DESIGN OF A SYSTEM TO INVESTIGATE THE RELATIONSHIP BETWEEN
FEEDBACK AND DELIVERY MEDIUM FOR A NOVEL MOTOR TASK

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Mechanical Engineering

by

Ashley Humpal

December 2022
COMMITTEE MEMBERSHIP

TITLE: Design of a System to Investigate the Relationship Between Feedback and Delivery Medium for a Novel Motor Task

AUTHOR: Ashley Humpal

DATE SUBMITTED: December 2022

COMMITTEE CHAIR: Eric R. Espinoza-Wade, Ph.D.
Professor of Mechanical Engineering

COMMITTEE MEMBER: Stephen M. Klisch, Ph.D.
Professor of Mechanical Engineering

COMMITTEE MEMBER: Charlie T. Refvem, M.S.
Lecturer of Mechanical Engineering
ABSTRACT

Design of a System to Investigate the Relationship Between Feedback and Delivery Medium for a Novel Motor Task

Ashley Humpal

Stroke is a chronic, lifelong illness, and full recovery requires continuous physical and cognitive rehabilitation. Such long-term rehabilitation is cost-prohibitive; however an approach to providing long-term therapy that has recently gained traction is the use of socially assistive agent (SAA) systems. These systems make use of non-contact communication devices and can be used to guide people through a variety of rehabilitative tasks. They have the potential to supplement current rehabilitation practices by providing motivation during intense exercises, and can extend the reach of the therapist into remote and home settings.

Though SAA systems have been used in a variety of rehabilitative and assistive contexts, there remain questions regarding the best design for such systems. Currently there is a lack of detail on what type of feedback optimizes user performance, and the role that the delivery medium (e.g., a human coach, a tablet, or a robot) plays in user performance. The purpose of this thesis is the design of a system to investigate the interaction between feedback and medium type when implemented for a novel motor task.

The selected task is modeled on the shuffleboard game, with the delivery medium including a human coach and tablet and two types of augmented feedback. The designed system incorporates various hardware and software components. A vision system communicates with a laptop to record and analyze motor task data, with a program that also interfaces with a control circuit. The control circuit may transmit data through Bluetooth to a custom-built app on the tablet, which then provides
augmented feedback with audio dialogue. Otherwise, the human coach is provided
designed feedback from the laptop. An initial system evaluation was performed with
this constructed system using pilot participants to validate the design.

The initial system evaluation demonstrated the ability to improve participant perfor-
manence; however, it also demonstrated a high level of task difficulty. Several changes
may need to be incorporated to the system to ensure better learning for participants.
This includes changes to the physical setup, as well as changes to the frequency of
the augmented feedback. This thesis may be used as the foundation for future exper-
imentation with different delivery media or types of augmented feedback to discover
how to best optimize user performance for a novel motor task.
ACKNOWLEDGMENTS

I would like to express my gratitude toward my committee chair, Dr. Espinoza-Wade, who played a crucial role in the development of my system. His support and guidance allowed this thesis to be possible. I also want to thank my other committee members, Dr. Klisch and Charlie Refvem, who gave valuable input and recommendations. I additionally want to acknowledge the contributions of Juan Diego Luna for a portion of the AppInventor software, Nishka Chawla for contributing to a section of code, and Heon Joo for preliminary experimental design considerations. Last but not least, I want to extend thanks to my family and friends who provided tremendous support.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LIST OF TABLES</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF FIGURES</td>
<td>xi</td>
</tr>
</tbody>
</table>

## CHAPTER

1. **Introduction** ........................................ 1
2. **Background** ........................................... 3
   2.1 Stroke ............................................... 3
   2.2 Current Stroke Care ................................. 3
   2.3 Socially Assistive Agents and Augmented Feedback | 4 |
   2.4 Research Objective .................................. 6
3. **Methods** .............................................. 7
   3.1 Introduction .......................................... 7
   3.2 Background ............................................ 7
   3.3 Experiment Design .................................... 9
   3.4 Study Procedure ...................................... 13
   3.5 Specific KR and KP Feedback Used ................... 14
   3.6 Basic Software and Hardware Interaction .......... 14
   3.7 Software Front End (Python Scripts) ............... 15
   3.8 Software Back End (MicroPython Scripts) .......... 19
   3.9 AppInventor Code ..................................... 22
   3.10 IMU Software ......................................... 28
   3.11 Physical Components ................................. 28
   3.12 Hardware Additions .................................. 29
3.13 Calibration ................................................................. 32
3.14 Study Setup .............................................................. 34
3.15 After Participant Arrival ................................................. 35
3.16 Post Session Survey ...................................................... 37
3.17 Scoring of Trials ......................................................... 37
4  Results ................................................................. 39
   4.1 Introduction ............................................................ 39
   4.2 Participant Task Performance ......................................... 39
   4.3 Participants Residual Error ............................................ 42
   4.4 IMU Data .............................................................. 46
   4.5 Survey Results ......................................................... 55
5  Discussion and Interpretation of Results ......................... 59
   5.1 Introduction ............................................................ 59
   5.2 System Data Collection ............................................... 59
   5.3 Participant Zone Frequency .......................................... 60
   5.4 Residual Error Before and After Retention Trials ............... 61
   5.5 IMU Sensor Data Analysis ........................................... 61
   5.6 Survey .............................................................. 63
   5.7 Limitations of the System Design ................................... 64
   5.8 Experimental Conclusions and Recommendations ............... 65
   5.9 Future Applications ................................................ 66
6  Conclusion ............................................................ 67
BIBLIOGRAPHY .............................................................. 69
APPENDICES
   A  Methods ............................................................. 73
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Component List</td>
<td>28</td>
</tr>
<tr>
<td>3.2</td>
<td>Scoring</td>
<td>37</td>
</tr>
<tr>
<td>4.1</td>
<td>Number of Respondents for SUS Likert Scale Questions</td>
<td>57</td>
</tr>
<tr>
<td>4.2</td>
<td>Number of Respondents for Supplemental Questions</td>
<td>58</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Experimental Setup</td>
<td>11</td>
</tr>
<tr>
<td>3.2</td>
<td>Tabletop Dimensions</td>
<td>12</td>
</tr>
<tr>
<td>3.3</td>
<td>Process Flow</td>
<td>15</td>
</tr>
<tr>
<td>3.4</td>
<td>Frontend State Diagram</td>
<td>16</td>
</tr>
<tr>
<td>3.5</td>
<td>Backend State Diagram</td>
<td>19</td>
</tr>
<tr>
<td>3.6</td>
<td>Connecting to Device</td>
<td>23</td>
</tr>
<tr>
<td>3.7</td>
<td>Tablet While Waiting to Connect</td>
<td>23</td>
</tr>
<tr>
<td>3.8</td>
<td>Bluetooth Connection</td>
<td>24</td>
</tr>
<tr>
<td>3.9</td>
<td>Bluetooth Connected Notification</td>
<td>24</td>
</tr>
<tr>
<td>3.10</td>
<td>Visual of App during Trials</td>
<td>25</td>
</tr>
<tr>
<td>3.11</td>
<td>Reading Data</td>
<td>26</td>
</tr>
<tr>
<td>3.12</td>
<td>Tablet Visual While Playing Audio</td>
<td>26</td>
</tr>
<tr>
<td>3.13</td>
<td>Tablet Visual Immediately After Playing Dialogue</td>
<td>27</td>
</tr>
<tr>
<td>3.14</td>
<td>Code for Post-Verbalization Display</td>
<td>27</td>
</tr>
<tr>
<td>3.15</td>
<td>Code for Pre-Verbalization Display</td>
<td>27</td>
</tr>
<tr>
<td>3.16</td>
<td>Push Button Circuit</td>
<td>29</td>
</tr>
<tr>
<td>3.17</td>
<td>Physical Push Button Circuit</td>
<td>30</td>
</tr>
<tr>
<td>3.18</td>
<td>Encased Button</td>
<td>30</td>
</tr>
<tr>
<td>3.19</td>
<td>Nucleo-L476RG and DSD TECH HC-05 Connections</td>
<td>31</td>
</tr>
<tr>
<td>3.20</td>
<td>Physical Nucleo-L476RG and DSD TECH HC-05</td>
<td>31</td>
</tr>
<tr>
<td>3.21</td>
<td>Camera and Partition Setup</td>
<td>33</td>
</tr>
</tbody>
</table>
Chapter 1

INTRODUCTION

In the United States about 795,000 people are affected by stroke each year [1]. Between 2017-2018 costs related to stroke, such as for medicine and health care, accumulated to 53 billion dollars. Stroke generally leads to a decrease in overall quality of life as tasks that previously were easy to perform, are now more difficult due to the damage caused. In order to recover from stroke, rehabilitation exercises and different therapies are tried [2]. These can be conducted by a therapist meeting with individual patients which can take a up a great deal of time and resources. Meeting with a therapist on a regular basis requires money and resources for the patient. Costs for an individual stroke patient in the United States averages to $59,900 a year [3].

Because stroke is a chronic, lifelong illness, full recovery requires continuous physical and cognitive rehabilitation. However, such long-term rehabilitation is cost-prohibitive. One approach to providing long-term therapy that has recently gained traction is the use of socially assistive agent (SAA) systems. These systems make use of non-contact robots, and can be used to guide people through a variety of tasks. Through this ability, they demonstrate the potential to be used to supplement current rehabilitation practices by acting as a verbal coach to help motivate patients and talk them through exercises. If successful, these systems can extend the reach of the therapist into remote and home settings. This could then lead to long-term in-home therapy.

Though SAA systems have been used in the context of guiding people through tasks, there remain questions regarding the best design for such systems. Currently there
is a lack of detail on the type of feedback most relevant to ensure user performance is optimized, as well as what role delivery medium plays in the performance of users. While stroke rehabilitation is the long term target, the current goal of this thesis is the design of a system to investigate the interaction between feedback and medium type when implemented for a novel motor task. Through the design of a system and study procedure, this thesis can be used as the foundation for future experiments to investigate the relationship between feedback and media and to discover whether these factors contribute differentially to user performance.

This thesis is organized according to the following: Chapter 2 provides background on current stroke care and socially assistive agents. Chapter 3 describes the experimental methods and research design, including the study procedure and necessary construction. Chapter 4 presents the results and Chapter 5 will provide discussion and interpretation of those results and future applications of this study. Finally Chapter 6 summarizes all the findings of this thesis.
Chapter 2

BACKGROUND

2.1 Stroke

A stroke occurs when a blood vessel ruptures or blood flow is blocked in the brain which causes brain damage [4]. In the United States, each year about 795,000 people are affected by stroke [1]. It can negatively affect an individual’s speech, vision, movement and even lead to paralysis in certain parts of the body. Each individual is affected differently by stroke as it matters which part of the brain the stroke influenced and damaged. Depending on which area of the brain is harmed, the ability to perform its associated functions is also impeded [5]. Due to these negative effects, the quality of life for individuals who have had stroke decreases as they may have trouble doing tasks they previously found easy. Although stroke may affect speech, vision, and many other areas, the long term target of this thesis is addressing the motor aspects of stroke recovery.

2.2 Current Stroke Care

There are a variety of therapeutic interventions designed to restore motor function after stroke. One promising option based on research studies is Constraint-Induced Movement therapy, where the patient’s unaffected limb is restricted, while the stroke affected limb is repeatedly used in tasks as the difficulty increases [2]. This focuses on increasing and enhancing motor skill in the stroke affected limb. Another form of stroke therapy is the utilization of muscle strengthening exercises [2]. These exercises
focus on using the stroke-affected limb and increasing its strength through varying tasks. Another practice is mental practice with motor imagery [2]. In this therapy, the patient envisions the motion they want to do with their stroke affected limb. These forms of stroke therapy may be supervised with therapists whose purpose is to help direct patients, give them feedback, or to motivate them. The end of the formal process of rehabilitation is generally between three and four months [6]. However, stroke is a chronic disease, and the impairments continue long beyond this brief window. The current health system is lacking as it does not support stroke survivors in different areas such as maintaining activity and quality of life as well as social reintegration among other areas [6].

The current health system varies in stroke treatment as different facilities may follow their own distinct procedures, making treatment variable at times. The estimated direct cost of stroke annually in the United States is 28 billion dollars [7]. A portion of this overall cost comes from the money spent for hospital outpatient (medical care that does not require a hospital stay) or office-based provider visits, which is 2.4 billion dollars. It may be difficult for patients to find or afford to continue meeting with a therapist as it can be an additional expensive cost, especially if their insurance only covers a limited number of therapy sessions. For this reason, tools that can extend the effect of the therapist beyond this limited time window and facilitate long-term care have become increasingly attractive.

2.3 Socially Assistive Agents and Augmented Feedback

A socially assistive agent, SAA, is an agent (such as a robot) whose primary goal is to help the user through social interaction, such as coaching, and without any physical contact [8]. Through different interactions its objective is to help the user
with their progress in a measurable way. Socially assistive agents have been used in various tasks. They have been used to help the elderly by keeping everyone up to date with the schedules in their nursing homes [8]. They have also been used to interact with children with Autism spectrum disorders to help them in a therapeutic sense [8]. Specifically they have been used as playmates to help children learn social skills through social interactions with the SAA [9]. SAA has even been used with Constraint Induced Movement therapy for stroke rehab patients to help motivate them to continue exercises [8].

SAA systems utilize embodiment as well as provide augmented feedback [10]. Embodiment is a term used to describe the physical features of the agent such as its size and shape and the impression it gives to users consciously or subconsciously [10]. Embodiment affects the user’s performance. For example, in a weight-loss study by Kidd, participants in the experiment were made to track their weight either on paper, by computer, or by robot that had human-like features [11]. Participants rated the robot system the highest and used it for a longer duration than the other two methods. Each of the three methods were the same except for their delivery medium. In this case the presence of the physical robot made the users feel they had to be more responsible in regards to continuing the program.

SAA may also utilize augmented feedback. Augmented feedback is performance feedback that is not normally available through traditional senses. [10]. For instance, this feedback can take the form of motivational coaching in order help participants get better at motor tasks. There are different types of augmented feedback that are useful, which will be explored further in section 3.8. In spite of the maturity of these fields, the interaction between embodiment and feedback type is not well known. Furthermore, this interaction and its relevance to motor task practice has not been extensively studied.
2.4 Research Objective

The purpose of this thesis is the design of a system to investigate the interaction between feedback and medium type when implemented for a novel motor task. This system will focus on determining the differences in performance of individuals, based on the delivery method they work with, may it be SAA or human, and to determine features of the delivery medium (such as embodiment) that lead to those differences. Furthermore, a method for determining the relationship between the feedback type and progress in motor task performance will be constructed. The concept of comparing feedback was explored in a previous study where the rate of recovery of arm motor function was compared when doing a motor task with either KR (Knowledge of Results) or KP (Knowledge of Performance) feedback, whose definitions will be thoroughly discussed in section 3.8 [12]. The motor task in question for the study was pointing to an object with the stroke affected arm with vision obstructed and then being given feedback to correct the movement. This study found that KP feedback was more effective for participants as they got better with the motor task faster than those with KR feedback, and that after a month they retained the motor control skills they learned. This study worked only with a human giving the feedback and did not include the use of a SAA. This raises the question as to whether this finding holds true for SAA systems as well, or if delivery medium influences the users performance. The system presented in this thesis is designed to be used with a two by two cross-sectional study design to investigate the roles of embodiment (tablet vs. human) and feedback type (Knowledge of Results vs. Knowledge of Performance) in motor task performance in a population on control participants. The motivation is to determine how these features interact to affect user performance. This is the long-term goal of the system, however the immediate goal of this thesis is to determine if the system performs as expected and its validity.
Chapter 3

METHODS

3.1 Introduction

This chapter describes methodological considerations for the current study. The overall goal is to evaluate how motor task performance changes in response to varying feedback and feedback delivery media. This experiment involves human participants who performed an upper extremity motor task. The study involves the development of novel hardware, software, and methodologies to evaluate the relationship between feedback and media and to discover whether these factors contribute differentially to user performance.

3.2 Background

Motor task practice and augmented feedback are critical components of the current study. Individuals who have had stroke re-learn movement patterns that they may have lost due to the damage their stroke caused them, through motor task practice and motor learning [2]. Furthermore, studies have been conducted analyzing the effect of practicing and repeating motor tasks. In one such study by Crossman, factory workers working with a hand operated jig to make cigars were studied [13]. It was found the higher the number of cigars the workers had made and therefore the longer their practice time was, the faster their production time was. Essentially, practicing a motor task results in increased performance for the user and makes them better at performing the task. Additionally, motor learning and motor practice
tasks are already currently used to help with the recovery process for stroke patients. CIMT, Constraint-Induced Movement Therapy, includes repetitive tasks with the stroke affected limb while the non-affected limb is constrained [2]. CIMT relies on repeated, increasingly challenging motor tasks to foster motor learning and has been shown to be beneficial to stroke patients. This therapy will be used as a basis for the study.

Another important aspect of this study is augmented feedback. Augmented feedback provides supplemental information to whatever is naturally available [14]. One type of augmented feedback is Knowledge of Results (KR). KR is information that can be verbalized and discusses the outcome in terms of the goal, after the attempt has been made [14]. For example, if the goal is to land a target in a specific area, then one form of KR feedback can be that the participant was “12 units too long” or far away [14]. Knowledge of Performance (KP) feedback focuses on the movement pattern or motion and correcting it, one such example being “Your elbow was bent” [14].

Various studies with each of these types of feedback have been conducted. Kinematic feedback, which involves movement as it relates to position, coordination patterns, and other factors, is a form of KP feedback [14]. It has been used in the past in a study by Lindahl to increase industrial workers’ skill levels [15]. Foot pedal patterns of skilled workers were studied and KP feedback was given to new trainees to learn those patterns [15]. The workers who received the feedback were able to match the experience of workers who had nine months of experience after a minimum of 10 weeks, demonstrating the success of KP feedback as well as the effects of motor learning.

On the other hand, KR feedback has also shown success. In another experiment by Bilodeau et al., groups of subjects attempted a linear-positioning task, where one group was given no KR feedback and the other three groups were given differing rates of KR feedback [16]. The group that received the most KR feedback had the
biggest immediate decrease in error, while the group with no KR feedback had the
least decrease in error. These studies demonstrate that the success of KR and KP
may be specific to the task context. Therefore, development of new systems may need
to separately evaluate the roles of KR and KP.

Furthermore, researchers have investigated the rate, or frequency of feedback [14].
There has been no absolute rate of feedback that is the best, however feedback at a
frequency of 33% (once every third trial) has shown promise in prior work [17]. In
an experiment by Schmidt, individuals participated in ballistic timing task in which
they moved a handle and slide, into different target areas [17]. It was found that
feedback given at a rate of 33% had the least error, when compared to the rate of
100% feedback.

There have been studies that have compared the differential effects of KR and KP
feedback on user performance. One such study by Cirstea et al., had a human in-
structor give KR and KP feedback to stroke survivors who participated in a pointing
task [18]. It found that the KP feedback had better results than the KR feedback.

3.3 Experiment Design

The study by Cirstea suggests that KP feedback is preferred for optimal results when
a human provides the feedback. However, this raises the question on systems where
the delivery media may not be a human instructor or coach. For example, if designing
a robotic system, it is not known whether the same premise holds. This thesis seeks
to address this question.

The goal of the current study design is to determine the interaction between feedback
delivery media and feedback type. In order for there to be a need for feedback, the
coaches who provide the feedback must be saying something that the users themselves would not be able to observe. In this study, blocking user vision while participants do a motor task satisfies that criteria. The motor task we use in this study is an upper-extremity, goal-directed task. Participants will attempt to push a puck into a tabletop ‘goal zone.’ For this process, I needed to design a system that automatically recognizes puck location. Additionally, the KR or KP feedback provided by the coach must be tailored to the task.

Given the study goal, the study design must subscribe to specific requirements. First, it must use a learnable motor task. The selected task is modeled on the shuffleboard game, and will involve pushing a puck to a target location. Because augmented feedback is supplemental to readily available feedback, participants will be not have visual feedback during performance. For this process, a system that automatically recognizes puck location is needed. The puck location will be verbalized to participants post trial, after every third trial as this rate has shown promise. Additionally, the KR or KP feedback that the coach provides is something that will also be designed. Lastly, participants will be wearing an inertial measurement unit (IMU) armband that will track their motion data throughout the trials, specifically their different linear and angular velocities. In some KR and KP tasks, this metric can be used to both describe motor learning and the quality of the learning process so it is a component of the study.
Figure 3.1: Experimental Setup

Figure 3.1 depicts a top-down view of the experimental setup. The primary components of this experiment are a portable tabletop, a board to block participant vision, pucks, tablet and stand, control circuit, camera and tripod, and a laptop. For specific components refer to Table 3.1. The tabletop has markings showing zones one through ten, with an additional goal zone in the center. Refer to Figure 3.2 for the dimensions and spacing of the lines on the table top. At the front of the table top a board with a small rectangular sized cutout is attached to the table top. The cutout begins at the bottom of the board and is centered so it aligns to the middle of the tabletop, and is 13in.×3.25in. The cut out is for the puck so that it can still slide through
while the rest of the board obscures the users vision. At the back of the table top a camera is placed on a tripod so that it can view all the zones. The camera is there to monitor the location of the puck during the motor task. The camera is plugged into the laptop where code analyzes the images the camera takes. The laptop is also connected the control circuit, which monitors the task results. The tablet and stand are positioned to one side in front of the user so they have a clear view of it, as it will be giving them verbal feedback during the trials. Participants perform the task while seated. The purpose of the study is piloting the hardware with control participants.

![Figure 3.2: Tabletop Dimensions](image-url)
3.4 Study Procedure

Participants will first be given a Consent and Demographic form that has been approved by the Cal Poly Institutional Review Board, found in Appendix A.1 and A.2, to sign before beginning the experiment. They will then be seated at the test station. There is initially some verbal introduction from a human who is supervising. The participants are told what the focus of the task should be, which is to repeatably push pucks with their dominant arm and try to aim for the goal zone without being able to see the final position of their puck. This will require participants to focus on their arm movement and adjust it accordingly. There is a marked position for them to rest their elbow on while they push the puck. They will be given 5 practice trials with feedback each time to ensure they understand the procedure for the experiment.

Once they have had an opportunity to ask questions, the experiment will commence. They will be given the IMU armband and will be seated. They are allowed to adjust their seat height so that their arm can rest comfortably on the table. After every third trial in a session, the experimenter gives them augmented feedback to inform them of their performance. Refer to Appendix A.3 for the full script that will be used by experimenters. Each participant will see only one type of feedback from KR or KP, and one delivery medium from either the human or tablet. Each participant will perform 200 trials. After they finish their trials, they are given a 5 minute break and then receive a Retention test.

During the Retention test the participants perform 5 trials where they receive no feedback. The purpose of the test is to evaluate how much the participants have retained from practice. Afterwards, the experimenter gives them a few closing statements to conclude the entire session. The entire session takes approximately one to two hours.
3.5 Specific KR and KP Feedback Used

A participant will receive either KR or KP feedback. The KR feedback groups will be given positional feedback, specifically what zone their puck is in or if it is out of bounds. The KP feedback groups will be given feedback relative to their pushing movement strength. They will be told to push harder or softer at different levels. Both blocks of groups will be informed when they have reached the goal area or if their puck was not detected. This feedback will be given after every third trial. Knowing the result of their last trial through the feedback will allow them to adjust accordingly for the next trial.

3.6 Basic Software and Hardware Interaction

This experiment relies on a robust hardware/software interaction. From a software standpoint the platforms used were Python, MicroPython, and AppInventor. Figure 3.3 depicts a flow diagram of the interaction between the different software platforms. The camera connects to the laptop and captures the puck location. A python script running on the laptop extracts the puck location using OpenCV, a tool designed for image-based programming [19]. After the puck zone is determined from the pixel positions, a character representing the zone is sent over serial communication to the control circuit. The control circuit utilizes the Micropython platform to determine when to give the necessary feedback and to communicate with the tablet via the attached Bluetooth device. Installed on the tablet is the app “Motor Training Task” which AppInventor was used to create. It is programmed to read the received Bluetooth character and play its respective dialogue on the tablet. This process repeats until all the trials are completed.
3.7 Software Front End (Python Scripts)

The frontend code facilitates picture taking and extracts puck location. One file facilitates the camera module. In it, one method initializes the camera. Another creates a reference image of the tabletop taken prior to any trial while the tabletop is empty. If it is unable to capture the snapshot successfully it sends a message to the user, otherwise the image is saved as a PNG file type and is converted to gray scale. The next method takes an additional image, converts it to gray scale and subtracts the reference image from it while applying a threshold. Then it locates the pixels that remain, which is the part of image that has changed (the new puck location) and saves the $x$ and $y$ values. If no change has been detected it appends the value -1 for those instead. Those pixel values are stored and saved for further use. The last
method closes all images and releases the camera from use. The file is attached in Appendix A.4.

![Figure 3.4: Frontend State Diagram](image)

The second file, found in Appendix A.5, controls the timing of the camera images. The input parameters necessary to run the file are feedback type and media type. “1” and “2” are used to represent KR and KP feedback respectively, while “3” and “4” are used to represent the tablet or human experimenter. The file contains a finite state machine with a total of six states. A finite state machine (FSM) is a template or framework that can be used in software requiring logic. It contains multiple states, represented by ovals or circles, that are transitioned to and from by various inputs, which are represented by text above arrows. Processes begin at one state, receive an input, and then transition to their next state. Figure 3.4 gives an overview of the states and their transitions. Before it begins cycling through states it first initializes
the camera and takes a reference image of the blank tabletop. In the Initialization or Init State (S0) it waits for the backend to send it a letter, specifically an “s” so it knows that the tablet is playing the intro message. It also prints a message so a human experimenter knows to begin the introduction. Afterwards, it transitions into Trial (S1) where it first checks to make sure the trial limit has not been exceeded. If it has not, it checks for another letter, “f”, from the backend indicating a break has started. If it has received that character, it transitions to Break (S2), where it continually loops until it reads the same letter to exit that state and transitions back to Trial (S1).

If the break character has not been received the code continues through Trial (S1). It sends the character “t” to the backend notifying it that a trial has started. It waits a certain amount of seconds than takes a snapshot, after the participant has pushed a puck. Once the trial is completed the frontend sends the character “v” to the backend notifying it the trial has ended. It then averages the $y$ min and max pixel values to find where the center of the puck is located, and appends that value to a list. The $y$ average gets compared to a set of if else statements that separate each zone into ranges. If the average falls within one of the ranges, then the puck is located in that zone. The frontend then sends a distinct character specific to the zone to inform the backend which zone the puck is in. The zone number additionally gets appended to a list. Each of the if statements for the zone ranges are also formatted so that the console prints the feedback the tablet or human will be saying in the first trial and every third trial after that. This makes it easier for the human coach to keep track of when to give feedback, and can also be used to check the accuracy of the tablet. The code then transitions back to the beginning of Trial (S1) after checking for another break.
If the trial counter ever exceeds the trial limit, then it transitions to Retention Start (S3). This state sends the character “r” to the backend notifying it that the Retention trials will start in 5 minutes. After waiting 5 minutes it then sends the character “q” to indicate that the Retention Trials are now starting. It transitions to Retention Start state (S4) the retention trials begin. The state is very similar to the Trial State (S1) except it does not provide feedback nor check for the button since the duration for these trials is much shorter. After the trials are complete it transitions to the End State (S5), the final state. In this state it closes out the camera, and waits to receive an encoded ascii list from the backend with data. It formats the list and all the other lists it appended and saves all the data to a CSV file. The CSV file has the input parameters in the first row, with the experimenter type in the first column and media type in the second. The second row and onward contain the trial number, zone number, y pixel position value, and feedback data received from the backend in separate respective columns.
3.8 Software Back End (MicroPython Scripts)

Figure 3.5: Backend State Diagram
The Micropython code (located in Appendix A.6) serves as the backend and contains a finite state machine and several methods. There are a total of 11 states. Figure 3.5 gives an overview of the states and their transitions. It also sets up a timer and button object. The button is handled by an interrupt that is triggered by the voltage falling edge when the button is pressed. A callback function is then executed where a global variable, “button” is set to the value 1. This indicates that the button has been pressed which is important later in the code.

Another method initializes all the local variables and input parameters, such as “test_type” which the user inputs to specify whether the code will run the KR or KP trials. If “test_type” is input as 1, then the code will use the KR feedback, and if it is 2 it will use the KP feedback. An additional method takes a number and sends it via Bluetooth to the tablet, which will be discussed more in depth in the next section.

The code begins in the Init State (S0) and immediately transitions to the Welcome State (S1). In this state the finite state machine sends a character, “s” to the frontend signifying that the introduction message is about to play. It also sends a character via Bluetooth to the tablet to start playing the audio introduction message. It transitions to Receive Data (S2) where it waits to receive a character in ASCII from the frontend notifying it that the trial has started. It also checks if the global variable “Button” has been set to 1 or if it remains 0. If it has be set to 1, that means the button was pressed and the interrupt triggered. If it has been triggered it resets the button value to 0 and sends a message, as the character “f”, to the fronted to signal a break is needed. It also makes sure to clear any other messages the frontend may have sent at that point. It then transitions to Break (S7) where it loops until the button has been pressed again and the global variable “button” equals 1 again. Once that occurs it transitions back to Receive Data (S2).
When in Receive Data (S2), if a character is received signifying the trial has started, the finite state machine sends a Bluetooth character to the tablet to play the “Trial Start” audio and then the finite state machine transitions to Trial End (S3). In this state, the finite state machine (FSM) waits until it receives another ASCII character notifying it that the trial has ended and sends a corresponding Bluetooth character to make the tablet play the “Trial Over” dialogue. Once that occurs the FSM transitions to Converting Data (S4).

In Converting Data (S4) the data is decoded and its unicode value is found. Then based on whether the test_type variable equals 1 or 2 the FSM transitions to Send KR (S5) 5 or Send KP (S6). Each state works with a different type of feedback. In both states feedback is given during the first trial and every third trial after that with the use of a counter help ensure accuracy. There are a set of if else statements to determine which character was received, and a corresponding Bluetooth message is sent. The feedback for each trial is appended to a list and the FSM transitions back to Receive Data (S2). This process is repeated until the total trial counter is reached. When that counter has been reached the FSM transitions to Retention Start (S8) where the retention trials start. This state loops until it receives a character signaling the retention test will start in five minutes and sends a character via Bluetooth to the tablet to play that dialogue. State 8 loops until it receives another character signaling the retention test is actually starting and sends another Bluetooth character. It then transitions to Retention Test B (S9) which has similar functionality to Receive Data (S2), where it waits for a character to signify the trial is starting. It then transitions to Retention Test F (S10) which is similar to Trial End (S3) where it waits for a character to signify the end of a trial. The FSM loops between State 9 and 10 until the retention counter has been reached, and then from Retention Test B (S9) it transitions to End (S11) which is the final state.
In the End State (S11), a Bluetooth character is sent to tablet so it plays an audio message stating the experiment session is over. Then the list of feedback data is formatted and sent to the frontend. Then the system exits. To actually execute this code, a separate main file exists, attached in Appendix A.7. It simply creates the task and takes a user parameter, either a 1 for KR feedback or a 2 for KP feedback.

3.9 AppInventor Code

AppInventor is a visual programming tool designed for creating apps on Android tablets and devices. These apps use coding ‘blocks’ to develop software functionality. The AppInventor code follows several steps. The first step is to initialize the app and to start the connection process, as seen in Figure 3.6.
Figure 3.6: Connecting to Device

The device address to be connected to is manually input into the code. While the app is loading, the app’s screen displays two circles that appear to be eyes as seen in Figure 3.7.

Figure 3.7: Tablet While Waiting to Connect
After initialization, the app searches for a paired device as indicated on Figure 3.8. A notification appears after alerting whether the device is connected or not in this block in Figure 3.9.
If the tablet is connected to the Bluetooth device it will appear to smile, shown in Figure 3.10. In between trials, this is also how the app will appear to the participant. This is to help the participant feel at ease while they are doing trials.

![Figure 3.10: Visual of App during Trials](image)

The next step is the app searching for any data sent to it via Bluetooth, occurring in the first step of the block seen in Figure 3.11. Once the app on the tablet receives a character it checks that character in a set of if else statements. If that character matches one of the app’s characters, the app will then play an audio dialogue for the given statement.
While the app is playing the dialogue, the image of the smile on the tablet screen changes so that instead of the face smiling it looks like its opened its mouth as its speaking, depicted in 3.12. The rest of Figure 3.11 show the blocks of code that execute these tasks.
After the tablet has finished the audio dialogue the image then changes once more to look like a closed mouth, seen in Figure 3.13.

![Figure 3.13: Tablet Visual Immediately After Playing Dialogue](image)

Another clock is then triggered from that step. Figure 3.14 shows the block for changing the tablet face and the clock.

![Figure 3.14: Code for Post-Verbalization Display](image)

Figure 3.15 depicts the last step where after the clock is triggered (meaning the audio dialogue has finished playing), the image changes back to a smiley face. This process is repeated for each trial as the app receives more characters via Bluetooth.

![Figure 3.15: Code for Pre-Verbalization Display](image)
3.10 IMU Software

The inertial measurement unit (IMU) device used in this study is the MindRove armband [20]. It uses custom software that allows for data to be streamed and visualized in realtime. However, the MindRove armband is recommended for this procedure as it is accompanied by the MindRove application which allows participants to easily record data. Data recordings can be started and saved at any time and are automatically converted and stored in the form of CSV files for easy access. The application can record data related to muscle activation (EMG data), linear acceleration and angular velocity about the \( x \), \( y \) and \( z \)-axes, and time related data.

3.11 Physical Components

A variety of different items can be used for this system, including for the camera, tablet, control circuit, and Bluetooth device. Refer to Table 3.1 for the integral components used for this initial study.

<table>
<thead>
<tr>
<th>Description</th>
<th>Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Circuit</td>
<td>Nucleo-L476RG</td>
</tr>
<tr>
<td>Tablet</td>
<td>Samsung Tablet A</td>
</tr>
<tr>
<td>Bluetooth antenna</td>
<td>DSD Tech HC 05</td>
</tr>
<tr>
<td>Camera</td>
<td>Nova Webcam</td>
</tr>
<tr>
<td>Laptop</td>
<td>Dell Precision 5530</td>
</tr>
<tr>
<td>IMU (accelerometer/gyroscope)</td>
<td>MindRove armband</td>
</tr>
</tbody>
</table>

The Samsung Tablet A is an android tablet, which is compatible with the AppInventor infrastructure which influenced why it was chosen. The DSD Tech HC-05 is also compatible with android devices in addition to different Nucleo models so it was also chosen based off those specifications. The MindRove armband can measure linear
velocites and angular velocities, which can be used in future experiments to examine
the relationship between trial performance and physical motion.

3.12 Hardware Additions

Both a Bluetooth device and button are necessary additions to the control circuit. The button is a simple push button that closes a switch when pressed. Its circuit is shown in Figure 3.16.

![Push Button Circuit](image)

**Figure 3.16: Push Button Circuit**

When the switch is closed, current flows to the other side of the switch to the pin, changing its state from low to high. This change can be read in the control circuit with the use of an interrupt, as was done for this experiment. The button allows the human supervisor to pause the trials if pushed at the request of the participant. Figure 3.17 shows the actual wiring of the push button, which was later put into a case. The case can be seen in figure 3.18. The procedures required when assembling the button were simply the wire stripping, and utilizing a drill press to create a hole in the case to mount the button.
The Bluetooth device enables the communication between the control circuit and tablet. Different types of control circuits may work with varying Bluetooth devices. Figure 3.19 shows the pin connections necessary to have the DSD TECH HC-05 device work successfully with the Nucleo-L476RG as the control circuit.
The actual DSD TECH HC-05 device and the Nucleo-L476RG that were used are pictured in Figure 3.20. The Nucleo manual was used to determine these connections [21].
3.13 Calibration

Before testing with any participants, calibration of the system was needed. The calibration of the web camera that was used was necessary. The software calculated the pucks location with the camera by taking a new image each trial and subtracting that with the original reference image. This signifies the importance of the rigidity of the camera to ensure it does not move or change its angle. The camera was mounted on a tripod where it could observe the tabletop and it was fixed in that position. After putting the camera in the final position, each zone needed to be calibrated to get its correct limits. This was done by marking the center of a puck and placing it on several locations on a zone’s line. The software was then used to calculate the average $y$ pixel values on the line, and those values were additionally averaged to get one average $y$ value for each of the lines surrounding the zones. Those were then integrated into the frontend code and tested to ensure accuracy.

An issue that was noticed after continual testing was the effect of the experimenter’s shadow. When the tablet plays the audio dialogue saying “Trial Over” the experimenter walks over to the tabletop and collects the puck. At all other times the experimenter will be standing to the side and during those times their shadow may be captured on the tabletop. This would skew the trial results as the software averages the $y_{max}$ and $y_{min}$ pixel values. Eventually an extra light source was added to minimize the shadows as well as an extra partition that the experimenter would stand behind until it was time to collect the puck, which can be seen in Figure 3.21a and Figure 3.21b respectively.

Another issue that occurred was the puck would sometimes roll off the tabletop. That was solved by adding borders to the tabletop, as can be seen in Figure 3.22.
Pucks were allowed to bounce off the side borders, but not the border along the back of the tabletop as that would skew accuracy of the trials greatly if the puck were to re-enter the zone areas. This was ensured by having a small gap between the tabletop
and back border so the puck would not be able to bounce back up on the tabletop, as seen in Figure 3.23.

Figure 3.23: Back Border

For the specific procedure used during this experiment please refer to Appendix A.8.

3.14 Study Setup

Prior to participant arriving at the study location, several tasks must be completed. The hardware must be activated and started in a specific sequence. The laptop must be charged with unnecessary applications closed. The control circuit with the Bluetooth device must be powered on and connected to laptop alongside the camera. Additional lighting should be turned on at this point. It is important to make sure that individual’s shadows or movements cannot be captured by the camera. It may be necessary to use extra partitions for the experimenter to stand behind or cover the participant’s other’s limbs if they are in view of the camera. Next, if the tablet is going to be the delivery medium, then it should be powered on and connected
to a charger and on a stand. Subsequently, the human supervisor must go into the Bluetooth settings menu of the tablet to pair it to the Bluetooth device connected to the control circuit. After, the app can be launched from AppInventor or if it has been downloaded, then directly from the tablet. The tablet smiley face should appear if the Bluetooth connection was properly established with the tablet.

Next, the group that the participant will in for the experiment should be randomly selected. Once that has been determined then the human supervisor will manually go into the file that executes the backend code and change the input parameter to a 1 or 2 depending if it is KR or KP feedback. This must similarly be done for the frontend code, but the the experimenter type parameter must additionally be declared (3 for tablet, 4 for human). Furthermore, the inertial measurement unit, IMU, armband must be powered on. The antenna must be connected to a usb port and the laptop connected to its WIFI network. The IMU application must be installed on the laptop.

3.15 After Participant Arrival

Once the participant arrives they must sign the consent form and fill out the demographic form. The procedure is then explained to them. They will be told to keep their elbow on the red cross on board and to push the puck into the goal zone. It will be explained to them that they must do this for 200 trials, take a 5 minute break, and then do 5 retention trials. They are also allowed to take a break during the 200 trials if they so desire. They will be allowed to view the zones on the board and the type of feedback will be explained to them. They will be given an opportunity to do 5 practice trials where the human supervisor will give them feedback for each trial so they understand the experiment process and have an opportunity for questions. After they have finished the practice trials they will be instructed to place the IMU
armband along the front of their dominant arm’s elbow, so that it rests comfortably, centered on the outside of their elbow as depicted in Figure 3.24.

![MindRove Armband Positioning](image)

**Figure 3.24: MindRove Armband Positioning**

Once the participant is ready to begin, the human supervisor must hit record on the software used to record the IMU data. All relevant data must be saved to the Participant’s ID number folder. The frontend file must then be started first and then the main file executing backend next. If the experimenter is the tablet for the session then the human supervisor will only need to collect the puck on the board after the trial is over. The human experimenter stands behind the partition and when they hear the audio dialogue saying “Trial Over” they walk around the partition to the board and pick up the puck, and place it onto the tabletop within the participants reach.

If the experimenter is the human however, then in addition to picking up the pucks after each trial they have a few more tasks. They must give the participant the introductory, retention, and concluding messages as well as inform the participant when the trial begins and ends and their feedback. This dialogue is identical to that which the tablet gives and is attached in Appendix A.3. During the trials, the frontend will be printing the dialogue in the console for when the trial starts and ends,
and the designated feedback, so the human coach can glance at that text and read it aloud. This is to minimize errors as they do not need to keep track of the timing of the trials or order of steps independently.

3.16 Post Session Survey

After the participant completes all the trials, they take a survey. The survey is on a Google Form, and the participant enters their Participant ID number to begin. One section of the survey utilizes the System Usability Scale. This is an accredited scale that consists of 10 questions that are used to help assess the usability of a system. The second part of the survey is a section of supplemental questions about the difficulty of the task and the feedback. The survey is included in Appendix A.9.

3.17 Scoring of Trials

After the trials are conducted, a definitive method for discerning a participant’s progress is needed. Therefore, each zone will be assigned a point score as shown in Table 3.2. The zones closer to the Goal zone are worth more points.

<table>
<thead>
<tr>
<th>Zones</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>GOAL</td>
<td>6</td>
</tr>
<tr>
<td>6 and 5</td>
<td>5</td>
</tr>
<tr>
<td>7 and 4</td>
<td>4</td>
</tr>
<tr>
<td>8 and 3</td>
<td>3</td>
</tr>
<tr>
<td>9 and 2</td>
<td>2</td>
</tr>
<tr>
<td>10 and 1</td>
<td>1</td>
</tr>
<tr>
<td>Out of Bounds</td>
<td>0</td>
</tr>
</tbody>
</table>
For each participant, after their trials are completed their total points for the session will be added to get their final score. All participant’s scores in an individual group will be totaled. The four categories that the points can possibly go to are: human coach KR, human coach KP, SAA coach KR, and SAA coach KP. The points accumulated per category will be compared to determine if the performance between groups differentiates at all. Additionally, each participant’s error during the initial trials versus the retention trials can be tabulated and compared, in order to determine whether this task can be considered a learning task. Rates of learning can also be analyzed and compared. For specifics on future study development please refer to Appendix A.10.
4.1 Introduction

This chapter discusses the pilot experiment. The purpose of the pilot was to judge the usability and cohesiveness of the experiment for participants as well as the human overseeing the experiment. This chapter will include task performance data, relevant IMU data, and survey results.

4.2 Participant Task Performance

Four participants (one for each level of each independent variable) participated. Participant 01 (P01) received the human-KP condition; P02 received the tablet-KR condition; P03 received the human-KR condition; and P04 received the tablet-KP condition. Task performance is visualized using histograms, where 200 trials are binned into the relevant zones and areas. Zones 1 through 10 refer to those of Figure 3.2. These histograms will be analyzed in the next chapter to determine task difficulty.
Figure 4.1: Participant Performance Histograms

The areas represented include: OB(f) (out of bounds before the zones); 1-5 (the zones immediately before the goal); goal (the target area); 6-10 (the zones immediately after the goal); OB(r) (out of bounds at the rear); ND (puck not detected).
4.3 Participants Residual Error

In order to determine whether participants improved through trials, we can calculate their residual error. Residual error informs how much the experimental value (the zone in which the puck lands) differentiates from the target value (the goal zone). The residual errors before and during Retention trials are plotted in the following figures. The error plots for the first 200 trials are shown for each participant. An additional error plot for the retention trials is also depicted. The negative $y$-axis values on the plot indicate the puck was located in an area before the goal zone while the positive error values indicate the puck was located in an area after the goal zone.

![Figure 4.2: Participant 01 Error vs. Trials](image)

Figure 4.2: Participant 01 Error vs. Trials
Figure 4.3: Participant 01 Retention Trial Error

For P01, the absolute residual error average for the 200 trials was 23.9343 cm. The absolute residual error average for the Retention trials was 41.8436 cm.

Figure 4.4: Participant 02 Error vs. Trials
For P02, the absolute residual error average for the 200 trials was 23.1305 cm. Their absolute residual error average for the Retention trials was 15.2287 cm.
For P03, the absolute residual error average for the 200 trials was 24.5658 cm. Their absolute residual error average for the Retention trials was 19.0712 cm.
For P04, the absolute residual error average for the 200 trials was 22.9511 cm. Their absolute residual error average for the Retention trials was 24.1851 cm.

### 4.4 IMU Data

Each participants’ IMU data were recorded during all trials. The armband was placed on the dominant arm, distal/anterior to the olecranon. The sensor recorded linear accelerations in the $x$, $y$, and $z$ axes as well as angular velocities around the axes (see Figure 4.10).
Two peaks from each participant’s linear acceleration and angular velocity are be presented in this section. One set is from an early trial (prior to trial 50) and the other from the participant’s last Retention trial. The pulse width for each plot was also found manually, by searching for where plot is flat immediately before and after the profile and evaluating those time values. These two times are subtracted to get the pulse width.
Figure 4.11: P01 Angular Velocity and Linear Acceleration vs. Time

Figure 4.11 displays Participant 01 data for the trials. Participant 01 had the greatest angular velocity about the $y$-axis. The maximum acceleration occurred in the direction of the $z$-axis. The pulse width is 1434 ms.
This Retention trial data set for Participant 01 is displayed in Figure 4.12. The pulse width is 2970 ms.
Figure 4.13 displays Participant 02 data for the trials. Participant 02 had the greatest angular velocity about the $x$-axis. The maximum acceleration occurred in the direction of the $z$-axis. The pulse width is 566 ms.
Figure 4.14: P02 Retention Angular Velocity and Linear Acceleration vs. Time

This Retention trial data set for Participant 02 is displayed in Figure 4.14. The pulse width is 550 ms.
Figure 4.15: P03 Angular Velocity and Linear Acceleration vs. Time

Figure 4.15 displays Participant 03 data for the trials. Participant 03 had the greatest angular velocity about the y-axis. The maximum acceleration occurred in the direction of the z-axis. The pulse width is 392 ms.
Figure 4.16: P03 Retention Angular Velocity and Linear Acceleration vs. Time

This Retention trial data set for Participant 03 is displayed in Figure 4.16. The pulse width is 420 ms.
Figure 4.17: P04 Angular Velocity and Linear Acceleration vs. Time

Figure 4.17 displays Participant 04 data for the trials. Participant 04 had the greatest angular velocity about the $y$-axis. The maximum acceleration occurred in the direction of the $z$-axis. The pulse width is 352 ms.
This Retention trial data set for Participant 03 is displayed in Figure 4.18. The pulse width is 530 ms.

4.5 Survey Results

Participants took a survey after they finished the Retention Trials using an online form. One section of the survey was the System Usability Scale. The SUS is a reliable, valid scale used to evaluate usable and unusable systems or products. It includes ten questions rated on a scale from strongly disagree to strongly agree [22]. The other section of the survey consisted of 6 supplemental questions.

The SUS questions include:
Q1: I think that I would like to use this system frequently.
Q2: I found the system unnecessarily complex.
Q3: I thought the system was easy to use.
Q4: I think that I would need the support of a technical person to be able to use this system.
Q5: I found the various functions in this system were well integrated.
Q6: I thought there was too much inconsistency in this system.
Q7: I would imagine that most people would learn to use this system very quickly.
Q8: I found the system very cumbersome to use.
Q9: I felt very confident using the system.
Q10: I needed to learn a lot of things before I could get going with this system.

The supplemental questions include:

Q1: How frequently did you look at the experimenter?
Q2: Feedback was not provided on every trial. Did you feel the feedback was too frequent, just right or not frequent enough?
Q3: How useful was the verbal feedback provided by the experimenter?
Q4: How challenging was the task?
Q5: Did your performance improve during the experiment?
Q6: If you answered 'Yes' to the prior question, what helped you to improve your performance?

For the survey Likert scale questions for the SUS section, 1 represents strong disagreement, while 5 represents strong agreement. Refer to Table 4.1 for participant responses to the SUS survey questions.
Table 4.1: Number of Respondents for SUS Likert Scale Questions

<table>
<thead>
<tr>
<th>Response</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td></td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q2</td>
<td></td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q3</td>
<td></td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q4</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q5</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q7</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q8</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q9</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q10</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We also included supplemental questions, with responses presented below. Supplemental Question 1 asked “How frequently did you look at the experimenter?”

![Figure 4.19: Supplemental Question 1](image)

Supplemental Question 2 asked “Feedback was not provided on every trial. Did you feel the feedback was too frequent, just right or not frequent enough?”

![Figure 4.20: Supplemental Question 2](image)
Supplemental Question 3, asked how useful the verbal feedback provided by the experimenter was. The rating 1 indicates “Not at all useful”, while 7 indicates “Very Useful.” Similarly, Supplemental Question 4 asked “How challenging was the task?” The score of 1 represents “Not challenging at all” while the score 7 represents “Very challenging.”

<table>
<thead>
<tr>
<th>Response</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q3</td>
<td></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q4</td>
<td></td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Supplemental Question 5 asked whether participants felt they improved during the experiment.

![Figure 4.21: Supplemental Question 5](image)

If participants answered “yes” to question 5, they were asked what helped them to improve their performance for Supplemental Question 6. The three responses given are displayed below:

- Feedback (both of the responses and of the sound of the puck rolling). I wish I could have had this verbal feedback every time.
- The prospect of reaching the goal zone motivated me throughout the experiment
- Focusing on repeatability
5.1 Introduction

This chapter describes the extent to which the system met our design goals, as contextualized by participant performance. Participant results are examined to determine whether this motor task is a learning task. IMU data were also analyzed to determine if participants performed their movements as expected. This chapter also discusses current limitations of the system and possible solutions, as well as other data analyses that can be performed with a larger sample size.

5.2 System Data Collection

The system was created with several goals in mind. It must ensure accurate determination of puck locations (i.e., that the camera takes pictures at specific times and for those images to be processed and analyzed automatically). The system was successful in keeping the correct timing of the pictures, as that remained constant for all participants. Regarding the accuracy of the processed picture data, it was able to get the correct zone a majority of the time. Approximately two to three times for each participant (among the 200 trials) it would return that there was “no puck detected” when there was a puck on the tabletop. I believe this is due to a combination of factors; the resolution of the camera, the lighting of the room, and the puck color. I believe if a better resolution camera were used and the threshold of the color of the puck in the Python camera code were more fine tuned, this issue may be addressed.
The system also successfully collected the trial number, puck location in pixel values, zone number, and feedback. Every trial has this complete set of data for all participants. Additionally, the sensor data (i.e. angular velocities and linear accelerations) were collected without issues. The online survey form was also completed without issues. It took participants less than 10 minutes to complete.

5.3 Participant Zone Frequency

For every trial, the resulting zone the puck was located in was recorded. The histograms of this data for each participant are located in section 4.2, Figure 4.1. As can be observed for every participant, the category with the highest frequency was the “0” or fail category. A trial is considered a fail if the puck is not detected, or it is out of bounds before or after the zones. Each participant had at least 40 percent of their trials result in a failure. This indicates that the task as it is now may not be a learnable task, and the design may require change.

The aspects of the physical design that may influence this result are the dimensions of the tabletop, and the puck itself. While testing, it was noted that the pucks were found to be quite light and easy to push, which may lead to participants frequently losing control over the puck. It is easy to overestimate the force needed to push the puck, and afterwards the participants may overcompensate by pushing the puck too softly. Unfortunately, the out of bounds area before the zones is a relatively large area when compared to that of the individual zones, so pushing softly will not get the puck far into the zone territory if at all. This may also explain why the data shows participants generally had a higher frequency of pushing their puck into the zones before the goal (Zones 1–5).
Another important observation to note is that although all participants highest frequency result was a fail, the second highest was the goal zone. This indicates that there is still potential learning with this motor task. Another way to verify is the learnability is to compare performance during the trials and then during the retention trials.

5.4 Residual Error Before and After Retention Trials

The residual errors for each trial were plotted in section 4.3. Each participants’ results and performance varied. There seems to be no apparent trend in the data as they vary across participants and the sample size is too small to determine any relationship. Additionally, because of this reason, the current pilot participant data cannot be used to determine a relationship between feedback, delivery medium, and task performance. If this task was of reasonable difficulty and induced learning, we would expect the error to decrease as trial number increases.

While data across participants cannot be compared, an individual participant’s performance during the trials can be assessed. The averages for the individual participants error were calculated for the trials and Retention trials. P02 and P03 had less error during the retention trials. However for P01 and P04, the error increased. This makes determining whether participants are getting better difficult as the results are inconclusive with such a small sample.

5.5 IMU Sensor Data Analysis

An IMU armband was placed on each participant’s dominant forearm, anterior to the olecranon. Participants were instructed to keep their elbows at a fixed location on the
tabletop. Though they were allowed to move their forearm and wrist, they were instructed not to use their shoulders when pushing the puck. Participants did abide by these instructions for the majority of the experiment but occasionally moved their elbows from the marked tabletop location momentarily. Participants used a multi-joint reaching and pushing motion to move the puck. This type of motion was analyzed in a study by McCrea et al., where it was found that a linear velocity profile we expect for a multi-joint reaching motion is a smooth bell shaped profile [25]. Although our sensor records angular velocities rather than linear velocities, we should expect such profiles as the existing literature by McCrea suggests those velocity profiles are generally representative of a Bi-directional reach-to-grasp motion.

Because we are measuring angular velocity we expect to see a positive then negative peak (similar to a single period of a sinusoidal wave). In theory a sinusoidal profile represents the change in angle that the IMU experiences during a single trial. P01, P02 and P04 very clearly display a sine wave. This indicates the participants sped up as they rotated their arm to execute the pushing motion, and rotated it in the opposite direction to get it back into the initial position. P03’s data is more bell-shaped, indicating their rotation of their arm was more limited. These results indicate there can be a variety in the pushing motion.

The direction of acceleration demonstrating the greatest amplitude was the z-axis. This is expected as the z-axis points outward from the sensor and perpendicular to the participant’s forearms. They also resembled impulse plots as the acceleration magnitude quickly increases and then decreases. The pulse widths for participants also varied. For three participants, the pulse widths increased during Retention trials. For the remaining participant, their pulse widths decreased. Generally, pulse width increases as task difficulty decreases. Nonetheless, these data suggest the IMU is sensitive to such changes.
With the ideal sample size, these data can be used to relate movement kinematics to successful and failed pushes. Pulse height, pulse width, and the number of peaks in pulses can be used to potentially differentiate better and worse performance.

5.6 Survey

Each participant took a survey post Retention Trials. Participants said they had difficulty answering some of the System Usability Scale questions as many did not feel applicable to their role, specifically those who had the human coach. For that reason although the SUS survey results are included, they have limited significance as they were not equally applicable to all groups, but are relevant as they are often used to analyze the quality of socially interactive agent systems. The Supplemental Questions however can still be applied to each group as they are more qualitative.

In regards to the first question that asked for the frequency at which participants looked at the coach, the results were mixed. It ranged from “Not at all” to “Only occasionally” as shown in Figure 4.19. Both coaches provided feedback with a form of audio dialogue so it was not necessary for participants to look at them. Furthermore, the coaches were positioned to the right side of the participant, and all the participants were right handed. This caused them to face more to the left to better position their hand for when they pushed the puck. A lack of viewing the coach may lower the effects embodiment had during this experiment.

The second question asked about the frequency of the feedback, shown in Figure 4.20. The results were split in half, with some believing they received the “Just right” amount while other believed it was “Not frequent enough.” There was a trend within participants when they received their augmented feedback. They would always overcompensate in their pushes for the trial after. If they were told they were in zone
1 or to push harder, they would push much too hard and into the out of bounds area after the zones for example.

For question 3, the score of 1 represented “Not at all useful” while the score 7 represented “Very useful.” The general consensus from these two questions were that the feedback was quite useful. For question 4, the score of 1 represented “Not challenging at all” while the score 7 represented “Very challenging.” All participants rated the difficulty either a 5 or 6 indicating it was very demanding. This strengthens the requirement for changing the design of the experiment. However, in question 5, three of four participants indicated that they felt they were improving during the task, and the last indicated they were unsure. It appears the general process of the system and experiment work, however it could be made slightly easier. The last questions asked what led to improvement for the task if they answered yes on the last question and one participant mentioned the feedback and how they wanted it every trial. In order to make the task easier, the rate of feedback could be slightly increased. Another mentioned they liked the amount of repeatability while the last was motivated by the goal.

5.7 Limitations of the System Design

During the pilot evaluation, there were a total of 6 participants, however only the data for 4 of them was used. That is because issues occurred during their trials. One participant accidentally knocked over a board that was serving as an extra partition to block them from view of the camera. Because the system works by taking a reference image and then a subsequent image during a trial and then subtracts them, this caused a major issue. Because the board’s location was altered, the board would always register as a changed pixel because of its slightly different position. Since the
max and min $y$ pixels were averaged to obtain the puck location, this would result in skewed data. Therefore it did not make sense to continue the trials. A solution for this is to fixture the partition more, so that it is sturdy. For the second participant, it was a clerical error. The trial number parameter that was input in the code was not the 200 trials but instead a different value. That made this participant’s data different from the others, making it unusable. In the future these code parameters will be double checked before starting the experiment with the participant.

5.8 Experimental Conclusions and Recommendations

Ultimately, the system that was designed to investigate the interaction between feedback and medium type for a novel motor task shows some initial signs that it may increase the user’s performance, however it needs several changes as it is too difficult for users in its current state. The physical setup must be changed so that the task is not as difficult which may be done by decreasing the out of bounds front area. Additionally, rather than every third trial for receiving feedback, it can be changed to every other trial. Furthermore, the light-weight pucks can be changed out for heavier pucks so that it is harder for the participant to lose control and they can better guide their pucks to the goal zone. Lastly, the system can be made more robust by improving fixturing of partitions and other physical components.

Another component of the system that can be improved is the IMU placement. In the experiment to evaluate system performance, the armband was manually placed on the dominant arm, anterior to the olecranon. However, it is very easy to have the IMU position slightly vary between participants, which can add noise to data. In a study by McGrath, the method of using Principle Component Analysis (PCA) was proposed to calibrate a knee axis estimator [26]. This approach can possibly be
adopted in later experiments to lessen the chance of noise in the data. In the method, the angular velocity vector was found between the IMU and several positions of the limb. Principle Component Analysis, which is a method to statistically simplify high dimensioned data to a lower dimension, was then used [26, 27]. In the study, the knee axis was estimated with PCA. The analysis in this work can be applicable to placing the IMU on the forearm and used in this system.

5.9 Future Applications

This thesis and designed system and experiment can be used as the basis for other future experimentation. It incorporates different delivery medium, types of augmented feedback, and a motor task. In the future, other motor tasks can be used in place of the shuffleboard task. Other delivery mediums, such as a robot, can also be used. With a large enough sample size, the relationship between delivery medium and feedback can be explored to discover how they affect user performance and find the combination that optimizes user performance the most for a specific task.
Stroke dramatically decreases the quality of life in individuals it affects. It may significantly impact a person’s motor control, making tasks that were once easy considerably more difficult [5]. Through therapeutic interventions such as Constraint-Induced Movement Therapy (CIMT), stroke patients can regain some of their lost motor control [2]. However, stroke is a chronic illness and needs continual treatment and significant resources, but therapy and other treatments are expensive [6]. A Socially Assistive Agent (SAA) is a tool that can extend the effect of the therapist and facilitate long-term care for stroke patients.

SAA has been used previously in rehabilitative contexts to help motivate patients through exercises [8]. It can help by providing augmented feedback and has had varying levels of effectiveness as the type of feedback (Knowledge of Results or Knowledge of Performance) influences the participant’s performance. Another factor that influences performance is delivery medium, through embodiment. There is a general lack of detail on pairing of feedback and delivery medium that optimizes user performance the most.

This thesis sought to conceive a system and procedure that can test different types of feedback and delivery medium. In particular it tested KR and KP feedback, with delivery media (a tablet and human coach) for a shuffleboard task. Through testing the system it was found that although the task was difficult, there were some initial signs of the potential to increase the user’s performance, as all participants’ second highest frequency for zone location was the goal zone. Data with the IMU armband
was also successfully taken, meaning with a larger sample size, the kinematics related to successful and failed pushes can be determined. For instance, the velocity profiles, acceleration profiles, and pulse widths can be examined to determine a relationship between them and a successful trial.

This shuffleboard task is just one example of a motor task that can be incorporated into CIMT for stroke patients, which is the long term goal of the system. The system can be used as a framework and modified for a multitude of other motor tasks, as its integration of a vision system, control circuit, and IMU sensor is modular. The system collects data on the participant through those components and records the results automatically, which was successfully shown through this pilot system evaluation with participants. Furthermore, the system can be modified to test other delivery media and adopted for future experimentation to investigate the relationship between feedback and media for specific motor tasks. Once the optimal conditions are discovered, the system can continue to be used to help individuals with their therapeutic interventions. It can ultimately help extend the effects of therapy and help patients recover.

   Y. Bleyenheuft, “Rehabilitation of Motor Function after Stroke: A Multiple
   Systematic Review Focused on Techniques to Stimulate Upper Extremity
   Central:PMC5020059 [DOI:10.3389/fnhum.2016.00442]

   A. Gheorghe, and F. D. Muresanu, “The economic burden of stroke: a
   systematic review of cost of illness studies,” J Med Life, vol. 14, no. 5,

   https://www.cdc.gov/stroke/about.htm, Control and Prevention,
   September 2022.

   https://www.hopkinsmedicine.org/health/conditions-and-
   diseases/stroke/effects-of-stroke, ”Accessed 10 September 2022”.

   F. Deruyter, J. J. Eng, B. Fisher, R. L. Harvey, and et al., “Guidelines for

   A. P. Carson, A. M. Chamberlain, A. R. Chang, S. Cheng, S. R. Das, and


[19] “Introduction to opencv-python tutorials.”

  https://docs.opencv.org/4.x/d0/de3/tutorial_py_intro.html.


APPENDICES

Appendix A

METHODS
A.1 Consent Form
INFORMED CONSENT TO PARTICIPATE IN A RESEARCH PROJECT:
“The Effects of Augmented Feedback and Delivery Modality on Performance of an Upper–Extremity Motor Task”

INTRODUCTION
This form asks for your agreement to participate in a research project on performance feedback and communication interfaces. Your participation involves performing repeated tasks with your dominant arm in a lab setting while wearing a wrist sensor. You will be asked to push a small object towards a target area on a flat surface. You will be provided feedback about how closely the object is to the target by a human, robot, or tablet-based computer program. It is expected that your participation will take approximately one hour. The potential risks from this project are typical of what would be expected when performing a repeated task (e.g., fatigue) and loss of confidentiality (e.g., your data could be made public). In the long term, this research may contribute to a better understanding of how to design technologies for people living with movement disorders. If you are interested in participating, please review the following information.

PURPOSE OF THE STUDY AND PROPOSED BENEFITS

• The purpose of the study is to determine if the type of feedback and the interface through which it is delivered (human, robot, tablet) affects movement performance.

YOUR PARTICIPATION

• If you agree to participate, you will be asked to provide demographic information and to attend a single data collection session. During this session, you will perform multiple trials of a goal directed task. For this task, you will push an object towards a target on a flat surface. You will not see the object after it is pushed, but will be provided with feedback on how well you performed (i.e., how close you were to the target). You will be monitored using a motion sensor (worn on the wrist) and a camera. You will also be asked to fill a survey at the end of your data collection session.

• Your participation will take approximately one hour on the Cal Poly campus (Building 13).

PROTECTIONS AND POTENTIAL RISKS

• Please be aware that you are not required to participate in this research, refusal to participate will not involve any penalty or loss of benefits to which you are otherwise entitled, and you may discontinue your participation at any time.

• The possible risks or discomforts associated with participation in this study are typical of performing tasks with your arm, such as fatigue.

• Your confidentiality will be protected by ensuring that only study personnel have access to your data. Additionally, you will be assigned a participant ID number. Other than the consent forms, all data will use the participant ID to limit potential exposure of personal information.

• Any collected data will be destroyed approximately five years after study completion (Dec. 31, 2027). This will allow for ample time to analyze and publish study results. If data sharing is required for publication, only de-identified data (free of information that ties it to your name) will be shared.

RESOURCES AND CONTACT INFORMATION

• If you should experience any negative outcomes from this research, please be aware that you may contact the research team including: Eric Wade, PhD (617-308-0498).
• This research is being conducted by Dr. Eric Wade in the Department of Mechanical Engineering at Cal Poly, San Luis Obispo. If you have questions regarding this study or would like to be informed of the results when the study is completed, please contact the researcher(s) Eric Wade, PhD (erwade@calpoly.edu, 617-308-0498).

• If you have concerns regarding the manner in which the study is conducted, you may contact Dr. Michael Black, Chair of the Cal Poly Institutional Review Board, at (805) 756-2894, mblack@calpoly.edu, or Ms. Trish Brock, Director of Research Compliance, at (805) 756-1450, pbrock@calpoly.edu.

**AGREEMENT TO PARTICIPATE**

If you are 18 or older and agree to voluntarily participate in this research project as described, please indicate your agreement by signing below. Please retain a copy of this form for your reference, and thank you for your participation in this research.

<table>
<thead>
<tr>
<th>Signature of Volunteer</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Signature of Researcher</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A.2 Demographic Form
**Participant Demographic Form**

Date: ____________________

Name: ____________________

Age: ____________________

Height: ____________________

Gender: Male _____ Female _____ Transgender _____
        Non-binary _____ Prefer not to respond _____

Arm dominance: L _____ R _____

Any surgery on the dominant arm?
A.3 Experimenter Dialogue
**Experiment Dialogue**

**Tablet Introduction**
Hello, I am Ash, your virtual assistant. Today I will be coaching you as you participate in this exercise. The goal of this exercise is to push the puck into the designated GOAL zone. As you can see from the table, the goal zone is near the middle of the desk surface, and is sandwiched between numbered zones. You will be unable to see the goal zone, but I will give you feedback every third trial as you do the exercise to help correct your aim. Please begin when you hear trial start.

**Feedback specific to KR**
Your puck is in zone 1.
Your puck is in zone 2......
......Same dialogue continued up through zone 10
Your puck is out of bounds before the zones
Your puck is out of bounds after the zones

**Feedback Used in Both**
Trial Start
Trial Over
You reached the goal zone. Good job!
No puck was detected.
Break time started
Break time over

**Feedback specific to KP**
Push a lot harder. For when puck is out of bounds in front of the zones
Push a lot softer. For when puck is out of bounds behind the zones
Push harder. For zones 1 - 5
Push softer. For zones 6-10
Retention soon
We will take a 5 minute break then start the Retention trials.

Retention Trials Beginning
Please prepare for the retention test. This test evaluates how much you have retained from practice. You will perform five trials during which you will receive no feedback. As with your prior set of trials, you will be prompted when it is time to push the puck.

Concluding Dialogue
Thank you for participating. This concludes the exercise.
A.4 Camera Module
This class is used to initialize the camera, and subsequently take pictures, then return values for min and max of difference between images.

To use:
```python
>> import snp
>> mycam = snp.snapper() ; Create instance
>> mycam.setup() ; Initialize the camera
>> mycam.blank() ; Take the reference image
>> thing = mycam.curnt() ; Take subsequent images; min/max values returned
>> mycam.close() ; Close, release camera
```
ymax = max(y_vals)
img_name2 = "opencv_frame_{}.png".format(self.counter)
cv2.imwrite(img_name2, self.frame)
# print("{}" written!".format(img_name2))
self.counter += 1
return xmin,xmax,ymin,ymax

except ValueError:
    xmin = -1
    xmax = -1
    ymin = -1
    ymax = -1

def closer(self):
    ""
    Close camera and release
    ""
    self.cam.release()
    cv2.destroyAllWindows()
A.5 Frontend
```python
import serial
import time
import snp
from snp import snapper
import csv
import shares

ser = serial.Serial(port='COM3', baudrate=115200, timeout=1)

import sys

class SAAFront:
    '''
    @brief      A finite state machine to control windshield wipers.
    @details    This class implements a finite state machine to control the
                operation of windshield wipers.
    '''

    ## Constant defining State 0 - Initialization
    S0_INIT = 0

    ## Constant defining State 1 - Trial
    S1_TRIAL = 1

    ## Constant defining State 2 - Break
    S2_BREAK = 2

    ## Constant defining State 3 - Retention Start
    S3_RETENTION_START = 3

    ## Constant defining State 4 - Retention Test
    S4_RETENTION_TEST = 4

    ## Constant defining State 5 - End
    S5_END = 5

    def __init__(self, test_type, experimenter_type):
        #input parameters:
        #test_type is KR (1) or KP(2) and experimenter tablet (3) or human (4)
        ...
        @brief          Creates an object.
        ...
        
        ## The state to run on the next iteration of the task.
        self.state = self.S0_INIT
        self.runs = 0
        self.interval = 1

        ## The timestamp for the first iteration
        self.start_time = time.time()

        ## The "timestamp" for when the task should run next
        self.next_time = self.start_time + self.interval
        self.counter = 0
        self.trial_total = 200 # trial limit
        self.trial_totalr = 205 #Retention trial limit
        self.avglist = []
        self.trialnumber = []
```

```python
self.zonenumber = []
self.study_param = []

self.mycam = snp.snapper()  # create camera object
time.sleep(1)
self.mycam.setup()  # initialize camera
time.sleep(1)
self.mycam.blank()  # take reference camera picture
time.sleep(1)

self.TrialStatus = 0
self.current_time = 0
self.present_time=0
self.fcount = 0
self.firsttrial=1
self.experimenter = experimenter_type
self.test_type = test_type
self.feedbacklist=[]

def run(self):
    '''
    @brief      Runs one iteration of the task
    '''
    self.curr_time = time.time()
    if self.curr_time > self.next_time:
        if(self.state == self.S0_INIT):
            if ser.read().decode('ascii') == "s":  # if s received from backend,
                commence
                print('Intro Message ')
                time.sleep(30)
                self.study_param.append(self.experimenter)
                self.study_param.append(self.test_type)
                self.transitionTo(self.S1_TRIAL)
            else:
                self.transitionTo(self.S0_INIT)

        elif(self.state == self.S1_TRIAL):# Run State 1 Code
            if self.counter < self.trial_total:  # if trial limit not exceeded
                if ser.read().decode('ascii') == "f":  # if f received from backend
                    print("Break time started")
                    self.transitionTo(self.S2_BREAK)  # transition to break state
                else:
                    status = "t"
                    ser.write(str(status).encode('ascii'))  # send t to backend
                    indicating trial started
                    time.sleep()
                    print(""Trial Start"")
                    time.sleep(7)
                    thing = self.mycam.curnt()  # take puck picture
                    completion = "v"
                    ser.write(str(completion).encode('ascii'))  # send v to backend
                    to indicate pic taken
```
print('"Trial Over"')
time.sleep(4)

if thing is None:  # if no changes detected
    self.Yavg = -1 # set pixel x and y variables to -1
    self.Xmin = -1
    self.Xmax = -1
    self.Ymin = -1
    self.Ymax = -1
else:
    self.Xmin = thing[0]
    self.Xmax = thing[1]
    self.Ymin = thing[2]
    self.Ymax = thing[3]

self.zonenumber.append(3)
print('z: 3')
ser.write(('c').encode('ascii'))

else:

    if self.Yavg >= 242:
        # append this zone value
        print('z: 3')
        ser.write(('c').encode('ascii'))
    else:
        # set feedback counter to 1
        self.fcount = 1
        self.firsttrial = 0
        self.fcount += 1
        self.fcount -= 1

    if self.fcount == 0:
        # reset counter
        self.fcount = 1
        if self.test_type == 1:
            # if input parameter is KR feedback
            print('"KR: Your puck is in Zone 1"')
            else:
                # if input parameter isn't KR feedback, then it must be KP feedback
                print('"KP: Push harder"')

elif self.Yavg > 193.5 and self.Yavg < 209.5:  # if y averaged value is in this region
    print('z: 2')
    ser.write(('b').encode('ascii'))
    if self.firsttrial == 1:
        self.fcount = 0
        self.firsttrial = 0
        self.fcount += 1
        self.fcount -= 1

    if self.fcount == 0:
        # reset counter
        self.fcount = 1
        if self.test_type == 1:
            # if input parameter is KR feedback
            print('"KR: Your puck is in Zone 2"')
            else:
                # if input parameter isn't KR feedback, then it must be KP feedback
                print('"KP: Push harder"')

elif self.Yavg > 209.5 and self.Yavg < 225.5:
    print('z: 3')
    ser.write(('c').encode('ascii'))
    if self.firsttrial == 1:
        self.fcount = 0
        self.firsttrial = 0
        self.fcount += 1
        self.fcount -= 1

    if self.fcount == 0:
        # reset counter
        self.fcount = 1
        if self.test_type == 1:
            # if input parameter is KR feedback
            print('"KR: Your puck is in Zone 3"')
            else:
                # if input parameter isn't KR feedback, then it must be KP feedback
                print('"KP: Push harder"')
if self.firsttrial==1:
    self.fcount=1
    self.firsttrial=0
    self.fcount= self.fcount -1
elif self.Yavg>=311.5 and self.Yavg<330:
    print ('"KP: Push harder"')
else:
    print ('"KP: Push harder"')

elif self.Yavg>=242 and self.Yavg<258.5:
    self.zonenumber.append(4)
    print('z: 4')
    ser.write(('"z: 4"').encode('ascii'))
    if self.firsttrial==1:
        self.fcount=1
        self.firsttrial=0
        self.fcount= self.fcount -1
    if self.fcount==0:
        self.fcount=3
        if self.test_type == 1:
            print ('"KR: Your puck is in Zone 4"')
        else:
            print ('"KP: Push harder"')

elif self.Yavg>=258.5 and self.Yavg<276:
    self.zonenumber.append(5)
    print('z: 5')
    ser.write(('"z: 5"').encode('ascii'))
    if self.firsttrial==1:
        self.fcount=1
        self.firsttrial=0
        self.fcount= self.fcount -1
    if self.fcount==0:
        self.fcount=3
        if self.test_type == 1:
            print ('"KR: Your puck is in Zone 5"')
        else:
            print ('"KP: Push harder"')

elif self.Yavg>=276 and self.Yavg<311.5:
    self.zonenumber.append('®')
    print('z: ®')
    ser.write(('"z: ®"').encode('ascii'))
    if self.firsttrial==1:
        self.fcount=1
        self.firsttrial=0
        self.fcount= self.fcount -1
    if self.fcount==0:
        self.fcount=3
        if self.test_type == 1:
            print ('"KR: You reached the goal zone. Good job!"')
        else:
            print ('"KP: You reached the goal zone. Good job!"')

elif self.Yavg>=311.5 and self.Yavg<330:
self.zonenumber.append(6)
print('z: 6')
ser.write(('g').encode('ascii'))
if self.firsttrial==1:
    self.fcount=0
    self.firsttrial=0
self.fcount= self.fcount -1
if self.fcount==0:
    self.fcount=3
    if self.test_type == 1:
        print ("KR: Your puck is in Zone 6")
    else:
        print ("KP: Push softer")
else:
    print ("KP: Push softer")

elif self.Yavg>=330 and self.Yavg<349.5:
    self.zonenumber.append(7)
    print('z: 7')
    ser.write(('h').encode('ascii'))
    if self.firsttrial==1:
        self.fcount=1
        self.firsttrial=0
        self.fcount= self.fcount -1
    if self.fcount==0:
        self.fcount=3
        if self.test_type == 1:
            print ("KR: Your puck is in Zone 7")
        else:
            print ("KP: Push softer")
else:
    print ("KP: Push softer")

elif self.Yavg>=349.5 and self.Yavg<368:
    self.zonenumber.append(8)
    print('z: 8')
    ser.write(('i').encode('ascii'))
    if self.firsttrial==1:
        self.fcount=1
        self.firsttrial=0
        self.fcount= self.fcount -1
    if self.fcount==0:
        self.fcount=3
        if self.test_type == 1:
            print ("KR: Your puck is in Zone 8")
        else:
            print ("KP: Push softer")
else:
    print ("KP: Push softer")

elif self.Yavg>=368 and self.Yavg<388:
    self.zonenumber.append(9)
    print('z: 9')
    ser.write(('j').encode('ascii'))
    if self.firsttrial==1:
        self.fcount=1
        self.firsttrial=0
        self.fcount= self.fcount -1
    if self.fcount==0:
        self.fcount=3
        if self.test_type == 1:
            print ("KR: Your puck is in Zone 9")
        else:
            print ("KP: Push softer")
else:
    print ("KP: Push softer")
elif self.Yavg>=388 and self.Yavg<408.5:
    self.zonenumber.append(10)
    print('z: 10')
    ser.write('k'.encode('ascii'))
    if self.firsttrial==1:
        self.fcount=1
        self.firsttrial=0
        self.fcount=self.fcount - 1
    if self.fcount==0:
        if self.test_type == 1:
            print ('"KR: Your puck is in Zone 10"')
        else:
            print ('"KP: Push softer"')

elif self.Yavg== -1:
    self.zonenumber.append(-1)
    print('z: no puck')
    ser.write('l'.encode('ascii'))
    if self.firsttrial==1:
        self.fcount=1
        self.firsttrial=0
        self.fcount=self.fcount - 1
    if self.fcount==0:
        if self.test_type == 1:
            print ('"KR: No puck was detected"')
        else:
            print ('"KP: No puck was detected"')

elif self.Yavg< 193.5:
    self.zonenumber.append(-2)
    print('z: out front')
    ser.write('m'.encode('ascii'))
    if self.firsttrial==1:
        self.fcount=1
        self.firsttrial=0
        self.fcount=self.fcount - 1
    if self.fcount==0:
        if self.test_type == 1:
            print ('"KR: Your puck is out of bounds before the
zones"')
        else:
            print ('"KP: Push a lot harder"')

elif self.Yavg> 408.5:
    self.zonenumber.append(-3)
    print('z: out back')
    ser.write('n'.encode('ascii'))
    if self.firsttrial==1:
        self.fcount=1
        self.firsttrial=0
        self.fcount=self.fcount - 1
    if self.fcount==0:
        if self.test_type == 1:
            print ('"KR: Your puck is out of bounds after the
zones"')
        else:
            print ('"KP: No puck was detected"')
else:
    print('Trials Complete')
    self.transitionTo(self.S3_RETENTION_START)

elif(self.state == self.S2_BREAK):
    if ser.read().decode('ascii') == 'f':
        print('Break time Over')
        time.sleep(3)
        self.transitionTo(self.S1_TRIAL)
    else:
        self.transitionTo(self.S2_BREAK)

elif(self.state == self.S3_RETENTION_START):
    ser.write(('r').encode('ascii')) #write message to backend that retention will soon begin
    time.sleep(300) # 5min break
    print('retention starting message')
    ser.write(('q').encode('ascii')) #write retention beginning now message to backend
    time.sleep(20)
    self.transitionTo(self.S4_RETENTION_TEST)

elif(self.state == self.S4_RETENTION_TEST):
    if self.counter < self.trial_total:
        #if retention trial limit not exceeded
        status = 't'
        ser.write(str(status).encode('ascii')) #write message to indicate trial started to backend
        time.sleep(1)
        print('Trial Start')
        time.sleep(7)
        thing = self.mycam.current()  #take picture
        completion = 'v'
        ser.write(str(completion).encode('ascii')) #write message to indicate trial over to backend
        print('Trial Over')
        time.sleep(4)

        if thing is None:
            self.Ymin = -1
            self.Ymax = -1
            self.Xmin = -1
            self.Xmax = -1
        else:
            self.Xmin = thing[0]
            self.Ymax = thing[1]
            self.Ymin = thing[2]
            self.Xmax = thing[3]

        self.counter += 1
        print('Trial fin : '+ str(self.counter))
self.trialnumber.append(self.counter)

# print(self.Yavg)
self.avglst.append(self.Yavg)

if self.Yavg>=193.5 and self.Yavg<209.5:  
    self.zonenumber.append(1)  
    ser.write('a'.encode('ascii'))

elif self.Yavg>=209.5 and self.Yavg<225.5:  
    self.zonenumber.append(2)  
    ser.write('b'.encode('ascii'))

elif self.Yavg>=225.5 and self.Yavg<242:  
    self.zonenumber.append(3)  
    ser.write('c'.encode('ascii'))

elif self.Yavg>=242 and self.Yavg<258.5:  
    self.zonenumber.append(4)  
    ser.write('d'.encode('ascii'))

elif self.Yavg>=258.5 and self.Yavg<276:  
    self.zonenumber.append(5)  
    ser.write('e'.encode('ascii'))

elif self.Yavg>=276 and self.Yavg<311.5:  
    self.zonenumber.append('G')  
    ser.write('G'.encode('ascii'))

elif self.Yavg>=311.5 and self.Yavg<330:  
    self.zonenumber.append('H')  
    ser.write('H'.encode('ascii'))

elif self.Yavg>=330 and self.Yavg<349.5:  
    self.zonenumber.append('l')  
    ser.write('l'.encode('ascii'))

elif self.Yavg>=349.5 and self.Yavg<368:  
    self.zonenumber.append('I')  
    ser.write('I'.encode('ascii'))

elif self.Yavg>=368 and self.Yavg<388:  
    self.zonenumber.append('J')  
    ser.write('J'.encode('ascii'))

elif self.Yavg>=388 and self.Yavg<408.5:  
    self.zonenumber.append(10)  
    print('z: 10')
elif self.Yavg == -1:
    self.zonenumber.append(-1)
    print(‘z: no puck’)  
    ser.write(‘(-1).encode(‘ascii’))

elif self.Yavg < 193.5:
    self.zonenumber.append(-2)
    print(‘z: out front’)  
    ser.write(‘(‘-2’).encode(‘ascii’))

elif self.Yavg > 408.5:
    self.zonenumber.append(-3)
    print(‘z: out back’)  
    ser.write(‘(‘-3’).encode(‘ascii’))

    time.sleep(6)  # delay for experimenter picking up puck

    self.transitionTo(self.S4_RETENTION_TEST)

else:
    print(‘Retention Trials Complete’)  
    self.transitionTo(self.S5_END)

elif self.state == self.S5_END:
        self.mycam.clser()  # exit camera
        time.sleep(2)
        listdecode = ser.readline().decode(‘ascii’)  # read list sent from backend
        self.listdecode = listdecode
        if listdecode != ‘0’:  # if any key is read
            slist = listdecode.strip(‘[]\r\n’)  # strip special characters
            sslist = slist.split(‘,’)
            for n in range(len(sslist)):
                self.feedbacklist.append(str(sslist[n]))  # append previous list members to new list

        ser.close()

        while True:
            with open(‘SAAdata.csv’, ‘w’, newline=‘’) as datafile:
                # save data to csv file
                file = csv.writer(datafile)
                file.writerow([self.S4_trials, self.zonenumber, self.
                                avglist, self.feedbacklist])
                break

        sys.exit(‘All Trials have been completed and data recorded’)  

else:
    pass

    self.runs += 1

    # Specifying the next time the task will run
    self.next_time = self.next_time + self.interval

    def transitionTo(self, newState):  # defines transitionTo state
        ‘’’
        @brief Updates the variable defining the next state to run
        ‘’’
        self.state = newState

    if __name__ == “__main__”:
#first num is feedback type - 1 is KR , 2 is KP
#second num is human or tablet - 3 is tablet, 4 is human

task1 = SAAFront(1,4)

for N in range(10000000000):
    task1.run()
A.6 Backend
import pyb
from pyb import UART
import utime
import micropython
import sys

micropython.alloc_emergency_exception_buf(200)  # for troubleshooting

myuart = UART(2)
uart3 = UART(3, 9600)
bButton = pyb.Pin(pyb.Pin.board.PA10, mode=pyb.Pin.IN)

## Create a timer object (timer 2) in general purpose counting mode
myTimer = pyb.Timer(2, prescaler=79, period=0x7FFFFFFF)

button = 0

class SAA:
   def myCallback(self):
      ...
Function called by interrupt to set button value to 1 to indicate it has been pressed.
@brief          Makes button variable global and set it to 1.

```python
global button
button = 1
utime.sleep(.5)
```

def __init__(self, test_type):
    #pulled from EncoderFSM.py lab
    
    @brief # pulled from EncoderFSM.py lab
    
    self.state = self.S0_INIT
    
    self.runs = 0
    
    self.interval = int(1e3)
    
    self.start_time = utime.ticks_us()
    
    self.next_time = utime.ticks_add(self.start_time, self.interval)
    
    self.receiveddata = []
    self.sentdata = []
    self.counter = 0
    self.test_type = test_type
    self.fcount = 3
    self.totaltrialnumber = 200  #200
    
    self.totaltrialnumberr = 205
    
    self.firsttrial = 1

    def run(self):
        
        @brief  Runs one iteration of the task
        
        global button

        self.curr_time = utime.ticks_us()
        if (utime.ticks_diff(self.curr_time, self.next_time) >= 0):
            ```
if(self.state == self.S0_INIT):
    # Run State 0 Code
    self.transitionTo(self.S1_WELCOME) #begins transition

elif(self.state == self.S1_WELCOME): #State 1
    # Run State 1 Code
    pyb.delay(10000)
    msg = "s"
    myuart.write(msg)
    pyb.delay(2000)
    self.sendBluetooth(13);
    self.transitionTo(self.S2_RECEIVE_DATA)

elif(self.state == self.S2_RECEIVE_DATA):
    if self.counter < self.totaltrialnumber: #if trial max not reached
        if button == 1: #if button pressed
            val = "f"
            myuart.write(val) #send val to frontend
            button = 0
            self.sendBluetooth(20) #send number to tablet via Bluetooth to play break started dialogue
            if myuart.any():
                clearmsg = myuart.readline().decode('ascii') #clear any message frontend may have sent
                self.transitionTo(self.S7_BREAK)
            else:
                myuart.any(): #if message received from frontend
                letter = ord( myuart.read().decode('ascii') ) #get value for character received
                if letter == 116:# value for letter t
                    self.sendBluetooth(22)##TRIAL START SOUND
                    self.transitionTo(self.S3_TRIAL_END)
                else:
                    self.transitionTo(self.S2_RECEIVE_DATA)
    else:
        self.transitionTo(self.S8_RETENTION_START)

elif(self.state == self.S3_TRIAL_END):
    if myuart.any():#if character received from frontend
        letter = ord( myuart.read().decode('ascii') )
        if letter == 118: # if characer value is 118 (v)
            self.sendBluetooth(23)##TRIAL END SOUND #bluetooth message to
            #play trial end sound
            self.transitionTo(self.S4_CONVERTING_DATA)
        else:
            self.transitionTo(self.S3_TRIAL_END)
    else:
        self.transitionTo(self.S3_TRIAL_END)

elif(self.state == self.S4_CONVERTING_DATA):
if myuart.any():  # if anything received from frontend
    self.counter += 1
    self.dist= myuart.readline().decode('ascii')  # read key
    self.distance= ord(self.dist)  # get unicode value of key

    if(self.test_type==1):
        self.transitionTo{self. S5_SEND_KR }
    elif(self.test_type==2):
        self.transitionTo{self. S6_SEND_KP}
    else:
        self.transitionTo{self. S4_CONVERTING_DATA}

elif(self.state == self.S5_SEND_KR):
    if self.firsttrial==1:  # if its the very first trial
        self.fcount=1  # set the fcount to 1 so that in later line when it
gets subtracted it reaches condition
        # to give feedback this is because feedback is given the very
        # first trial, and every third trial afterwards
        self.fcount=1
        self.firsttrial=0  # set first trial indicator to 0 so that code
doesn't enter this conditional statement again
        self.fcount= self.fcount -1  # each trial, subtract one from counter

    if self.fcount==0:
        self.fcount+=
        if self.distance == 97:  # check what message from
            # frontend was
            self.sendBluetooth(1);  # based on message, send Bluetooth
            # message
            pyb.delay(3000);  # delay to give human or tablet chance to play
            audio for feedback
            self.sentdata.append('Zone 1');
            self.transitionTo{self. S2_RECEIVE_DATA }

        elif self.distance == 98:
            self.sendBluetooth(2);
            pyb.delay(3000)
            self.sentdata.append('Zone 2');
            self.transitionTo{self. S2_RECEIVE_DATA }

        elif self.distance == 99:
            self.sendBluetooth(3);
            pyb.delay(3000)
            self.sentdata.append('Zone 3');
            self.transitionTo{self. S2_RECEIVE_DATA }

        elif self.distance == 100:
            self.sendBluetooth(4);
            pyb.delay(3000)
            self.sentdata.append('Zone 4');
            self.transitionTo{self. S2_RECEIVE_DATA }

        elif self.distance == 101:
            self.sendBluetooth(5);
            pyb.delay(3000)
            self.sentdata.append('Zone 5');
            self.transitionTo{self. S2_RECEIVE_DATA }

        elif self.distance == 102:
            self.sendBluetooth(11);
            pyb.delay(3000)
            self.sentdata.append('GOAL');
self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance == 103:
    self.sendBluetooth(6);
    pyb.delay(3000)
    self.sentdata.append("Zone 6");
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance == 104:
    self.sendBluetooth(7);
    pyb.delay(3000)
    self.sentdata.append("Zone 7");
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance == 105:
    self.sendBluetooth(8);
    pyb.delay(3000)
    self.sentdata.append("Zone 8");
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance == 106:
    self.sendBluetooth(9);
    pyb.delay(3000)
    self.sentdata.append("Zone 9");
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance == 107:
    self.sendBluetooth(10);
    pyb.delay(3000)
    self.sentdata.append("Zone 10");
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance == 108:
    self.sendBluetooth(11);
    pyb.delay(3000)
    self.sentdata.append("Out of Bounds front");
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance == 110:
    self.sendBluetooth(24);
    pyb.delay(3000)
    self.sentdata.append("Out of Bounds back");
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance == 108:
    self.sendBluetooth(18);
    pyb.delay(3000)
    self.sentdata.append("No puck detected");
    self.transitionTo(self.S2_RECEIVE_DATA)

else:
    self.transitionTo(self.S5_SEND_KR)
else:
    self.sentdata.append("NA"); # if no feedback for this
    # trial, append NA to list
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.state == self.S6_SEND_KP: # same order as part 5
    if self.firsttrial==1: # if its the very first trial
        self.fcount=1 # set the fcount to 1 so that in later
        # line when it gets subtracted it reaches condition
        # to give feedback
    # this is because feedback is given the very first trial,
    # and every third trial afterwards
self.firsttrial=0  # set first trial indicator to 0 so
# that code doesn't enter this conditional statement again
self.fcount=1
self.firsttrial=0  # State 1
self.fcount= self.fcount -1

if self.fcount==0:  # if the feedback counter is 0
    self.fcount=3  # reset counter

if self.distance==109:  # check what message from frontend was
    self.sendBluetooth(14);  # based on message, send
    # Bluetooth message
    pyb.delay(3000)  # delay to give human or tablet
    # chance to play audio for feedback
    self.sentdata.append('too soft');
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance==97 or self.distance==98 or self.distance==
99 or self.distance==100 or self.distance==101 ;
    self.sendBluetooth(16);
    pyb.delay(3000)
    self.sentdata.append('soft');
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance == 102:
    self.sendBluetooth(11);
    pyb.delay(3000)
    self.sentdata.append('GOAL');
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance==103 or self.distance==104 or self.distance==
105 or self.distance==106 or self.distance==107 :
    self.sendBluetooth(17);
    pyb.delay(3000)
    self.sentdata.append('hard');
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance==110:
    self.sendBluetooth(15);
    pyb.delay(3000)
    self.sentdata.append('hard');
    self.transitionTo(self.S2_RECEIVE_DATA)

elif self.distance==108 :
    self.sendBluetooth(18);
    pyb.delay(3000)
    self.sentdata.append('No puck detected');
    self.transitionTo(self.S2_RECEIVE_DATA)

else:
    self.transitionTo(self.S6_SEND_KP)
else:
    self.sentdata.append('NA');  # if no feedback for this trial, append
NA to list
    self.transitionTo(self.S2_RECEIVE_DATA)

elif(self.state == self.S7_BREAK):
    if button==1:  # only occurs when button pressed again,
# indicating break is over
        button = 0
        val = "f"
myuart.write(val)  # send message to frontend that break is over
self.sendBluetooth(21)  # message sent via Bluetooth to
# play break is over dialogue
self.transitionTo(self.S2_RECEIVE_DATA)

else:
    pass

elif(self.state == self.S8_RETENTION_START):
    if myuart.any():
        letter = ord(myuart.read().decode('ascii'))
        # print('intro')
        if letter == 114:  # if message received was a r
            self.sendBluetooth(25)  # send Bluetooth message to play
            # Retention trials starting soon
            self.transitionTo(self.S8_RETENTION_START)
        if letter == 113:  # if message received was a q
            # Retention trials starting now
            self.transitionTo(self.S9_RETENTION_TESTB)

    else:
        self.transitionTo(self.S8_RETENTION_START)

elif(self.state == self.S9_RETENTION_TESTB):
    if self.counter < self.totaltrialnumberr:  # if Retention trial limit not exceeded
        if myuart.any():
            letter = ord(myuart.read().decode('ascii'))
            if letter == 116:  # if the character value is for the letter t
                self.sendBluetooth(22)  # TRIAL START SOUND
                self.transitionTo(self.S10_RETENTION_TESTF)
            self.sendBluetooth(23)  # send Bluetooth message signaling tablet to play trial is over sound
            self.transitionTo(self.S9_RETENTION_TESTB)

    else:
        self.transitionTo(self.S9_RETENTION_TESTB)

elif(self.state == self.S10_RETENTION_TESTF):
    if myuart.any():  # if message received
        letter = ord(myuart.read().decode('ascii'))
        if letter == 118:  # if character value for v
            self.counter += 1
            self.sentdata.append('NA')  # append NA since no feedback
            self.sendBluetooth(23)  # Bluetooth message signaling tablet to play trial is over sound
            self.transitionTo(self.S9_RETENTION_TESTB)

    else:
        self.transitionTo(self.S10_RETENTION_TESTF)

elif(self.state == self.S11_END):
    pyb.delay(2000)
    self.sendBluetooth(19)
    myuart.write('
'.format(str(self.sentdata)).encode('ascii'))  # send list of feedback to frontend
    self.counter = 0;
    pyb.delay(10000);
    sys.exit('All trials have been completed.')
else:
    # Invalid state code (error handling)
    pass

    self.runs += 1

    # Specifying the next time the task will run
    self.next_time = utime.ticks_add(self.next_time, self.interval)

def transitionTo(self, newState):
    # defines transitionTo state
    self.state = newState

# Function to write inputs
def sendBluetooth(self, number):
    # Function to send character via Bluetooth.
    uart3.write(chr(number));
A.7 Backend Main
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

"""
Created on Thu May 12 13:57:56 2022
@author: ashleyhumpal
"""

from pyb import UART
from SAA import SAA

## 1 for KR FEEDBACK, 2 FOR KP FEEDBACK

task2 = SAA(1)

while True:
    task2.run() #runs task
A.8 Procedure
Procedure

**Before Participant Arrives**
1. Turn on laptop and login (ensure battery is high or that it's on charger)
2. Ensure that Excel is closed
3. Connect Nucleo to power
4. Plug in USBs to Nucleo and Connect to laptop
5. Plug in Camera USB cable to Laptop
6. Plug in lamp and turn it on
7. Hit Black Button on Nucleo to Reset it
8. Open Spyder and Putty application
9. Open Spyder to "SAAFrontend1.py" file
10. Power on Tablet and place on stand with charger connected
11. Randomly determine Feedback type and Experimenter type
12. Go into "SAAFrontend1.py" file, scroll to bottom where “if_main_” statement is located.
   Located at bottom is “ SAAFrontend ( , )” Space before comma is for experimenter, if tablet type 3, if human type 4. Space after is for feedback, if KR type 1, if KP type 2.
   Example if conducting tablet KR trial: “SAAFrontend(3,1)” goes there.
13. Locate “SAAmain.py” file. Where it has “task2 = SAA( )” enter number 1 if conducting KR feedback and 2 if conducting KP feedback. Ex, if doing KR then it would look as follows: “task2 = SAA(1) “ Make sure to save after.
14. Open Anaconda Powershell and type following and hit enter at each step
   a. cd desktop
   b. cd SAATesting
   c. ampy put SAAmain.py SAAmain.py
15. Go to putty and load “Working” settings into Putty
16. Confirm that MindRove armband is working by:
   a. Turn on armband (green LED should light)
   b. Ensuring MindRove antenna is connected to USB.
c. Click on wireless symbol 📱 to confirm connection to MindRove network
d. If password required, enter: #mindrove
e. Open MindRove software from desktop shortcut – if waves (green/blue trajectories) are not displaying, then close and open again

*If doing back to back trials, repeat steps 2,6,10-16

**When Participant Arrives**

1. Have Participant Sign Consent and fill out Demographic Forms
2. Bring Participant around table and show them zones on table
3. Verbally Inform them of who experimenter will be and if they need a break to inform you trial prior
4. Inform them to keep their elbow on and to not kick or move table, and to keep hands behind line after moving puck
5. Allow participant to do 5 practice trials where you give them verbal feedback after each one
6. Ask participant if they have question otherwise you will begin
7. Put MindRove armband on participant
8. In MindRove software application, click on ‘Record’ button. This will allow you to name the .csv file where data will be saved. Select a name pxx.csv (participant ID number)

**To start Trials**

1. If experimenter is tablet, load Motor Training Test App on tablet and allow it to face user
2. Run “SAAFrontend1.py” file
3. Once console print “blanking” go to putty console and type “execfile(SAAmain.py)’”
4. The Intro message should begin – if human is experimenter read introduction from Experimenter Dialogue document to user when python console prints “Intro Message”

5. If human experimenter, read everything in console that is in quotes (“trial start, trial end, your puck is in zone.. push harder ” etc…)

6. Whenever “Trial is over” is printed in console or said by tablet, use hand or rake to quickly grab puck (only have 10 seconds max). Make sure to keep all body parts behind partition afterwards.

7. If user requests break, hit the black button and the break will start after immediate trial is over. Hit the button again to resume trials.

8. Once console prints “trials completed” inform user of retention test and that they will take it without feedback in 5 minutes

9. Once 5 minutes have passed, Spyder will prompt you to read Retention Starting dialogue, which can be read from Experimenter Dialogue document

10. Conduct retention test and keep picking up pucks

11. Once conclusion message is read, trials are officially over.

**After Trials**

1. Tell participant their ID number

2. Remove armband, and press ‘Save’ on the MindRove software application

3. Tell them to fill out Google Form and submit it

4. Thank them for their time and tell them they can leave

5. Go into “SAATesting folder” and copy “SAAdata.csv” into “Participant Data” folder and save as “SAAdata_participant_xx.csv”

6. Open “SAAdata_participant_xx.csv” and manually type in participant ID number on upper right corner of file

7. Manually clear original SAAdata.csv” file for next user

8. Go into “SAAFrontend1.py” and delete number parameters entered previously for the settings of the trials so its just spaces
9. Go into "SAAmain.py" and manually delete number parameters too, and save and use Anaconda PowerShell to put it to the nucleo again
A.9 Survey Questions
Post-Experiment Survey

* Required

1. Enter Participant ID Number *

__________________________

System Usability Scale

2. I think that I would like to use this system frequently *

Mark only one oval.

__________________________

Strongly disagree

1

2

3

4

5

__________________________

Strongly agree

__________________________
3. I found the system unnecessarily complex *

Mark only one oval.

Stronly disagree

1
2
3
4
5

Strongly agree
4. I thought the system was easy to use *

*Mark only one oval.*

<table>
<thead>
<tr>
<th>Strongly disagree</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strongly agree</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5. I think that I would need the support of a technical person to be able to use this system

* Mark only one oval.

Strongly disagree

1

2

3

4

5

Strongly agree
6. I found the various functions in this system were well integrated *

Mark only one oval.

Strongly disagree

Strongly agree
7. I thought there was too much inconsistency in this system *

*Mark only one oval.*

<table>
<thead>
<tr>
<th>Strongly disagree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Strongly agree</th>
</tr>
</thead>
</table>

118
8. I would imagine that most people would learn to use this system very quickly *

Mark only one oval.

Strongly disagree

1  

2  

3  

4  

5  

Strongly agree
9. I found the system very cumbersome to use *

Mark only one oval.

Strongly disagree

1

2

3

4

5

Strongly agree
10. I felt very confident using the system *

*Mark only one oval.*

Strongly disagree

1

2

3

4

5

Strongly agree
11. I needed to learn a lot of things before I could get going with this system *

Mark only one oval.

Strongly disagree

1

2

3

4

5

Strongly agree

Supplemental Questions

12. How frequently did you look at the experimenter?

Mark only one oval.

☐ At least once per trial
☐ Once every few trials
☐ Only occasionally
☐ Not at all
13. Feedback was not provided on every trial. Did you feel the feedback was:

*Mark only one oval.*

☐ Too frequent  
☐ Just right  
☐ Not frequent enough

14. How useful was the verbal feedback provided by the experimenter?

*Mark only one oval.*

1

2

3

4

5

6

7

Very useful
15. How challenging was the task?

*Mark only one oval.*

Not challenging at all

1

2

3

4

5

6

7

Very challenging

16. Did your performance improve during the experiment?

*Mark only one oval.*

☐ Yes

☐ No

☐ I am not sure.
17. If you answered 'Yes' to the prior question, what helped you to improve your performance?
A 2-way Analysis of Variance (ANOVA) non-parametric may be used for the analysis of the study. This method of analysis was chosen because the ANOVA method is ideal for studies in which there are three or more groups being compared and when there are two primary independent variables. The current study is cross-sectional, between-groups study that uses a 2x2 design. The two independent variables are feedback delivery media and feedback type, with two levels each. For feedback type the two levels are KP and KR. For the delivery medium the two levels are the tablet and human. The four groups being compared are: KP human, KR human, KP tablet, KR tablet. The two independent variables’ influence together can be examined in this two-way method as well as each variable’s effect independent of each other [23]. The ANOVA method can be used to investigate the change score, which is the difference between the participants average performance score before and after the Retention trials. Specifically, the performance score will be the error for each trial, so the smaller the score the better the trial. The average error may be calculated for the initial trials and then similarly the average error can be calculated for the Retention trials. Through analysis, attached in Appendix A.11, the optimal sample sizes can be determined for each group. Furthermore, the IMU data can be used with a large enough sample size to differentiate the motion between a good trial (one that scores high) and a bad trial (one that scores low). It can also determine what the variability in between participants or individuals may be during the experiment and discover if there is a pattern.

The result I would expect to see when experimenting with the ideal sample size is the KR feedback scoring higher than the KP feedback, as the latter may be too relative or vague for some users. Amongst KR feedback, the human coach may only
very slightly perform better than the SAA coach because participants may find the SAA less personable which could affect their performance. My hypothesis is that the difference will be essentially negligible.

### A.11 G*Power Sample Size Analysis

The ideal sample size can be calculated using the software G*Power. The settings necessary are the test family, which is a “F test”. The statistical analysis required is a 2-way analysis of variance (ANOVA) test. In G*Power this can be input by changing the Statistical test setting to “ANOVA: Fixed effects, special main effects and interactions.” Alpha, the significance of the test, will be set to the standard of 0.05, while the power is set to the standard value of 0.8. For a complete look of the settings input to G*Power please reference Figure A.1. In order to calculate the sample size, the effect size (the estimate of how much better each group should perform) is also needed. The greater the effect size, the greater the differential effects in the study. In the case of when there is not enough data to properly identify the effect size, an alternative is collecting it from existing studies that have similar variables that are being tested. For this experiment, a study by Zhexenova et al. was examined for its effect size. The goal of the study was to determine what type of learning aid was most effective in helping children learn a new language [24]. The different learning aids were a human teacher, a robot with writing tablet, and a tablet only. The effect size comparing the effectiveness between the tablet and human teacher was specifically calculated in the study and found to be 0.713. Using this effect size, the ideal sample size can be calculated with the G*Power software for this experiment and is found to be 18. In future experiments, the ideal sample size of 18 per group should be used.
Figure A.1: G*Power Settings