has been manually stopped. Due to the projects focus on data collection, error handling has not been incorporated. In the future it will be necessary to decide how to handle data errors.

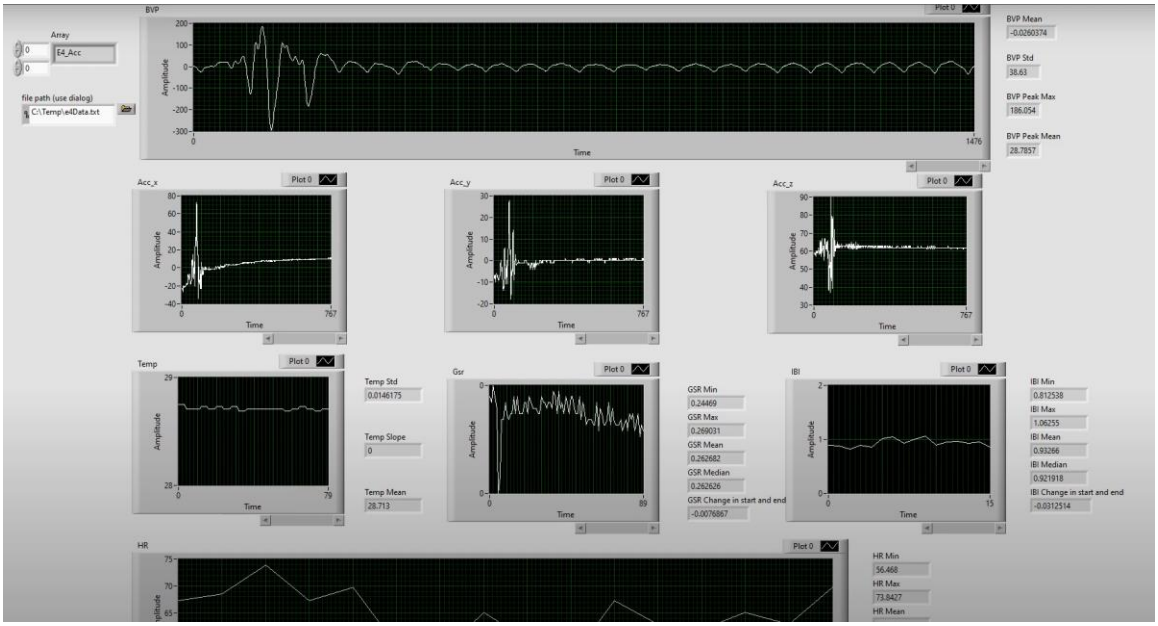

Figure 24. An Example of Real Time Data from E4

3.2 Controlling Zeno's Arm with the Empatica E4

As the second goal in this work, we added functionality to the social robot Zeno as a proof of concept. In the past, Zeno was used for an experiment with patients with autism. This experiment involved creating a LabVIEW application that would use a Kinect to mimic the therapist.

To move Zeno's arm with the Empatica E4 we created a project based on the Kinect demo for manipulation. Since Zeno uses environmental variables to move, these variables can be hijacked for a different purpose. This is done by setting constants on all the variables (or joints) other than the one of interest. Then, using the previously discussed SubVIs, it is possible to capture the accelerometer data. It was found that the Z axis showed the most pronounced change in data when moving the user's arm; therefore, it was used to control the movement of Zeno's arm.

The SubVI in Figure 25 shows how the data from the accelerometer is used as a binary switch. The coordinate from the Z axis defines which way the motor will move the arm. Using this technique, the robot seems like it is mimicking the movement of the wrist. However, it will always go to one of two predefined points.

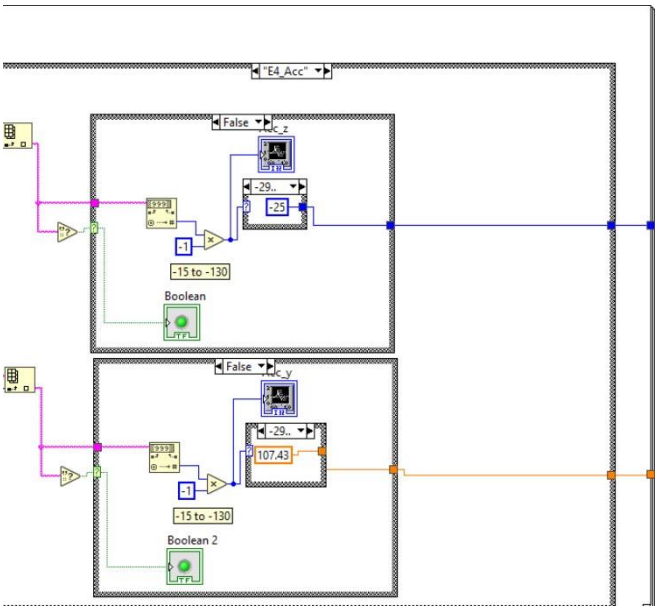

Figure 25. e4_Zeno_Move.vi's binary switch

3.3 Classifying Attentiveness

Our final goal was to experiment with using the Empatica E4 to classify data from the social skills group run by the University of Louisville Autism Center. During this study six subjects were split into two groups for robot interaction. Each group was placed into a room with a Nao robot (a robot produced by Softbank Robotics), Figure 26, and were given a list of questions to ask the robot. The Nao robot features a microphone and speech recognition software which enables it to interpret the response of the subject. Once a question was asked, the robot would respond and then wait for a second question,

this continued for a 3-to-6-minute session every week (for 12 weeks). The same subjects would wear the E4 wristband during the interaction with the robot, while the others would wear dummy wristbands that were not recording data. We collected E4 data from two subjects during the human-robot interaction sessions and used it in our classification algorithms. In order to collect a baseline, the wearables were turned on several minutes before the subjects entered the room.

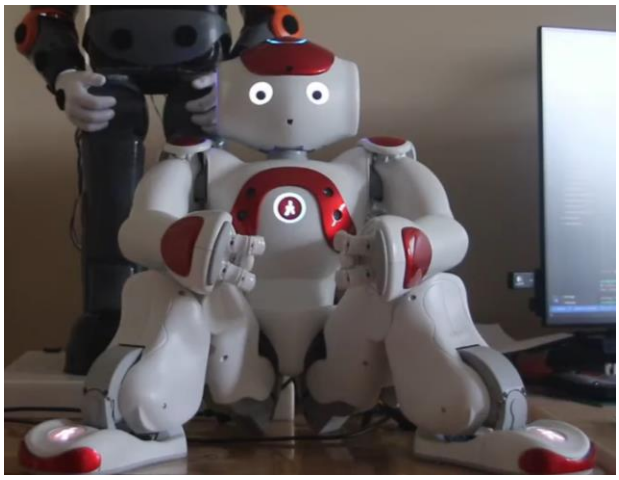

Figure 26 Nao Robot

3.3.1 Gathering and Labeling Data

The Empatica enabled us to gather BVP, GSR, HR, and Temp for both subjects for approximate total of three hours of data. The IBI sensor data was not used for classification due to possible discontinuities, therefore this could potentially inhibit the algorithms classification accuracy. The accelerometer data was not used due to its lack of use in previous studies [27]. This signal may have useful information about changes in attentiveness, however, this work focuses on other signals. During this same time, a time-synced video recording of both subjects which enabled the syncing of the E4 data and the video. A group of professional coding staff employed by the Department of Education

were asked to label whether each subject was attentive or inattentive every 20 second period while watching the collected videos.

3.3.2 Feature Extraction

The physiological data was split into 20 second periods and computed into different features, for use in our deep learning models. We computed the following features for every 20 second interval:

- BVP Peak Mean - Average of the peak amplitudes of the BVP signal
- BVP Peak Max - The maximum peak amplitude for the given time frame
- Tonic Mean - The average of tonic skin conductance, derived from the GSR signal
- Tonic Slope - The slope of the tonic skin conductance
- Phasic Peak Mean - The average of the phasic skin conductance (derived from the GSR signal)
- Phasic Peak Max – The maximum peak of the phasic skin conductance
- Phasic Peak Rate – The number of peaks per 20 second period multiplied by three
- HR Standard Deviation – Standard deviation of the heart rate
- HR Mean – Average of the heart rate
- Temp Standard Deviation - Standard deviation of the temperature
- Temp Mean – Average of the temperature

- Temp Slope – Slope of the temperature

Each feature was normalized to have a value between zero and one, for use in our models.

3.3.3 Deep Learning

The small dataset posed several challenges for use with deep learning. When both subjects are combined there are 543 usable data points of which 124 are inattentive and 419 are attentive. The usable points were defined as the features that had all of the data during the 20 second time frame during feature calculation. It was possible for some data to be lost due to a lost connection or a subject accidentally turning off the wristband.

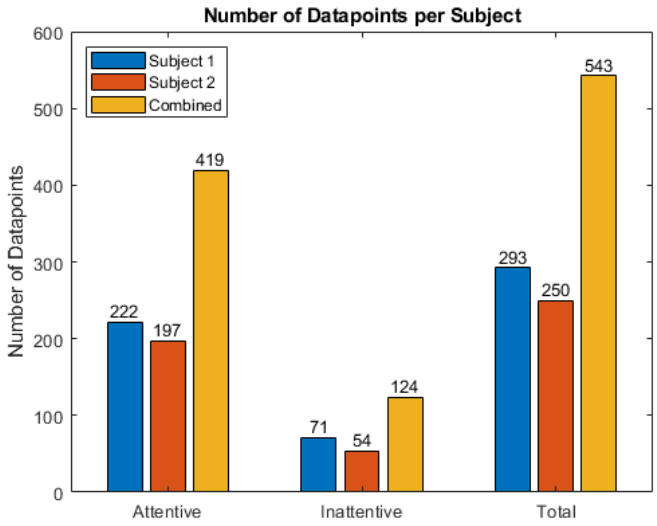

Figure 27. Number of Datapoints per Subject

This yields an unbalanced dataset with a very heavy bias towards attentive data. To combat this, we split the total data set into a Python dataframe of all the attentive points and a second dataframe containing all the inattentive points. We then randomly

sample our attentive dataset to have the same amount of data as the inattentive one. This ensures that there is no longer any bias towards attentive instances.

To ensure that the train and test set both have the same distribution of inattentive and attentive data (to prevent instances of the test set containing all of one type of data) we use sklearn's "train_test_split" method on both dataframes. We use a 20/80 test/train split due to the small amount of data. The two resulting training sets are concatenated together and randomized, with the same done to the test sets.

We experimented with combining the two subjects into one dataset as well as separating them. In addition, we tried several types of neural network layouts: shallow versus deep, epoch amount, and different loss functions. In total 54 models were trained using Google's Tensorflow [28], 18 on each dataset (subject 1, subject 2, and both).

3.3.4 Models Experimentation

To gain clarity on whether the Empatica E4 and deep learning neural networks can be used to classify attentiveness, it is first necessary to experiment with different hyperparameters. The goal of this work is to train a model that can classify attentiveness with an accuracy of greater than 50%. Experimenting with different hyperparameters will provide guidance for future development of a successful model.

Prior to choosing the variables to experiment with it was decided to keep the Input and Output layers consistent between networks. The Input Layer would require all 12 precomputed features from Section 3.3.2, and the output layer would utilize a tanh function. The tanh would ensure that the output would always be between zero and one, with zero being an inattentive classification, and one being attentive classification.

The first experiment involved varying the number of layers with different numbers of hidden layers. Figure 28 shows the layout of the “Shallow Network” which was used in our experimentation. This layout features a single hidden layer that had 16 hidden units.

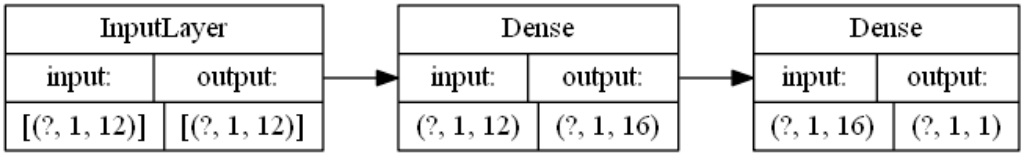

Figure 28. Shallow Network

Figure 29 shows the Two Layer Network layout, which builds upon the Shallow Network by adding a second hidden layer with 32 hidden units.

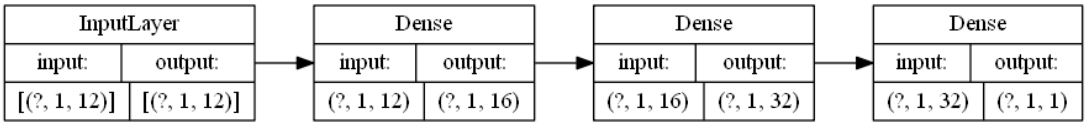

Figure 29. Two Layer Network

Figure 30 illustrates the Three Layer Network which builds upon the Two-Layer Network by adding a third hidden layer with 32 hidden units.

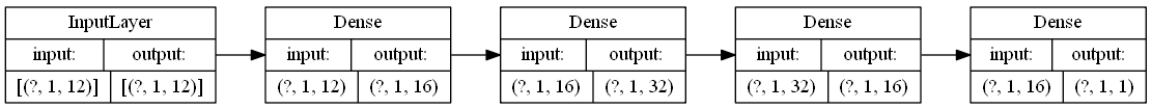

Figure 30. Three Layer Network

Finally, Figure 31 shows the Five-Layer Network which takes the Two Layer Network and places a fourth and fifth layer with 64 and 32 hidden units each. Table 2 shows the hyperparameters used for the Layer Size Experimentation. These constant parameters were chosen through experimentation on the training set.

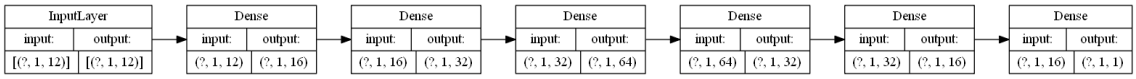

Figure 31. Five Layer Network

Table 2. Layer Size Experiments

Model Name	Epochs	Loss Function	Optimizers	Activation	Learning Rate
Shallow Network	4000	Adam	MeanSquaredError	ReLu	0.0002
Two Layer Network	4000	Adam	MeanSquaredError	ReLu	0.0002
Three Layer Network	4000	Adam	MeanSquaredError	ReLu	0.0002
Five Layer Network	4000	Adam	MeanSquaredError	ReLu	0.0002

The second experiment performed was on the number of epochs the Two Layer Network was trained for. Table 3 shows the naming convention between each model and the Epoch number.

Table 3. Number of Epochs Experiment

Model Name	Epochs	Loss Function	Optimizers	Activation	Learning Rate
Two_Layer_2000	2000	MeanSquaredError	Adam	ReLu	0.0002
Two_Layer_2500	2500	MeanSquaredError	Adam	ReLu	0.0002
Two_Layer_3000	3000	MeanSquaredError	Adam	ReLu	0.0002
Two_Layer_3500	3500	MeanSquaredError	Adam	ReLu	0.0002
Two_Layer_4000	4000	MeanSquaredError	Adam	ReLu	0.0002

The third experiment performed was on the different Loss Functions that are commonly used for Binary Classification (shown in Table 4) [29]. It should be noted that Two_Layer_MeanAbsoluteError is the exact same model as Two_Layer_2500 however it has been renamed for use in this experiment (it was not retrained).

Table 4. Loss Function Experiments

Model Name	Epochs	Loss Function	Optimizers	Activation	Learning Rate
Two_Layer_BinaryCrossEntropy	2500	Binary Crossentropy	Adam	ReLu	0.0002
Two_Layer_Hinge	2500	Hinge	Adam	ReLu	0.0002
Two_Layer_MeanSquaredError	2500	Mean Squared Error	Adam	ReLu	0.0002
Two_Layer_MeanAbsoluteError	2500	Mean Absolute Error	Adam	ReLu	0.0002

The fourth experiment performed was on the viewing the differences between the Adam and Stochastic Gradient Descent (SGD) optimization algorithms [30]. It should be noted that Two_Layer_Adam is the exact same model as Two_Layer_2500 (from the Epoch experiment) however it has been renamed for use in this experiment (it was not retrained).

Table 5. Testing Optimizers

Model Name	Epochs	Loss Function	Optimizers	Activation	Learning Rate
Two_Layer_Adam	2500	Mean Squared Error	Adam	ReLu	0.0002
Two_Layer_SGD	2500	Mean Squared Error	SGD	ReLu	0.0002

The fifth experiment performed was on the viewing the differences between the Sigmoid and ReLu activation functions. These activation functions were applied to both hidden layers, while the output layer kept its tanh activation function. It should be noted that Two_Layer_Sigmoid is the exact same model as Two_Layer_2500 (from the Epoch

experiment) however it has been renamed for use in this experiment (it was not retrained).

Table 6. Activation Functions

Model Name	Epochs	Loss Function	Optimizers	Activation	Learning Rate
Two_Layer_Sigmoid	2500	Mean Squared Error	Adam	Sigmoid	0.0002
Two_Layer_ReLu	2500	Mean Squared Error	Adam	ReLu	0.0002

3.3.5 Results

In total the above distinct models were trained on the three data sets: both subjects combined, subject one, and subject two. Figure 32 shows the results of training the four different models on the three different datasets. We can conclude that the Three Layer network would perform best on a combined subject dataset (based on the accuracy) and the Two Layer network would perform best overall based upon the average accuracy score across all three datasets. The performance of the models could be improved with more feature types and data. One way this could be performed is by using a publicly available dataset (such as AMIGOS) train attentiveness on any type of subject. The trained model can then be used for transfer learning to enable specialization on a specific subject.

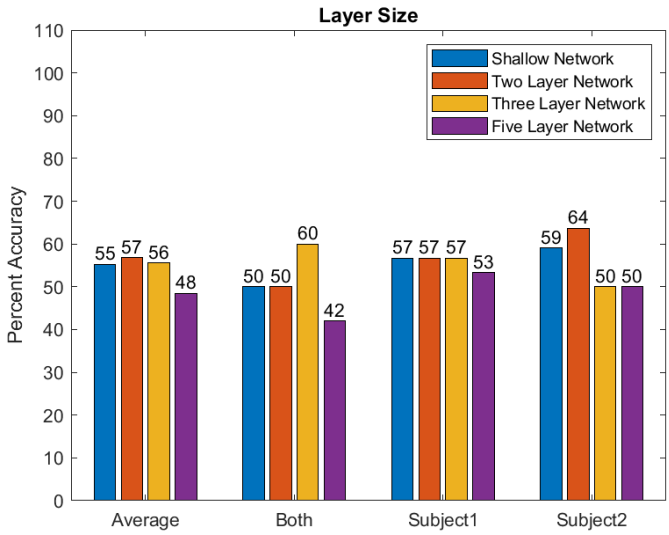

Figure 32. Layer Size Experiment Results

Figure 33 shows the results of training the Two Layer model on the three different data set for different epoch amounts. The number of epochs is the number of iterations a neural network will iterate over the training set to find the best weights. By increasing the number of epochs, it is possible to find the point at which the loss function is minimized, however, this risks overfitting on the training set, therefore performing poorly on the test set. We can conclude that the network would perform best training for 2500 or 3500 epochs. We chose to use 2500 in our experiments to prevent overfitting on the training set.

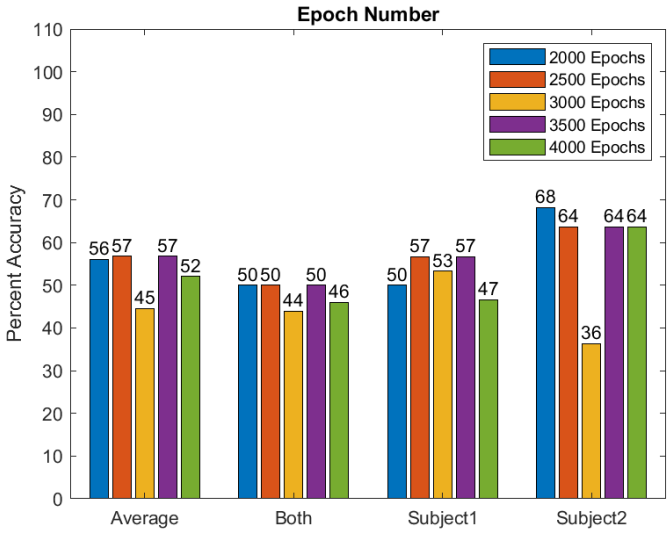

Figure 33. Epoch Number Experiment Results

Figure 34 shows the results of training the Two Layer model on the three different data sets for different loss functions. We can conclude that the network performs best by using the Mean Squared Error loss function.

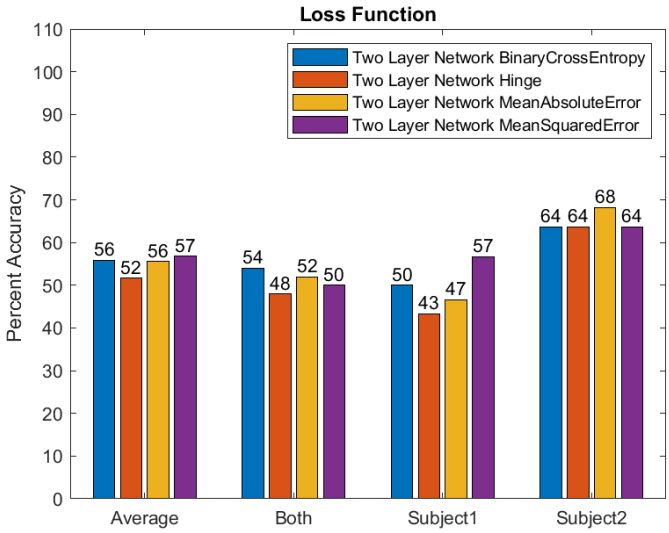

Figure 34. Loss Function Experiment Results

Figure 35 shows the results of training the Two Layer model on the three different data sets for different optimizer functions. We can conclude that the network performs best when using the Adam Optimizer.

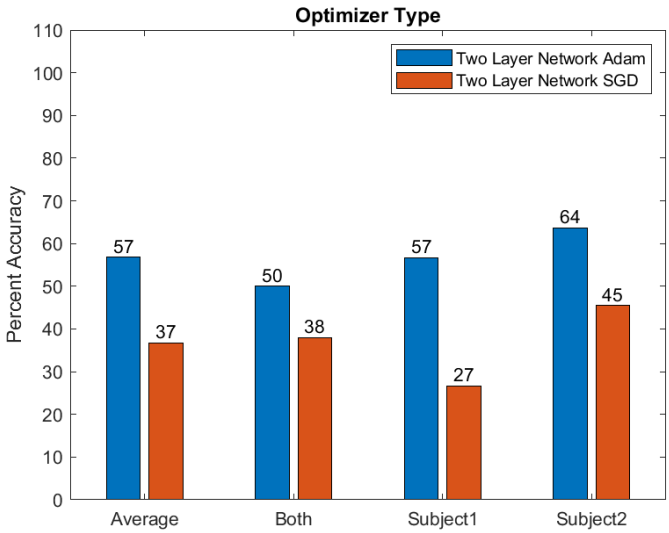

Figure 35. Optimizer Experiment Results

Figure 36 shows the results of training the Two Layer model on the three different data set for different activation layer functions. We can conclude that the network performs best by using the ReLu activation function during training.

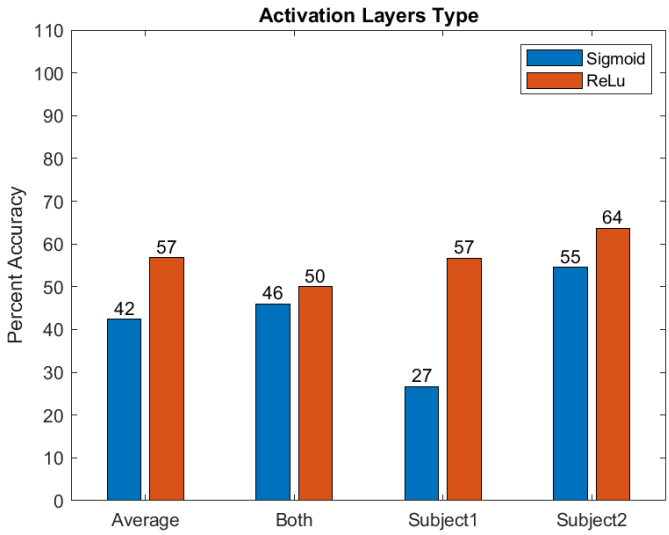

Figure 36. Activation Layers Experiment Results

3.3.6 Future Work

In conclusion, the data from the Empatica E4 is useful for classification of attentiveness and inattentiveness of an ASD subject. Unfortunately, due to the low amount of data, it was not possible to create a model that was able to perform better than 68% on at least one subject. The highest percentage found from all the experiments, per dataset, was 60% for the combined subject data, 57% for a model of Subject 1's data and 68% for a model of Subject 2's data. There were several models that performed at 57% accuracy for Subject 1, all shared the characteristics of using a ReLu activation function and Adam Optimizer. For Subject 2, there were two best performing models both were two-layer networks with a ReLu activation function, and an Adam optimizer. Overall, we suggest using a model with two hidden layers, trained for 2500 epochs, using the mean squared error loss function, the Adam optimizer, and ReLu activation functions for the two layers.

The best performing model, that was produced as part of this work, would match a human coder on predicting attentiveness or inattentiveness about two-thirds of the time. During new interventions with these subjects, the model could be used by a robot to test a new sample of physiological data, return a prediction of affect (e.g., attentiveness or inattentiveness), and decide about actions to take during the intervention. For example, if the model predicts the subject's signals indicate attentiveness, the robot can choose to continue with practicing a social skill. However, if the model predicts inattentiveness, the robot can remind the subject about the directions and about staying on task. In future research on model building, we would like to explore gradient boosting and collecting more data by creating a study that would create a case where the subject is inattentive.

CHAPTER 4

BAXTER TELEOPERATION USING OPENPOSE

Teleoperation of a robot is the process of remotely controlling the pose of the robot arm or end-effector by direct input from the human operator. This input is typically provided via a robotic mechanism local to the user, such as a gaming joystick controller, or another robotic manipulator. In this section, we achieve remote imitation operation of a robot manipulator arm, such as social robot Zeno's arms, or Baxter's dual manipulator arms, through a non-contact visual interface that estimates the position of the human hand, then calculates corresponding poses for the remote robot.

Teleoperating the next-generation autism robot will enable the ability to puppeteer the robot when needed. This also provides a way to test and record motions for automated performance of these motions later. In this work we propose using the Baxter robot, a dual robotic manipulator, which was developed by Rethink Robotics. This is a collaborative robot designed for assembly line tasks; therefore, it is well suited for repetitive movements. The purpose of the work presented in this chapter is to accomplish remote teleoperation of dual manipulator arms that mimic human arm movements, for instance those involved in social interaction, or during assembly line operations in factories. Examples include teaching by demonstration by a remote user of pick-and-place operations, or imitation of hand gestures for the purpose of navigation or communication with factory floor personnel or other robotic assets.

4.1 System Architecture

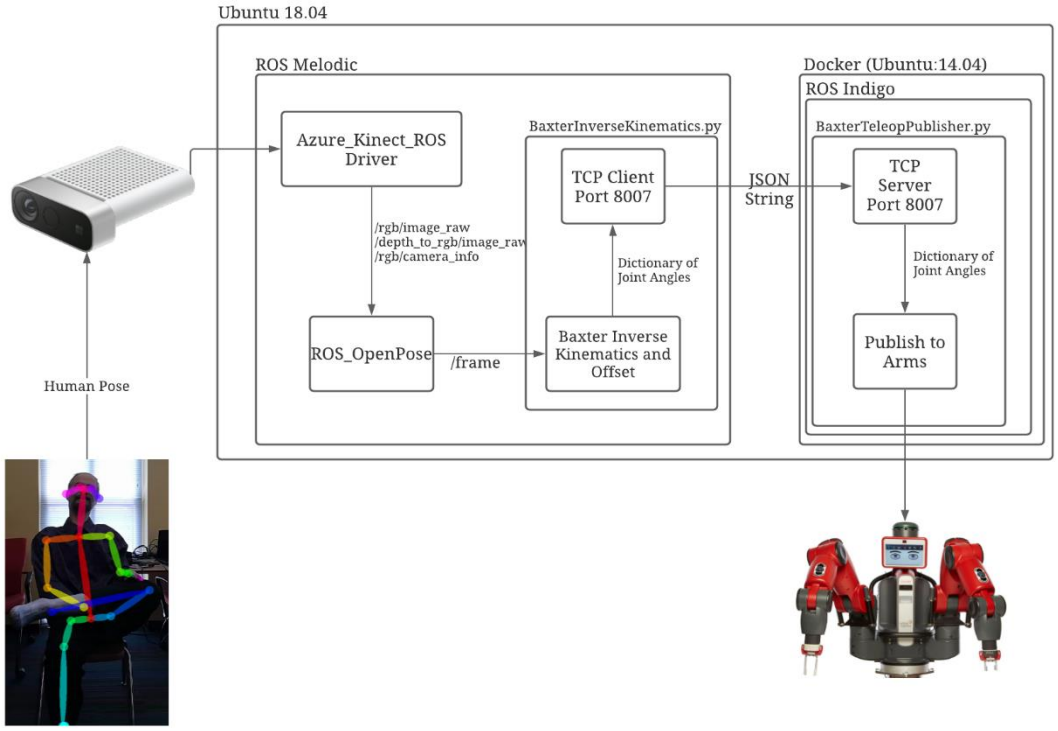

Figure 37. Baxter Teleoperation System Architecture

There are several pieces of the overall system that must be developed to teleoperate a dual manipulator system such as the Baxter robot. There must be a vision system that translates the user's motions into a Cartesian Coordinate frame. Next, something must compute the inverse kinematics to translate the human pose into a valid pose for the robot. Finally, the system must tell the robot the joint angles to move. Due to the different operating system requirements of the Azure Kinect, OpenPose, and Baxter, a virtualization platform called Docker is used.

The Azure Kinect ROS Driver mandates that Ubuntu 18.04 must be used to publish data from the camera to ROS. This ROS Driver creates a series of topics that allow a user to subscribe to the data from the camera. The “ROS_OpenPose” wrapper (developed by Joshi et al.) uses the “/rgb/image_raw”, “/depth_to_rgb/image_raw”, and “/rgb/camera_info” topics from the Azure Kinect to compute the XYZ coordinates of each body part [33].

The “/rgb/image_raw” topic gives a raw file image that is directly provided to OpenPose for processing. This yields X and Y coordinates of all the features the user has selected (we are currently only receiving hand and body data). Once the XY coordinates have been received, the “ROS_OpenPose” wrapper uses the “/rgb/camera_info” topic to get information about the camera calibration. This information is important to translate the XY pixel coordinates from OpenPose to distance data. Finally, the “/depth_to_rgb/image_raw” data is combined with the translated XY coordinates to form the complete XYZ data. The complete data is then published to the /frame topic as a custom message.

Once data is published to the /frame topic, the “BaxterInverseKinematics” subscriber uses the provided coordinates to calculate the inverse kinematics. The subscriber selects the shoulder, elbow, and wrist (from both sides) to compute the resulting joint angles. After these joint angles are computed, they must be offset to achieve the desired pose. This is done to match the initial pose from the motion camera to the initial position of Baxter. After the offset, the system uses a TCP socket to send data for publishing to the robot (which uses Docker).

Docker is a container-based virtualization platform that is meant to deliver an application as a package. Docker ensures that the developer and production environments for a piece of software are always the same and all dependencies are met. For the application of robotics Docker (and similar virtualization software) decouples the robotics software from the developer's hardware. Ensuring that a developer computer failure will not cause a robot to become inoperable because all the necessary software was installed in one place.

To satisfy the requirement that multiple people need to work on Baxter software and to isolate the robot from the outside world, a Docker container was developed to communicate with Baxter. This container installs ROS Indigo and Baxter-specific packages, creates a developer environment, and sets up a private Local Area Network (LAN). The created Docker container establishes a way for the system to communicate with Baxter through ROS Indigo.

On the ROS Indigo network, a node named "BaxterTeleopPublisher" is running that serves as a bridge between ROS Melodic and ROS Indigo. This node hosts a TCP server that receives data from the "BaxterInverseKinematics" node. When the server receives data, it will create a thread, load the valid json string into a dictionary, and create a "JointCommand" that is published to either the "/robot/limb/right/joint_command" or "/robot/limb/left/joint_command" (depending on which side of the robot the joint angles are for).

Finally, the Baxter moves to the desired position, in future work this portion will be improved so that the desired positions are constantly published. This is necessary because the robot will only begin moving to the desired position if that position is

published constantly. In future a thread will be used to constantly publish the current desired joint position.

4.2 Description of Imaging Hardware and Software

The Baxter robot features two arms each with seven degrees of freedom, an ultrasonic sensor grid, three cameras (one on each arm near the end-effector, and one on the main display), accelerometers, and sensor pads for robot manipulation by a human. In our lab the Baxter robot is outfitted with a pedestal (giving it a total height of around 6' tall, and a wingspan of approximately 103") and a pair of electric parallel grippers [34]. The orientation and labeling of Baxter joints can be seen in Figure 38.

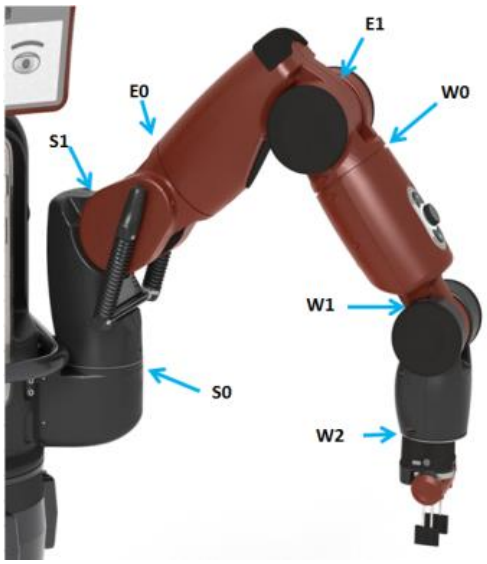

Figure 38. Joint Description [35]

Internally Baxter is running a computer with an Intel Core i7-3770 8MB 3.4 GHz processor with 4GB of DDR3 ram and a 128GB SSD [36]. This system runs ROS Indigo released for Ubuntu 14.04 LTS on July 22nd, 2014 [37]. Therefore, any control with

Baxter must be done over a system that is able to communicate with ROS Indigo. Unfortunately, there are still considerable inter-compatibility challenges between ROS versions, which necessitated special ROS nodes to bridge the gap between software modules written.

CMU's OpenPose software package enables body, face, and hand tracking using an RGB camera (such as a web camera). OpenPose can identify the 2D pose of multiple people in an image [38]. Using multiple 2d image sources of a subject, the software can produce XYZ coordinates of the 135 keypoints. However, others have implemented a wrapper that pairs an RGB-D camera with OpenPose. For the purposes of this research, we used the "ROS_Openpose" wrapper created by Joshi et al. [33]. The 2D image from the RGB-D camera is sent for evaluation by OpenPose, and the resulting information is then combined with Depth information from the camera.

Previous works have utilized the Kinesthesia toolkit from the University of Leeds [42]. However, this toolkit is only used in LabVIEW and is heavily tied to the use of the first-generation Xbox 360 Kinect. OpenPose is hardware-agnostic and allows us to inform the robot about hand grasping data (for control of a gripper) or facial features (to potentially enable gaze tracking).

The Azure Kinect is an RGB-D sensor released in March 2020 and is produced by Microsoft. It features a one megapixel depth sensor with wide and narrow field of view, a seven-microphone array for sound capture and far-field speech, a 12-MP RGB camera for color information, an accelerometer, gyroscope, and finally sync pins to synchronize streams from multiple devices [43]. This comes in a package size of 103x39x126 mms

and has a weight of 440g [43]. Finally, the Azure Kinect ROS Driver supports ROS Melodic in Ubuntu 18.04.

4.3 Kinematic Model of Baxter's Right Arm

To understand how Baxter's joints work, we created a model using Peter Corke's Robotics toolbox [45]. This enables the ability to model a robot using a series of links and Denavit-Hartenberg Parameters (DH Parameters). These parameters are used to attach reference frames to each link of the robot to generate forward kinematics. Table 7 shows the DH Parameters used to generate the right arm of the Baxter robot. Figure 39 shows the result that is created from the Robotics toolbox. Each joint row in the table has an entry for link twist (α_{i-1}), link length (a_{i-1}), link offset (d_i), and joint angle (Θ_i). To calculate the DH Parameters, first, a model of the robot arm is drawn and labeled with an X and Z axis. Figure 40 shows each joint and its corresponding axis using the drum notation.

Table 7. DH Table for Baxter

Joint	α_{i-1}	a_{i-1}	d_i	Θ_i
S0	0	0	0	$\frac{\pi}{2}$
P0	0	0	0	$\frac{\pi}{2}$
E0	0	L_0	L_1	Θ_3
S1	$\frac{\pi}{2}$	0	0	Θ_4
W0	$-\frac{\pi}{2}$	L_3	L_2	Θ_5
E1	$-\frac{\pi}{2}$	0	0	Θ_6

W2	$\frac{\pi}{2}$	0	L_4	Θ_7
W1	$-\frac{\pi}{2}$	0	0	Θ_8
Gripper	$\frac{\pi}{2}$	0	L_5	Θ_9

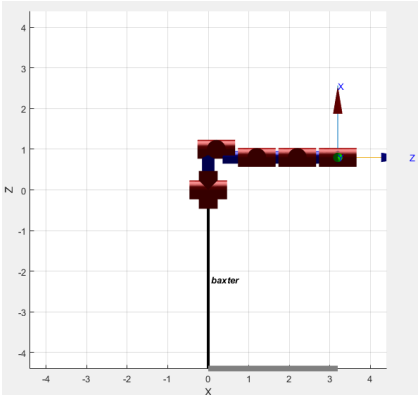

Figure 39. Kinematic model generated using Peter Corke's Robotics toolbox

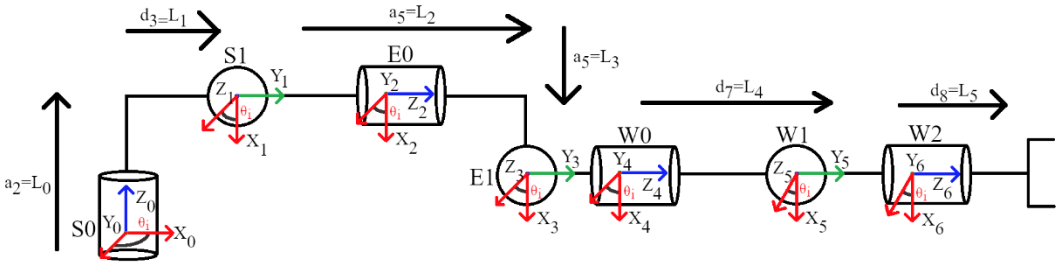

Figure 40. Deriving the DH Parameters

The joint angle Θ_i is the angle argument that defines how much to move the joint. Therefore, this column in the DH Table is generally set to the variable Θ. It is important to note that Joint S0 and P0 are set to $\frac{\pi}{2}$ to escape some of the limitations of the Peter Corke's toolbox. It was discovered that it was necessary to add a P0 joint to allow for the vertical link length of L0 between joints S0 and S1. To cope with this, the P0 joint is

frozen so that it has no kinematic bearing on the model. Also, it should be noted that Figure 39 and Figure 40 do not look the same. This is due to the nature of the toolbox, which necessitated a roll joint to be added to the serial link, before a pitch joint. Luckily, the swapping of these joints does not have any effect on the kinematics.

4.3.1 URDF Models

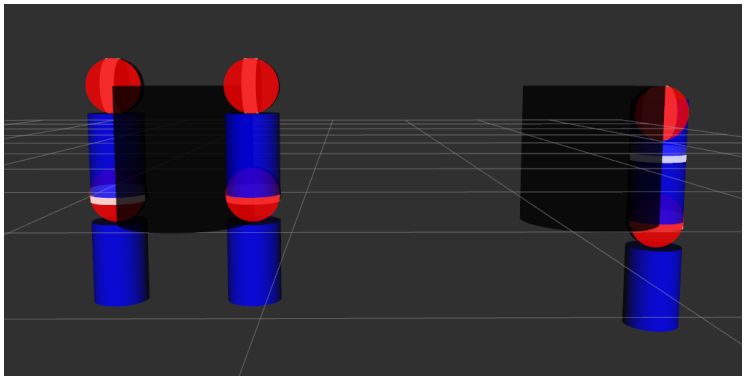

Figure 41. RVIZ Kinematic Arm models (Zeno left, Baxter Right)

To convert XYZ coordinates to joint angles for the robot, it is necessary to derive the inverse kinematics. In previous work [49] the kinematics of a humanoid robot, Zeno were previously derived for teleoperation. This work served as a basis for deriving Baxter's kinematics.

Zeno's arm has four degrees of freedom (DOF) including two DOF at the shoulder (Alpha and Beta), and two at the elbow (Gamma and Theta), as depicted in Figure 42. The Gamma and Theta angles of Zeno are identical to the elbow angles E0 and E1 of Baxter, while the first two degrees of freedom of the Baxter differ from Zeno's.

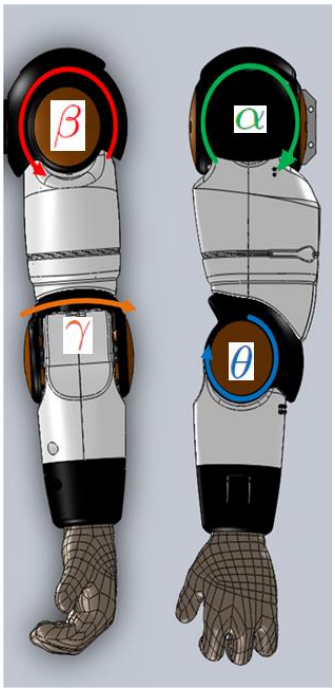

Figure 42. Zeno Kinematic Angles [49]

To validate the inverse kinematics of Baxter, we created a URDF world file of both Baxter and Zeno. This enables us to run both models side-by-side and compare whether the result of the teleoperation is identical. In addition, these models serve as a way to confirm that the information received from ROS OpenPose is adequate for teleoperation. Results of these comparisons are reported in section 5.6.

4.4 Inverse Kinematics

Inverse kinematics enables the Baxter robot to interpret the data from OpenPose and calculate appropriate joint angles so that its pose follows the pose of a human arm. In this section, we present the inverse kinematics calculations allowing imitation teleoperation with the help of the Azure Kinect sensor and OpenPose. Specifically, the inverse kinematics problems addressed is as follows: given the Cartesian positions of the

human arm reported by a depth camera sensor of the Azure Kinect, calculate joint coordinates of a robotic arm mechanism configured like the first four degrees of freedom of a Baxter arm, arranged as joints S0, S1, E0 and E1.

First, the origin and axes from OpenPose must be defined to enable the correct derivation.

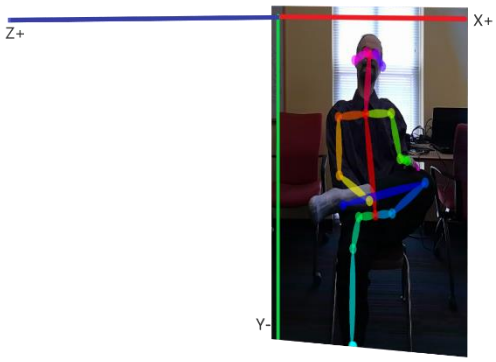

Figure 43. Origin and Axes of the OpenPose

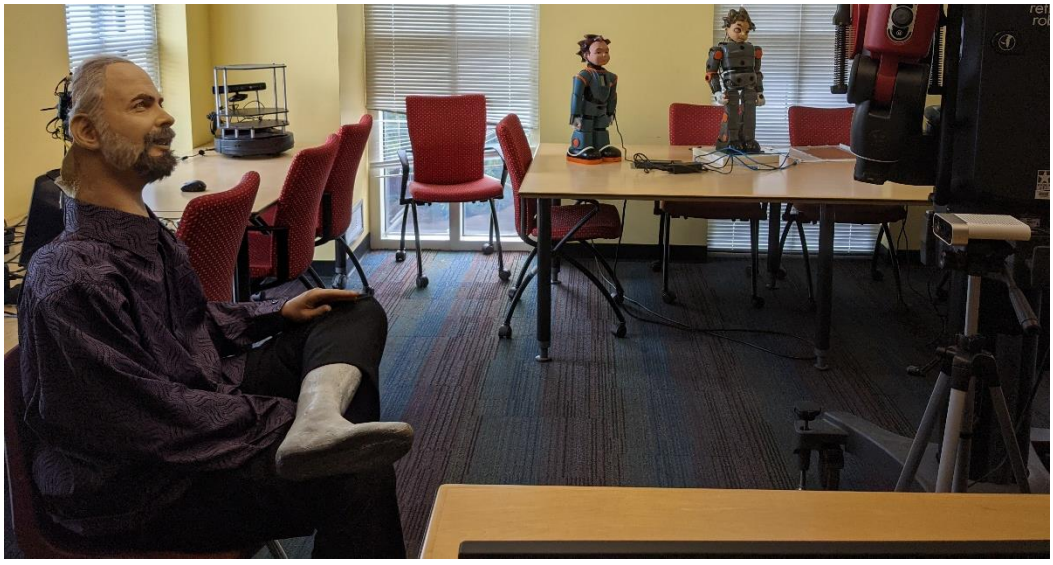

Figure 44. Azure Kinect vs Subject Location. Also depicted in our lab from left to right are robots PKD [17], Milo [13], Zeno [23][50], and Baxter

Using the OpenPose data, we extract the following Cartesian positions of the human wrist, elbow, and shoulder, and represent them as three-dimensional vectors:

$$\text{Position of Hand} = P_{hand} = (x_h, y_h, z_h) \quad (1)$$

$$\text{Position of Elbow} = P_{elbow} = (x_e, y_e, z_e) \quad (2)$$

$$\text{Position of Shoulder} = P_{shoulder} = (x_s, y_s, z_s) \quad (3)$$

Before we can find the joint angles for S0 and S1 (azimuth and elevation respectively), it is necessary to define the subject's shoulder joint as the origin of all other Cartesian positions. Therefore, the following differential vectors are created:

$$V_{se} = \begin{bmatrix} x'_e \\ y'_e \\ z'_e \end{bmatrix} = \begin{bmatrix} x_e - x_s \\ y_e - y_s \\ z_e - z_s \end{bmatrix} \quad (4)$$

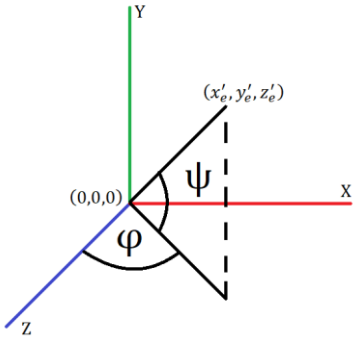

Figure 45. The Azimuth (φ) and Elevation angles (ψ)

Using the vector, the angles azimuth (φ) and elevation (ψ) can be found using the following trigonometric equations:

$$\varphi = atan2(x'_e, z'_e) \quad (5)$$

$$\psi = asin\left(\frac{y'_e}{\sqrt{x'^2_e + y'^2_e + z'^2_e}}\right) \quad (6)$$

This is different from Zeno's α and β angles (shown in the equations (7) and (8) from [49]) due to the different angle between S0 and α.

$$\alpha = atan2(-z'_e, -y'_e) \quad (7)$$

$$\beta = acos\left(\frac{x'_e}{\sqrt{x'^2_e + y'^2_e + z'^2_e}}\right) \quad (8)$$

We can also define V_{es} and V_{he} as the Vectors from the shoulder to the elbow and the hand to the elbow. They are defined as follows:

$$V_{es} = \begin{bmatrix} x_s - x_e \\ y_s - y_e \\ z_s - z_e \end{bmatrix} \quad (9)$$

$$V_{he} = \begin{bmatrix} x_h - x_e \\ y_h - y_e \\ z_h - z_e \end{bmatrix} \quad (10)$$

Before it is possible to derive the elbow angles, we must the define the normal to the plane created by the shoulder, elbow, and hand as n_c (illustrated below) [49]. This can be written as follows:

$$n_c = V_{se} \times V_{eh} \quad (11)$$

Figure 46. Normal to the plane formed by $V_{se} \times V_{eh}$

In order to track the amount this normal has rotated, we must create a rotation matrix $R_{yz}(\varphi, \psi)$ about the axis of rotation. The angles φ and ψ are first in the kinematic chain, which allows us to define the rotation matrix as follows:

$$R_{yz}(\varphi, \psi) = R_y(\varphi) \cdot R_z(\psi) = \begin{bmatrix} c\varphi \cdot c\psi & -c\varphi \cdot s\psi & s\varphi \\ s\psi & c\psi & 0 \\ -s\varphi \cdot c\psi & s\varphi \cdot s\psi & c\varphi \end{bmatrix} \tag{12}$$

Using the default surface normal (in the y direction) allows us to find the surface normal about the upper arm, n_i.

$$n_i = R_{yz}(\varphi, \psi) \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{13}$$

Finding the normal to the joints, n_c, and the normal formed by the axis of rotation, n_i, allows us to find the magnitude of $|\gamma|$, which is the joint angle for E0. This is done by taking the inverse cosine of the dot product between the two-surface normal and dividing by the multiplication of their magnitudes.

$$|\gamma| = cos^{-1}\left(\frac{n_i \cdot n_c}{|n_i||n_c|}\right) \tag{14}$$

Since inverse cosine does not give direction, we use the sgn function of the x component and multiply it by $|\gamma|$ to find the complete joint angle γ.

$$\gamma = |\gamma| * sgn\big(x\ componentof(n_i \times n_c)\big) \tag{15}$$

Finally, to find θ, we use the formula used to find the angle between two vectors. We use the inverse cosine to find a value between 0 and 180 degrees because we assume a person will be unable to bend their arm past their elbow.

$$\theta = cos^{-1}\left(\frac{V_{es} \cdot V_{he}}{|V_{es}||V_{he}|}\right) \tag{16}$$

4.5 Finding Joint Offsets

Finding joint offsets is necessary to match the initial pose of the user to the initial pose of the robot. This must only be done once per robot, after which time the offsets will match all subjects. To tune these offsets, we first ask the user to place their arms to their sides. This allows us to ensure that the φ, ψ, γ, and θ all yield a zero position that forces the robot to also have its arms to its sides. Following this, we ask the user to raise both hands above their heads to ensure that robot mimics their behavior. It was noticed that the S1 joint moved downward when given positive values and upward when given negative values. Therefore, it was necessary to invert the ψ angle by multiplying it by negative one. Next, the subject is asked to hold their arms in a "T" shape, which allows us to check that the γ angle was tuned correctly. Finally, a position where the subject held their

arms to their sides in a “T” shape, but bent at the elbows at 45 degrees up to their head allows us to check the θ angle.

4.6 Controller and Gain Tuning

Rethink Robotics provides several different control modes for using Baxter’s arms. These include the ability to provide joint positions, velocity, or torque values in order to move the arms. We use two versions of the Joint Position Controller which allows us to provide angles in radians to the robot. The normal joint controller features the ability to do collision avoidance, joint limit clipping, and collision detection. As well as provide Delta Scaling, Velocity Scaling, and High-Speed Scaling in order to ensure all joints achieve the desired pose at the same time (as seen in Figure 47) [51]. The various gains used by the controller are tuned from the factory, with spring and torque offsets being calibrated once a month using the arm calibration routine [54].

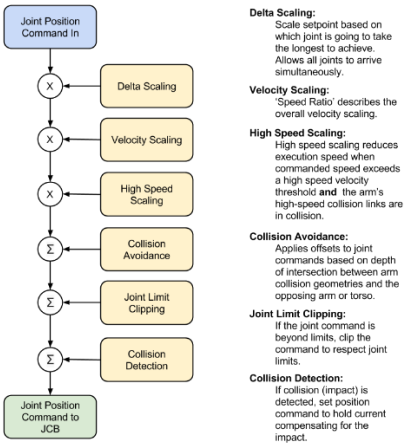

Figure 47. Baxter’s Joint Position Controller [52]

The Raw Joint Position Controller removes the safety systems while controlling the arms. The controller only allows for Delta Max Clipping, Joint Limit Clipping, and Collision Detection. It has been observed that the lack of Delta, Velocity, and High-Speed Scaling enables the robot to move its arms faster. However, the cost of this mode is the lack of a collision avoidance system built into the controller and the arms move more erratically causing some noise due to the violence of the motion [51].

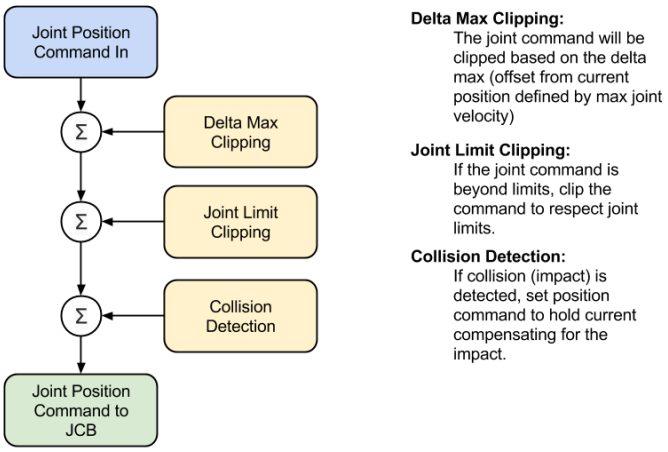

Figure 48. Baxter's Raw Joint Position Controller [53]

The OpenPose data was found to also produce noise during tracking of the human subject. Therefore, an Infinite Impulse Response filter applied to the output of the inverse kinematics and offsets. This decreased the peaks and valleys that were occurring due to these errors. The equation of this is shown below:

$$Angle_{output} = 0.6 * Angle_{current} + 0.4 * Angle_{previous} \quad (17)$$

4.7 Results

4.7.1 Simulated Results

In this section, we compare the resulting positions of the elbow of Baxter and Zeno kinematics to check the validity of our derived kinematic model. We asked a human subject to wave their hand, and then we calculate the joint angles of an arm configured as Baxter kinematics, as well as Zeno kinematic model represented by the URDF models of Figure 41. We then compare the calculated elbow position with forward kinematics of the two URDF robots after accounting for appropriate joint offsets. The acquired data yielded two datasets of 342 points of the subject waving (sampled at a frequency of 10hz). These two datasets were subtracted from each other, and then the difference was plotted corresponding to the difference in X, Y and Z.

As seen in Figure 49, the error stayed close to zero during operation. However, there were two instances where the error shot rose to around +/-0.688. It is unclear as to what caused this; more than likely some sort of error with motion tracking (because this only occurred for one point at a time).

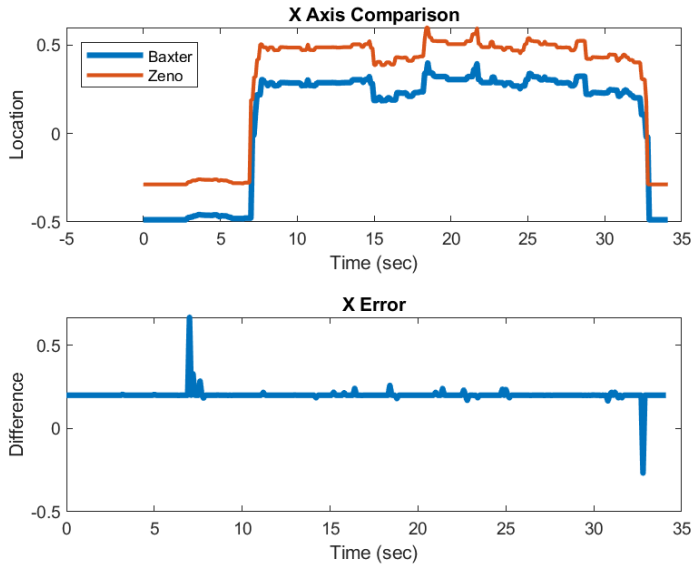

Figure 49. X Axis Actual and Error

A similar trend is seen in the Y error shown in Figure 50 where the error is very close to zero. In this figure, there are more frequent disturbances from the 0 crossing. This may be due to some issues with tuning Zeno's offsets. The large spikes occur in a similar place as to those in Figure 49. Therefore, we believe this was an error in motion tracking.

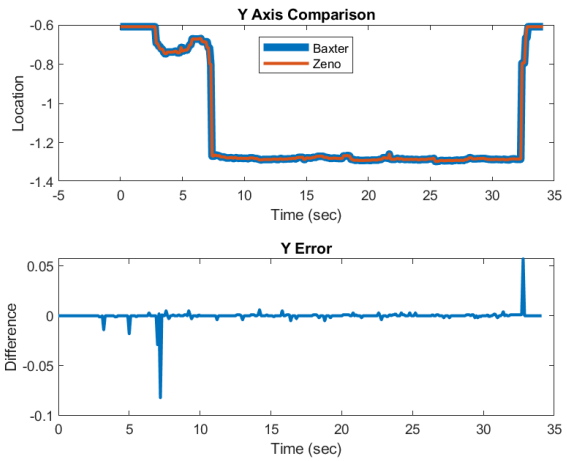

Figure 50. Y Axis Actual and Error

A similar trend is seen in the Z error shown in Figure 51; where the error is very close to zero. This may be due to some issues with tuning Zeno's offsets. The large spikes occur in a similar place as to those in Figure 49. Therefore, we believe this was an error in motion tracking.

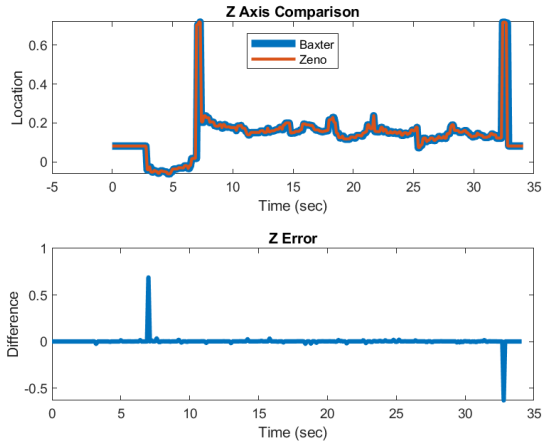

Figure 51. Z Axis Actual and Error

4.7.2 Experimental Results

In the section above, it has been validated that the derived inverse kinematics are almost identical to the desired result. Therefore, it is possible to use the real Baxter to test the results. To validate that the Baxter robot and the human have achieved the correct pose, we must view the trajectories of both the human and the robot. We view each joint and the inverse kinematics compared to the actual achieved joint positions to confirm the validity of the control scheme. In this case, we asked the subject to perform a hand wave using each arm (first right, then left, with the unused hand by the subject's side) and recorded the image data into a file. This file was used to supply images to the ROS OpenPose software which then allowed the rest of the system to function as if the motion was occurring live. The file was replayed three times per controller in order to ensure that the robot would have sufficient time to exit its initial position and use produce a movement. The resulting data that was produced by the robot was recorded into a comma-separated value file for later analysis.

In our first experiment we use the normal Joint Position Controller and observe the results. Figure 52, shows that the S0 joints on both sides were constant. This joint corresponds with the azimuth, or angle, which does not move during a handwave. It is important to note that the robot does not start achieving the desired angle immediately. This is because Baxter begins in the untucked position, which is illustrated in Figure 53.

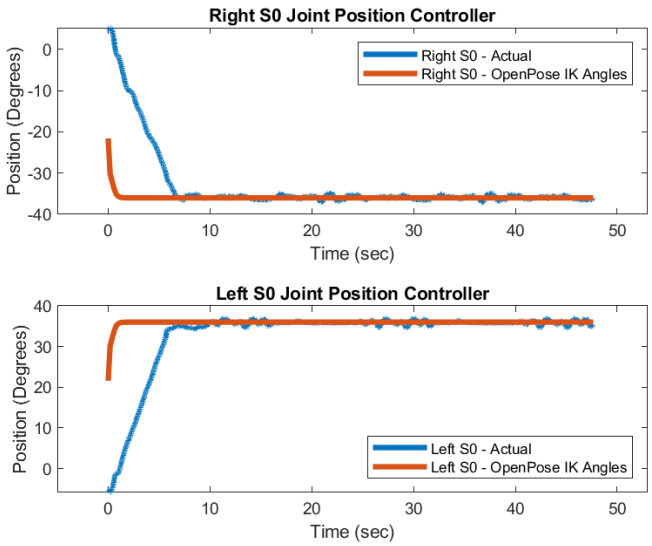

Figure 52. Left and Right S0 Joint Angles (OpenPose desired vs Baxter Position) for Joint Position Controller

Figure 53. Baxter Untucked Position 97[55]

Figure 54 shows the use of the Raw Joint Position controller during the same handwave experiment. It can be observed that there is more error in this type of controller, due to the speed of the end-effector during the experiment. The high velocity

causes the S0 joint to produce more error due to it moving around from the other joint movements.

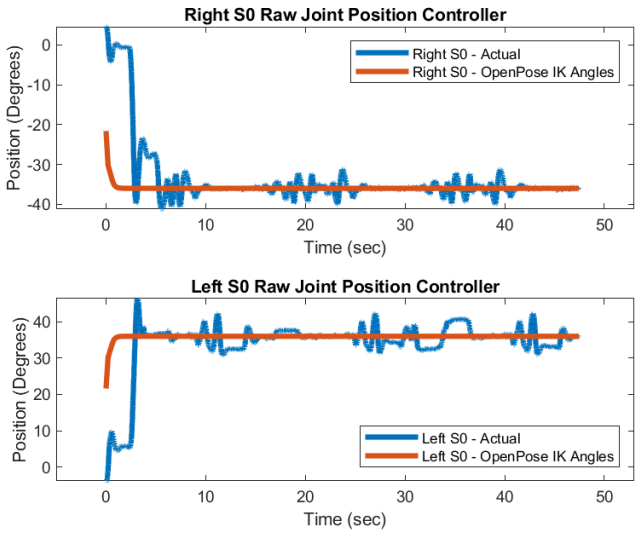

Figure 54. Left and Right S0 Joint Angles (OpenPose desired vs Baxter Position) for Raw Position Controller

The second DOF is the S1 joint (the elevation angle), which corresponds to ψ. This angle is expected to change throughout each handwave (because the unused hand is by the subject's side). The right S1 in Figure 55 confirms that the first arm to be used was the right. This can be observed due to the sharp change from the constant position 10 degrees to approximately 80 degrees. The small oscillations at the plateau of the OpenPose desired angles are where the hand is being waved. The move back to 10 degrees is the time when the left is being used. It can be observed that the right S1 joint, does not perform closely to the desired position acquired from OpenPose. This is most likely due to the Joint Position

Controller's speed limiting, which prevents the robot from achieving the full pose, before the next hand is used.

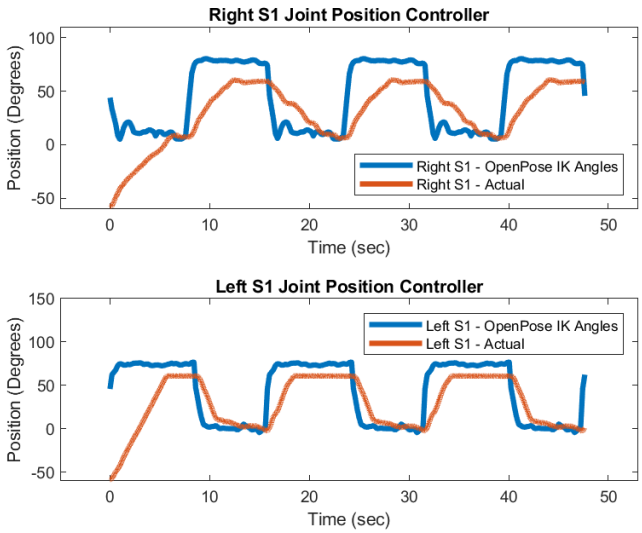

Figure 55. Left and Right S1 Joint Angles (OpenPose desired vs Baxter Position)

The speed limiting effect of the Joint Position Controller is not observed when using the Raw Joint Position Controller. In Figure 56, the error is much smaller than in Figure 55, and the actual versus desired graphs show that both plateau around the same point. Some minor error is observed, most likely caused by the gains of the robot controller and of course the velocity of the other joints causing oscillations in the lower DOF joints.

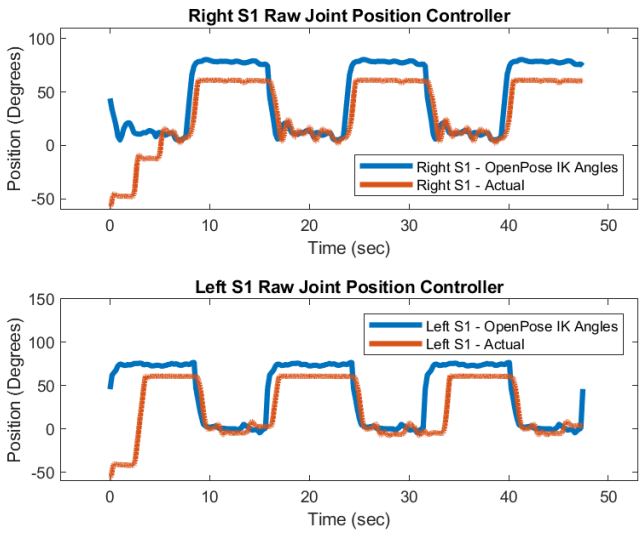

Figure 56. Left and Right S1 Joint Angles (OpenPose desired vs Baxter Position) for Raw Position Controller

The third DOF is the E0 or joint illustrated in Figure 57. It is expected that, during a handwave, the position will be greater than 180 degrees (to allow for the E1 angle to point up. During the right handwave, we see very high oscillations, however, it still stays above 180 degrees during the motion. This is most likely caused by using arcsin to compute this, since this function only has a domain between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$. During the right handwave, the left gamma angle hovers around 100 degrees. This is most likely an error in the offset tuning.

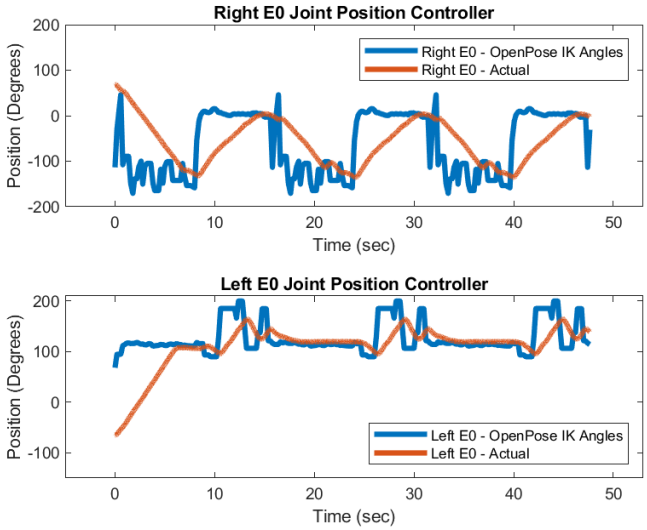

Figure 57. Left and Right E0 Joint Angles (OpenPose desired vs Baxter Position)

During the Raw Position Controller test, the E0 joint was able to follow the desired OpenPose angles closely when compared with the normal Joint Controller (Figure 58).

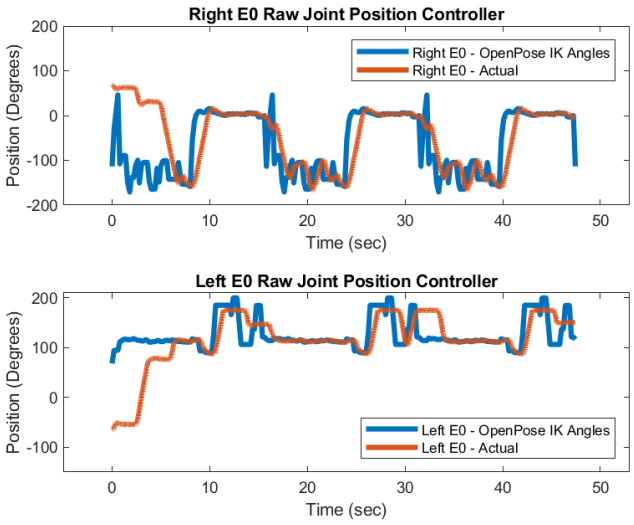

Figure 58. Left and Right E0 Joint Angles (OpenPose desired vs Baxter Position) for Raw Position Controller

Figure 59 illustrates the movement of the of the E1 or joint during the hand wave. The large oscillations are the points where the subject is performing the motion. The actual position of the robot in this figure, shows that it was unable to achieve the pose that was given to it.

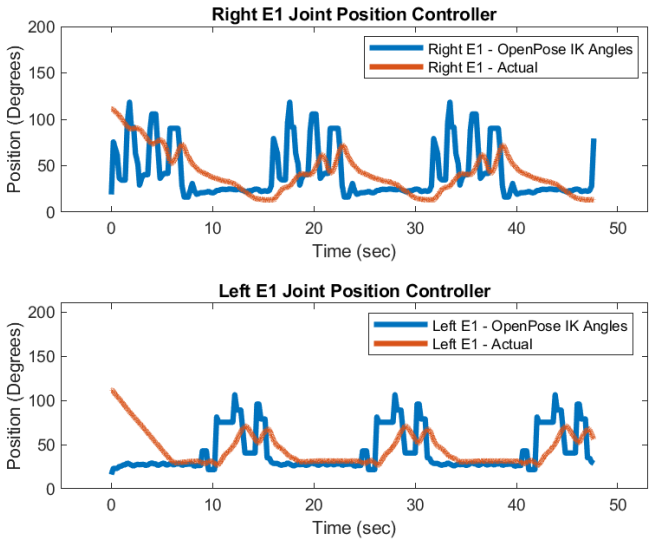

Figure 59. Left and Right E1 Joint Angles (OpenPose desired vs Baxter Position)

The Raw Position Controller (as seen in Figure 60) was able to achieve its desired position for both hands. It should be noted that the left OpenPose desired angles do not show as much oscillation as the right hand OpenPose desired angles. This is due to error in the inverse kinematics producing errors due to loss in motion capture. The subject must have performed the motion at a different speed for the left hand when the data was being recorded.

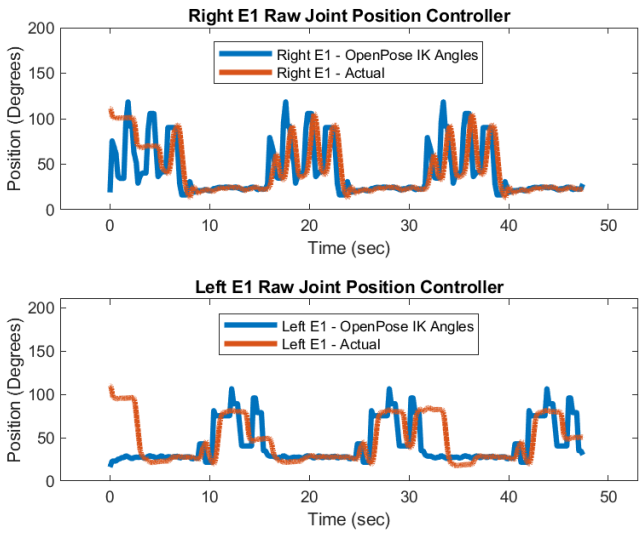

Figure 60. Left and Right E1 Joint Angles (OpenPose desired vs Baxter Position) for Raw Position Controller

Finally, Figure 61 illustrates the wrist angles for both arms, which are not currently implemented in the current control scheme. Therefore, the desired position will always be zero degrees, and the actual will also achieve the zero position. We see that, during the first few seconds, the actual graphs all move from some arbitrary value to zero. This is due to the location of these joints during the untucked position.

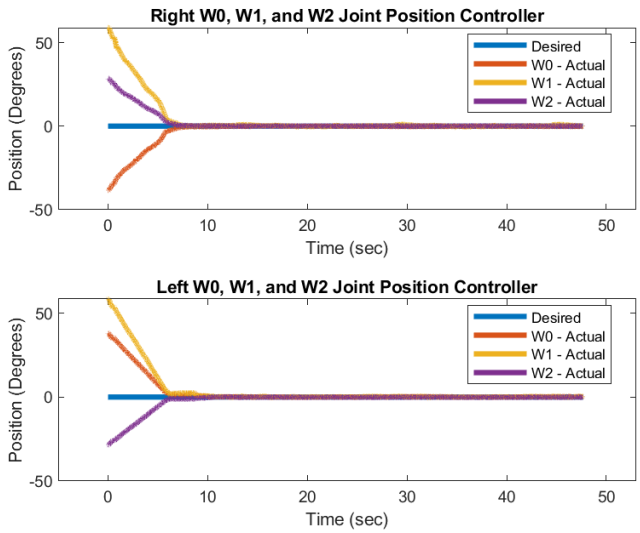

Figure 61. Left and Right Wrist Joint Angles (Openpose desired vs Baxter Position)

The Raw Joint Position Controller performs similarly to the normal Joint Position Controller (Figure 62). However, the actual joint position for all three joints on both end effectors show small oscillations during the movement. This is due to the velocity caused by the movement of the first four DOF, which causes oscillations in the rest of the joints in the kinematic chain.

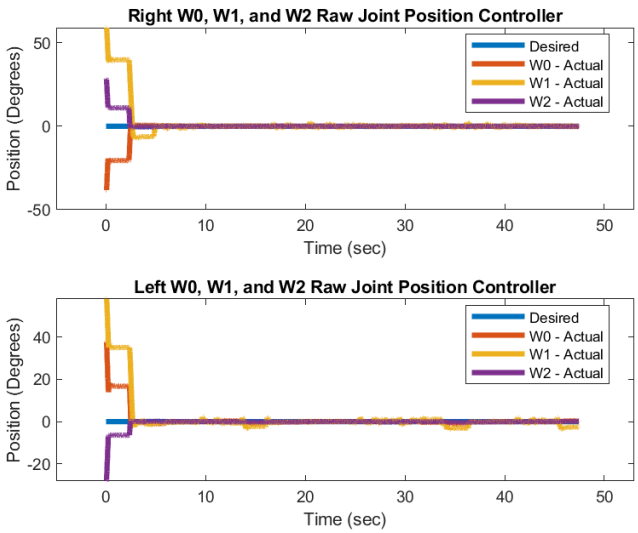

Figure 62. Left and Right Wrist Joint Angles (OpenPose desired vs Baxter Position) for Raw Position Controller

CHAPTER 5

SEGMENT-BASED ONLINE DYNAMIC TIME WARPING

Motion quality estimation is necessary for enabling a computer to "understand" the complexity of human movement and how it can be impaired. By giving a score for a motion it is possible to create a measure for diagnosing conditions such as stroke [56], cerebral palsy [57], spinal cord injuries [58], Parkinson's disease [59], and autism [23]. Pairing an algorithm designed to analyze two signals with a robot's ability to follow the same trajectory for hours at a times allows for the ability to teach a patient a motion through repetition.

Dynamic Time Warping (DTW) is one of several algorithms that enables the comparison between two-time signals; others include algorithm such as Euclidean distance or the nearest neighbor distance. In our previous work, Wijayasinghe and colleagues [23] recorded sequences from hand joint angles of autistic and neuro-typical subjects while they imitated the upper arm motions of social robot Zeno. In this work, we build upon the DTW algorithm and improve on our previous study, allowing us to give subjects a motion-quality score in real time, and adapt the robot to a similar speed of the human. This is important to allow the next-generation autism robot to track the progress of a subject.

5.1 Segment-based Online Dynamic Time-Warping Algorithm

Segment-based Online Dynamic Time Warping (SODTW) builds upon the Dynamic Time Warping algorithm by adding the segmentation of a reference sequence into a lookup table for easy comparison of the measured sequence. It is then applied to a cyclic motion to ensure that the best cost can be calculated while the subject performs the motion. This allows the algorithm to "understand" the subjects best case capabilities, which improves its use as a diagnostic tool.

The segmentation behavior is shown in Figure 63, here we see that the reference sequence is split into several segments for an entire period. Segments from the measured sequence are compared to every reference segment (using DTW) in order to find what we call the start point of both sequences. In Figure 63, the measured sequence becomes cyclic at around the 250 seconds marker; this informs the algorithm that the subject has begun performing the motion. The start point is then used as the beginning of the time sequence to compare against the reference sequence. In addition, the time taken from the beginning of the robot performing the motion (at time zero) and the time the subject begins to perform the motion (at time 250) can be called the reaction time.

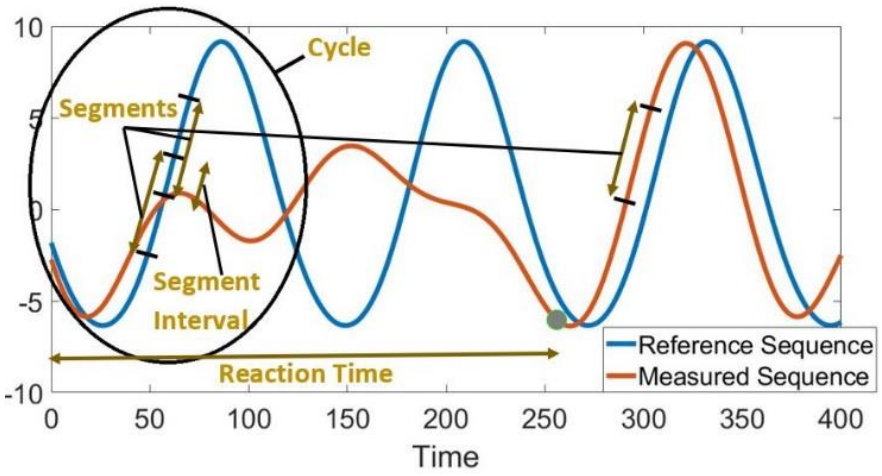

Figure 63. SODTW applied to a Cyclic Sequence

To use DTW on segments and the full trajectory sequence, we must define the DTW cost function (18). Given two discrete time sequences X and Y with the lengths m and n respectively, allows us to define DTW cost as:

$$D(i,j) = d(x_i, y_j) + min[D(i-1, j-1), D(i-1, j), D(i, j-1)] \qquad (18)$$

Where d(x,y) is the Euclidean distance between x and y, i=1,…,m, j=1,…,n, and D(m,n) is the DTW cost. When the smallest DTW cost is found we save the index of the reference segment and the measured sequence, which are called our start points. For the measured sequence, all the points starting from the start points until the length of one cycle is collected. This collected data is compared against the start point of the reference segment for the cycle length, using DTW; the computed DTW cost is compared with the previous smallest cost, and, if the new one is smaller, it is now called the best SODTW cost (the start point is also stored as a measure of reaction time). The algorithm and be summarized as follows.

Algorithm 1

Part 1: Initialization

1. Nseg = Number of segments for one cycle
2. SegLen=Length of each Segment
3. segInter= Segment Interval
4. Time= Elapsed time/measured sequence indices
5. Q= One dimensional reference trajectory with fixed length
6. S= One dimensional measured streaming data sequence (of changing length)
7. cycleLen = Length of one cycle sequence calculated as (Nseg-1)*segInter+segLen
8. DTW_O = A large default number
9. DTW_N = 0
10. Error_Tolerance = Error tolerance for SODTW cost

Part 2: Dividing one cycle of reference trajectory into segments

11. Seg = Segments created from Q (Seg is a matrix with Nseg row and segLen column)

Part 3: Finding the segment index and DTW cost between segments on reference and measured sequences

12. SWindow = Segment on measured trajectory, S, with the same length of the segLen
13. for i = 1 to Nseg
14. SEG = Seg(i,:)
15. DTW_S_W(i) = D(SEG, SWindow) (Equation (1))
16. end for
17. I = Index of the segment on reference trajectory for the minimum of DTW_S_W
18. D = Minimum DTW cost in DTW_S_W
19. T = Time/index on the measured trajectory

Part 4: DTW cost calculation for one cycle

20. Scycle = Cycle on S started from T and included Nseg segments
21. Qcycle = Cycle on Q started from I and included Nseg segments
22. DTW_N = D(Qcycle, Scycle) (Equation (1))
23. if DTW_N <= DTW_O – Error_Tolerance

24. DTW_O = DTW_N
25. Reaction_Time = T
26. end if

5.2 Validation with Human Subjects

To validate the SODTW, we used a pre-collected dataset of 55 physically healthy subjects over the age of 18 [23]. The motion recordings of the subjects were analyzed offline using our SODTW algorithm. We recruited an additional 13 subjects to perform the study to test whether the algorithm works in an online setting.

5.2.1 Experimental Setup

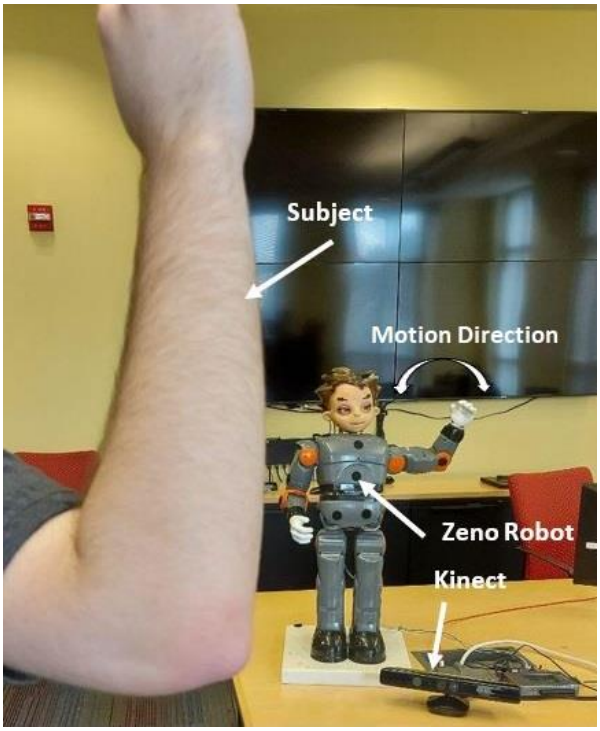

Figure 64. Experimental Setup

The physical setup of the experiment was the same in the pre-collected and real-time experiments. Social Robot Zeno was placed in front of a subject who was asked to

mimic the motions of the robot. An Xbox 360 Kinect (using the Kinesthesia toolkit) would be used to collect 3D data of the subject. Zeno, connected to an NI myRio controller, was programmed to execute a right handwave. Both studies used the same right hand recording file in order to ensure that the data can be used together.

To simulate a patient who was motor-impaired in some way, the subject was asked to use weights while performing the motion. Both studies created comma separated value (CSV) files that could be used for later analysis. The weights used were none, 5, and 15lb dumbbells that were held in the subject's hand during the duration of the motion.

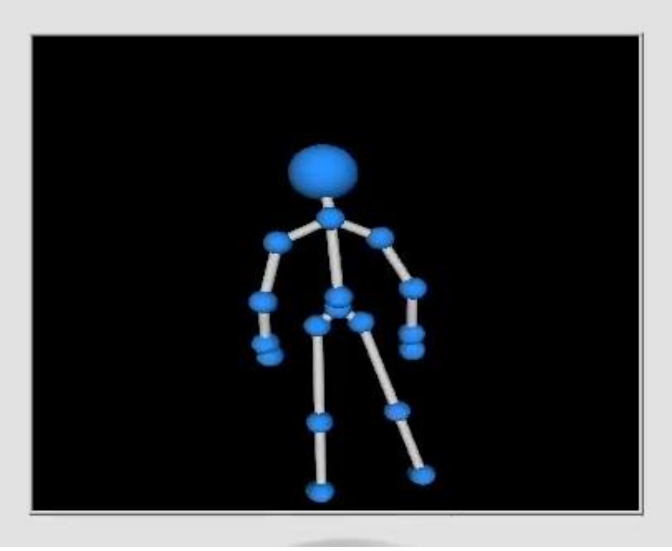

Figure 65. Sample Output of the Kinesthesia Toolkit

To create the measured sequence the Kinesthesia toolkit was used to get XYZ coordinates of the humans pose. These XYZ coordinates are converted to joint angles using the inverse kinematics for Zeno. In order to perform SODTW, we need to have a

cyclic motion. In a handwave, the cyclic joint is the θ angle, therefore its time sequence is analyzed by the algorithm.

5.2.2 Offline Validation

For the offline validation of our algorithm the joint angle data from the 55 previously recorded subjects was input into a MATLAB implementation of SODTW. This data had two conditions that were of interest: first, the performance of the motion without holding any weight, and a second, where the motion was performed while holding a 15lb weight. Only 44 of the 55 subjects had data for the 15lb weight, however, it should still be enough to validate our algorithm.

The first step of the algorithm necessitates the initialization of several constants, (as seen in Table 8). cycleLen is derived by taking the reference trajectory and finding the length of each cycle. The segLen and segInter are chosen and these allow for the Nseg to be derived.

Table 8 SODTW Parameter Initialization

SODTW Parameter	Values
Nseg	52
segLen	6
segInter	1
cycleLen	57

Figure 66 shows the SODTW cost of the 55 subjects that performed the motion without a weight. The average SODTW cost for these 55 subjects is $\mu=5.85$ with minimum $m=2.03$, maximum $M=27.1$ and standard deviation $\sigma=4.7$. In order to prove

that the SODTW algorithm is able to differentiate between our two impairment cases, it is necessary to compare the impairment model.

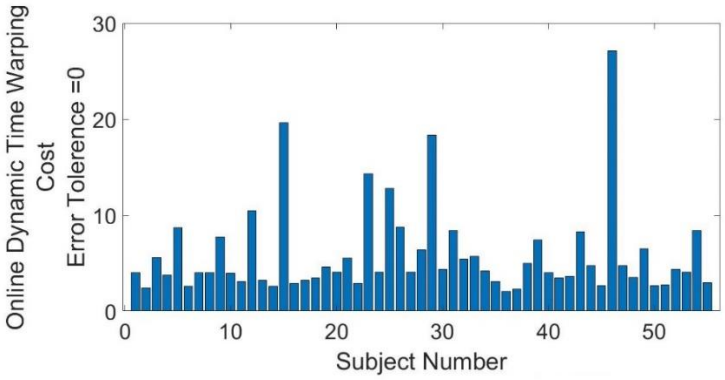

Figure 66. SODTW cost for 55 subjects with no weight

In Figure 67 we see the theta angle recording of a subject who performed the motion with, and without a 15lb weight. When the subject is unimpaired the SODTW=3.22; when the subject is asked to perform the same motion with the weight, the SODTW cost rises to 9.98. Showing that the SODTW algorithm can classify impaired versus unimpaired subjects.

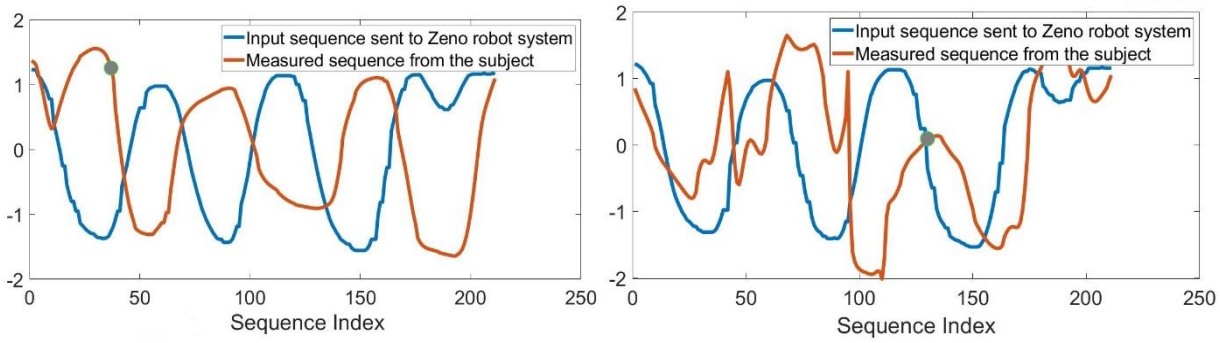

Figure 67. Sequence from a subject performing the motion with (right) and without (left) a weight

5.2.3 Online Validation

To show that the SODTW can perform in real time, we asked 13 subjects to perform in a study where they performed a handwave. The average SODTW cost for the subjects without the 15lb weight was 7.23 and increased to 9.5 when holding the weight. We hypothesize that the higher-than-average SODTW cost without weight had to do with subject confusion of the correct motion to perform. Due to a later discovered faulty motor, that may have caused the robots handwave to look incorrect during the experiment. This would have caused the subject to be confused on how to execute the motion.

5.3 Adaptive Imitation

The SODTW algorithm also enables us to adapt the robot to the speed of the subject. In the future, this could be used to teach the subject a motion. To do this, the reference trajectory was used to create two other trajectories (fast and slow). By under-sampling or oversampling the original trajectory, it was possible to slow/speed up the robot by either ½x or 2x the speed of the original. These new trajectories were also used in the SODTW to evaluate the performance of the user. After a period, the robot would identify the smallest SODTW score and switch to the associated reference trajectory. For instance, if the subject were asked to wave their hand quickly, the SODTW score for this motion would be close to or below 5, while the others would be higher. This would cause the robot to identify the fast reference trajectory as the best suited. Therefore, it would adapt and begin using the new reference trajectory as the new output to the motor commands. This behavior would also be applicable to both the normal and slow motions.

Figure 68 shows the SODTW scores for each of the 13 subjects during the adaptation experiment. In the right image, the subject was asked to perform the motion quickly. It can be seen that the SODTW score showed for the fast reference trajectory was lower than the normal reference trajectory for all subjects. In the left image, the subject was asked to perform the motion slowly. The SODTW score for the slow reference trajectory was lower than the normal reference trajectory for all subjects. Therefore, the SODTW algorithm enables the capability to recognize the ability of and adapt to the needs of the subject. In the future, this can enable the ability to teach a motion by slowing the robot to a speed that a subject can follow.

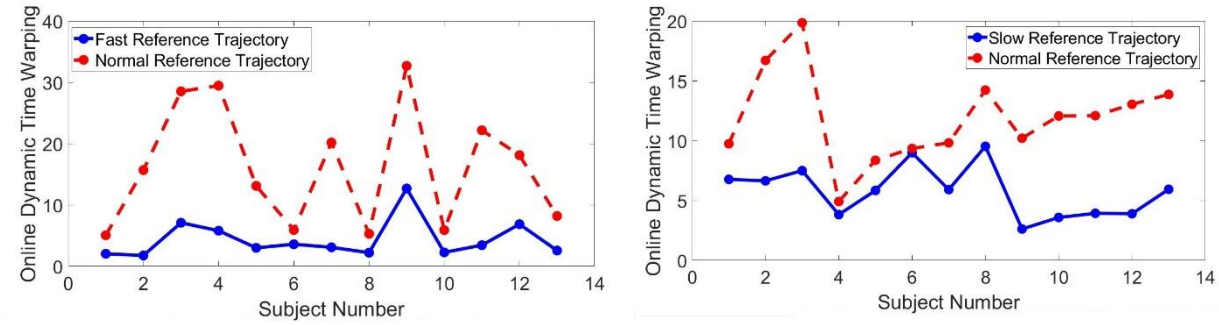

Figure 68. The performance of all 13 subjects during the Adaptation Experiment

CHAPTER 6

CONCLUSION AND FUTURE WORK

This thesis focused on developing subsystems for the next-generation autism robot, in order to facilitate its ability to help individuals with ASD. We explored the usage of a physiological wristband in classifying attentiveness and controlling a robot in real time. Teleoperation was also explored to inform a robot about the motion of a subject. Finally, we developed an algorithm that enables the evaluation of motion quality of a subject in order to either diagnose a subject or adapt the robot on the fly.

6.1 Conclusion

The physiological wristband, Empatica E4, is a wearable device that collects measurements in real time. Using this ability, and LabVIEW it is possible to extract features for emotion classification and move a robot based on accelerometer data. In addition, we were able to explore the physiological data from a social skills camp at the University of Louisville Autism Center to attempt to classify the attentiveness of a subject. The proposed neural network classifiers in this thesis achieved a 68% accuracy.

Gaining insight into the methods the wearable uses to gather data enables us to examine its feasibility for future use in studies with subject with ASD. This knowledge ensures that the next-generation autism robot will have a method of interpreting the

subject's attentiveness, and in future, their emotions. This information can be used to maximize the effectiveness of therapy and ensure that the subject feels safe.

Understanding motion is also an important part of the next-generation autism robot's abilities. Being able to teleoperate (or puppeteer) the robot enables the therapist to record motions for use in a session. By utilizing OpenPose as the vision subsystem for the robot, we can track the face, fingers, and whole body of a subject. This allows more information about the subject to be collected and will allow the robot to understand more types of gestures (such as ones that involve fingers) or facial expressions. Using Rethink Robotics' Baxter robot has shown that the teleoperation subsystem can be implemented on any ROS based robot. If the camera (Azure Kinect) system can communicate and provide web camera image to OpenPose the work is transferable.

Finally using skeletal tracking software on a robot enables us to develop algorithms that evaluate the motion quality of a subject. We proposed the Segment-based Online Dynamic Time Warping algorithm to gauge the best-case capabilities of the subject. This enables the robot to grade a subject on their motion performance and adapt to the motion of the subject. Adapting to the motion quality can help the subject learn, the robot can change its speed to match the capabilities of the subject.

By providing the ability to understand the motion of the subject, and adapt to their performance, it is possible to create a robot that can both diagnose and teach motions. It also enables the robot to mimic the motion of the subject which can help the subject stay engaged. Finally, by generating a score based upon the motion quality, it is possible for the robot to give the subject praise or reassurances based on threshold scores.

Informing the next-generation autism robot about the emotion and motion of the subject will enable a closed loop human-robot interaction. By monitoring the physiological signals from the wearable wristband, the robot could prompt the subject if they are inattentive or attempt to calm them if frustration is detected. Informing the robot about the motion of the subject ensures that it understands the movements. This signal data ensures that the robot is capable to adapt to the needs of its user.

6.2 Future Work

In the future, the subsystems developed in this thesis can be combined to help children with ASD. By improving upon the emotion classification and receiving the physiological signals in real time, it is possible to ensure that a robot can understand the emotional state of a subject. In addition, this technology can be useful for therapists and that oversee multiple students with ASD at once. By helping inform the therapist of the emotional state of all the students.

The teleoperation submodule of this thesis can be used to teleoperate different robots to conduct a study on which type of robot is more effective at keeping engagement of a child with ASD. The teleoperation portion also enables the use of finger and facial feature tracking. Therefore, it is possible to ensure that the next-generation autism robot can be knowledgeable about the face of the subject. Finally, it is necessary to investigate the speed of the current controller in order to ensure that the robot can function in real time.

When paired with a motion generation framework, such as Dynamic Movement Primitives, the SODTW algorithm can be used to help the robot teach a subject a desired

motion. This will enable the next-generation autism robot to adapt to the subject and help achieve mastery of various movements.

REFERENCES

[1] "Data & Statistics on Autism Spectrum Disorder," *Centers for Disease Control and Prevention*. Centers for Disease Control and Prevention, Sep-2020 [Online]. Available: https://www.cdc.gov/ncbddd/autism/data.html

[2] "Autism Spectrum Disorder," *National Institute of Mental Health*. U.S. Department of Health and Human Services [Online]. Available: https://www.nimh.nih.gov/health/topics/autism-spectrum-disorders-asd/

[3] "Research & Diagnostic Instruments," *NewYork-Presbyterian*. [Online]. Available: https://www.nyp.org/psychiatry/center-for-autism-the-developing-brain/research-diagnostic-instruments/ados-diagnostic-instrument#: :text=Autism Diagnostic Observation Schedule (ADOS)&text=The ADOS is a semi,or an autistic spectrum disorders.

[4] NSF Award Search: Award # 1838808 - SCH: INT: Adaptive Partnership for the Robotic Treatment of Autism. [Online]. Available: https://www.nsf.gov/awardsearch/showAward?AWD_ID=1838808

[5] *NSF Award Search: Award # 1849213 - RII Track-1: Kentucky Advanced Manufacturing Partnership for Enhanced Robotics and Structures*. [Online]. Available: https://www.nsf.gov/awardsearch/showAward?AWD_ID=1849213

[6] "Treatment and Intervention Services for Autism Spectrum Disorder," *Centers for Disease Control and Prevention*. Centers for Disease Control and Prevention, Sep-2019 [Online]. Available: https://www.cdc.gov/ncbddd/autism/treatment.html

[7] B. Scassellati, H. Admoni, and M. Matarić, "Robots for use in autism research.," *Annual review of biomedical engineering*, vol. 14, pp. 275–294, 2012, doi: 10.1146/annurev-bioeng-071811-150036.

[8] N. J. Rinehart, J. L. Bradshaw, A. V. Brereton, and B. J. Tonge, "Movement preparation in high-functioning autism and Asperger disorder: a serial choice reaction time task involving motor reprogramming.," *Journal of autism and developmental disorders*, vol. 31, no. 1, pp. 79–88, Feb. 2001, doi: 10.1023/a:1005617831035.

[9] G. K. Todorova, R. E. M. Hatton, and F. E. Pollick, "Biological motion perception in autism spectrum disorder: a meta-analysis.," *Molecular autism*, vol. 10, p. 49, 2019, doi: 10.1186/s13229-019-0299-8.

[10] B. Robins, K. Dautenhahn, and J. Dubowski, “Does appearance matter in the interaction of children with autism with a humanoid robot? Interact Stud,” *Interaction Studies*, vol. 7, Nov. 2006, doi: 10.1075/is.7.3.16rob.

[11] B. Robins, K. Dautenhahn, and P. Dickerson, “From Isolation to Communication: A Case Study Evaluation of Robot Assisted Play for Children with Autism with a Minimally Expressive Humanoid Robot,” in *2009 Second International Conferences on Advances in Computer-Human Interactions*, Cancun, Mexico, 1-7 Feb. 2009, pp. 205–211, doi: 10.1109/ACHI.2009.32.

[12] VinÃ, “Mirroring and recognizing emotions through facial expressions for a RoboKind platform,” in *2017 IEEE 5th Portuguese Meeting on Bioengineering (ENBENG)*, Coimbra, Portugal, 16-18 Feb. 2017, pp. 1–4, doi: 10.1109/ENBENG.2017.7889480.

[13] “Meet Milo!” [Online]. Available: https://www.robokind.com/robots4autism/meet-milohttps://www.robokind.com/robots4autism/meet-milo

[14] A. Kroiss, D. Sonego, and P. Rollins, “Using a Humanoid Robot as a Co-therapist withChildren with ASD,” 2016.

[15] J. Fasola and M. J. Mataric, “Using Socially Assistive Human-Robot Interaction to Motivate Physical Exercise for Older Adults,” *Proceedings of the IEEE*, vol. 100, no. 8, pp. 2512–2526, Aug. 2012, doi: 10.1109/JPROC.2012.2200539.

[16] Y. Bian, L. Zhao, H. Li, G. Yang, L. Geng, and X. Deng, “Research on Multi-modal Human-Machine Interface for Aerospace Robot,” in *2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics*, Hangzhou, China, 26-27 Aug. 2015, vol. 1, pp. 535–538, doi: 10.1109/IHMSC.2015.74.

[17] A. Habib, S. K. Das, I.-C. Bogdan, D. Hanson, and D. O. Popa, “Learning human-like facial expressions for Android Phillip K. Dick,” in *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, New Taipei, Taiwan, 18-22 Aug. 2014, pp. 1159–1165, doi: 10.1109/CoASE.2014.6899473.

[18] A. Rahim, A. Sagheer, K. Nadeem, M. N. Dar, A. Rahim, and U. Akram, “Emotion Charting Using Real-time Monitoring of Physiological Signals,” in *2019 International Conference on Robotics and Automation in Industry (ICRAI)*, Rawalpindi, Pakistan, 21-22 Oct. 2019, pp. 1–5, doi: 10.1109/ICRAI47710.2019.8967398.

[19] O. AlZoubi, S. K. D’Mello, and R. A. Calvo, “Detecting Naturalistic Expressions of Nonbasic Affect Using Physiological Signals,” *IEEE Transactions on Affective Computing*, vol. 3, no. 3, pp. 298–310, July-September 2012, doi: 10.1109/T-AFFC.2012.4.

[20] M. Saadatzi, F. Tafazzoli, K. Welch, and J. Graham, “EmotiGO: Bluetooth-enabled Eyewear for Unobtrusive Physiology-based Emotion Recognition,” 2016, doi: 10.13140/RG.2.2.19472.40964.

[21] Z. Jiang, L. Lu, X. Huang, and C. Tan, "Design of wearable home health care system with emotion recognition function," in *2011 International Conference on Electrical and Control Engineering*, Yichang, China, 16-18 Sept. 2011, pp. 2995–2998, doi: 10.1109/ICECENG.2011.6057832.

[22] A. Zunino et al., "Video Gesture Analysis for Autism Spectrum Disorder Detection," in *2018 24th International Conference on Pattern Recognition (ICPR)*, Beijing, China, 20-24 Aug. 2018, pp. 3421–3426, doi: 10.1109/ICPR.2018.8545095.

[23] I. B. Wijayasinghe, I. Ranatunga, N. Balakrishnan, N. Bugnariu, and D. O. Popa, "Human-Robot Gesture Analysis for Objective Assessment of Autism Spectrum Disorder," *International Journal of Social Robotics*, vol. 8, no. 5, pp. 695–707, 2016, doi: 10.1007/s12369-016-0379-2. [Online]. Available: https://doi.org/10.1007/s12369-016-0379-2

[24] K. C. Yang, C. H. Huang, and C. F. Le, "Applying microsoft kinect for windows to develop a Stroke Rehabilitation System," in *2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, Bali, Indonesia, 4-7 Dec. 2016, pp. 1923–1927, doi: 10.1109/IEEM.2016.7798213.

[25] W. Wei, C. McElroy, and S. Dey, "Towards On-Demand Virtual Physical Therapist: Machine Learning-Based Patient Action Understanding, Assessment and Task Recommendation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 9, pp. 1824–1835, Sept. 2019, doi: 10.1109/TNSRE.2019.2934097.

[26] "Wireshark · Go Deep." [Online]. Available: https://www.wireshark.org/

[27] K. C. Welch, U. Lahiri, N. Sarkar, Z. Warren, W. Stone, and C. Liu, "Affect-Sensitive Computing and Autism," in Affective Computing and Interaction: Psychological, Cognitive and Neuroscientific Perspectives, Hershey, PA, USA: IGI Global, 2011, pp. 325–343 [Online]. Available: https://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-61692-892-6.ch015

[28] *TensorFlow*. [Online]. Available: https://www.tensorflow.org/

[29] J. Brownlee, "How to Choose Loss Functions When Training Deep Learning Neural Networks," *Machine Learning Mastery*. Aug-2020 [Online]. Available: https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/

[30] "Module: tf.keras.optimizers : TensorFlow Core v2.5.0," *TensorFlow*. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/keras/optimizers

[31] "E4 wristband: Real-time physiological signals: Wearable PPG, EDA, Temperature, Motion sensors," *Empatica*. [Online]. Available: https://www.empatica.com/research/e4/

[32] K. C. Welch, U. Lahiri, Z. E. Warren, and N. Sarkar, "A System to Measure Physiological Response During Social Interaction in VR for Children With ASD," *Computational Models for Biomedical Reasoning and Problem Solving Advances in Bioinformatics and Biomedical Engineering*, pp. 1–33, 2019.

[33] R. P. Joshi, M. K. Broek, X. Z. Tan, A. Choi, and R. Luo, "ROS OpenPose," *GitHub Repository*. GitHub, 2019.

[34] R.L. Williams II, "Baxter Humanoid Robot Kinematics", Internet Publication, https://www.ohio.edu/mechanical-faculty/williams/html/pdf/BaxterKinematics.pdf, April 2017.

[35] rethinkrobotics, "Joint_description.png," *sdk.rethinkrobotics.com*. [Online]. Available: https://sdk.rethinkrobotics.com/wiki/a/images/b/b0/Joint_description.

[36] "About the Baxter Robot: Specs, Accessories, Case Studies," *Baxter Robots*. Oct-2018 [Online]. Available: http://collabrobots.com/about-baxter-robot/ http://collabrobots.com/about-baxter-robot/

[37] "Wiki," *ros.org*. [Online]. Available: http://wiki.ros.org/indigo http://wiki.ros.org/indigo

[38] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 1 Jan. 2021, doi: 10.1109/TPAMI.2019.2929257.

[39] T. Simon, H. Joo, I. Matthews, and Y. Sheikh, "Hand Keypoint Detection in Single Images Using Multiview Bootstrapping," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 21-26 July 2017, pp. 4645–4653, doi: 10.1109/CVPR.2017.494.

[40] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 21-26 July 2017, pp. 1302–1310, doi: 10.1109/CVPR.2017.143.

[41] S. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional Pose Machines," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 27-30 June 2016, pp. 4724–4732, doi: 10.1109/CVPR.2016.511.

[42] "Kinesthesia Toolkit for Microsoft Kinect by University of Leeds - Toolkit for LabVIEW Download," *VIPM*. [Online]. Available: https://www.vipm.io/package/kinesthesia_lib_kinesthesia_toolkit/

[43] "Azure Kinect DK – Develop AI Models: Microsoft Azure," – *Develop AI Models | Microsoft Azure*. [Online]. Available: https://azure.microsoft.com/en-us/services/kinect-dk/#features

[44] docs.microsoft.com, “Azure Kinect DK hardware specifications,” *Microsoft Docs*. [Online]. Available: https://docs.microsoft.com/en-us/azure/Kinect-dk/hardware-specification

[45] P. I. Corke, Robotics, Vision & Control: Fundamental Algorithms in MATLAB, Second. Springer, 2017.

[46] “Azure Kinect.” [Online]. Available: https://img-prod-cms-rt-microsoft-com.akamaized.net/cms/api/am/imageFileData/RWqOsq?ver=2e37

[47] *baxter-1200x630.jpg*. [Online]. Available: https://robots.ieee.org/robots/baxter/baxter-1200x630.jpg

[48] “Connecting Baxter with workstation,” *Connecting Baxter with workstation - Lofaro Lab Wiki*. [Online]. Available: http://wiki.lofarolabs.com/index.php/Connecting_Baxter_with_workstation

[49] N. Torres, N. Clark, I. Ranatunga, and D. Popa, “Implementation of interactive arm playback behaviors of social robot Zeno for autism spectrum disorder therapy,” in *ACM International Conference Proceeding Series*, 2012, p. 21, doi: 10.1145/2413097.2413124.

[50] Taghavi, N., Berdichevsky, J.M., Balakrishnan, N., Welch, K.C., Das, S.K, and Popa, D.O. 2021 “Online Dynamic Time Warping Algorithm for Human-Robot Imitation” accepted, IEEE International Conference on Robotics and Automation (ICRA 2021)

[51] *Joint_Position_Control*. [Online]. Available: https://sdk.rethinkrobotics.com/wiki/Arm_Control_Modes#.22Raw.22_Joint_Position_Control

[52] *Joint_Position_Control.png*. [Online]. Available: https://sdk.rethinkrobotics.com/wiki/a/images/8/88/Joint_Position_Control.png

[53] *‘Raw’_Joint_Position_Control.png*. [Online]. Available: https://sdk.rethinkrobotics.com/wiki/a/images/8/8a/%27Raw%27_Joint_Position_Control.png

[54] “Arm_Calibration,” *sdk.rethinkrobotics.com*. [Online]. Available: https://sdk.rethinkrobotics.com/wiki/Arm_Calibration

[55] *Tuck_Untuck.png*. [Online]. Available: https://sdk.rethinkrobotics.com/wiki/a/images/1/11/Tuck_Untuck.png

[56] T.-W. Lu et al., “Symmetrical kinematic changes in highly functioning older patients post-stroke during obstacle-crossing.,” *Gait & posture*, vol. 31, no. 4, pp. 511–516, Apr. 2010, doi: 10.1016/j.gaitpost.2010.02.012.

[57] B. Lofterød, T. Terjesen, I. Skaaret, A.-B. Huse, and R. Jahnsen, “Preoperative gait analysis has a substantial effect on orthopedic decision making in children with cerebral palsy: comparison between clinical evaluation and gait analysis in 60 patients.,” *Acta orthopaedica*, vol. 78, no. 1, pp. 74–80, Feb. 2007, doi: 10.1080/17453670610013448.

[58] K.-H. Lin, T.-W. Lu, P.-P. Hsu, S.-M. Yu, and W.-S. Liao, “Postural responses during falling with rapid reach-and-grasp balance reaction in patients with motor complete paraplegia.,” *Spinal cord*, vol. 46, no. 3, pp. 204–209, Mar. 2008, doi: 10.1038/sj.sc.3102100.

[59] A. P. Rocha, H. Choupina, J. M. Fernandes, M. J. Rosas, R. Vaz, and J. P. Silva-Cunha, “Kinect v2 based system for Parkinson’s disease assessment,” in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Milan, Italy, 25-29 Aug. 2015, pp. 1279–1282, doi: 10.1109/EMBC.2015.7318601

CURRICULUM VITA

NAME: Jacob M. Berdichevsky

DOB: Louisville, KY – Mar 26, 1997

EDUCATION: B.S., Computer Science Engineering
University of Louisville
2015-2020

B.S., Electrical Engineering
University of Louisville
2015-2020