Comprehensive Interactive Neurorehabilitation System Design and Implementation Through the Application of Interdisciplinary Research and Integrated Design Approaches

by

Michael Baran

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved October 2014 by the
Graduate Supervisory Committee:

Loren Olson, Co-Chair
Thanassis Rikakis, Co-Chair
Steven Wolf
Todd Ingalls

ARIZONA STATE UNIVERSITY

December 2014

ABSTRACT

Stroke is a leading cause of disability with varying effects across stroke survivors necessitating comprehensive approaches to rehabilitation. Interactive neurorehabilitation (INR) systems represent promising technological solutions that can provide an array of sensing, feedback and analysis tools which hold the potential to maximize clinical therapy as well as extend therapy to the home. Currently, there are a variety of approaches to INR design, which coupled with minimal large-scale clinical data, has led to a lack of cohesion in INR design. INR design presents an inherently complex space as these systems have multiple users including stroke survivors, therapists and designers, each with their own user experience needs. This dissertation proposes that comprehensive INR design, which can address this complex user space, requires and benefits from the application of interdisciplinary research that spans motor learning and interactive learning. A methodology for integrated and iterative design approaches to INR task experience, assessment, hardware, software and interactive training protocol design is proposed within the comprehensive example of design and implementation of a mixed reality rehabilitation system for minimally supervised environments. This system was tested with eight stroke survivors who showed promising results in both functional and movement quality improvement. The results of testing the system with stroke survivors as well as observing user experiences will be presented along with suggested improvements to the proposed design methodology. This integrative design methodology is proposed to have benefit for

not only comprehensive INR design but also complex interactive system design in general.

DEDICATION

To Mom and Dad

Thank you!

ACKNOWLEDGMENTS

Thanassis: Thank you for not laughing too loud when a nervous bioengineering student walked in to your office wondering if he could shift graduate career paths. The growth that you fostered in me from your advising has been inspirational and a great honor. I never would have guessed where this journey would have taken me and I am so grateful for all of the opportunities you have presented me.

Nicole: I am not sure what I can write here that fully encapsulates my gratitude to having the opportunity to work and collaborate with you. Let me start with: Thank you! The work that is presented here does not happen without your guidance, collaboration and friendship. (And I hope this dissertation stands as a testament to that.) Thank you for fighting in the “INR trenches” with me. I wish you all the best in your upcoming journeys and cannot wait to see the successful ventures you will embark upon.

Loren Olson, Todd Ingalls and the entire faculty at AME: Thank you for all of your guidance and instruction, from which I have learned so much.

Steve Wolf: Thank you for your all of your time, guidance and advisement on my work.

Assegid Kidane, Kelly Phillips, and Luke Kautz: Thank you for giving two graduate students their wings through your tireless dedication and patience.

Aimee Reiss, Meghan Buell, Zev Rymer, and all of the staff at Emory and RIC: Thank you for taking a chance on our work. I greatly appreciate your willingness and patience to interact with an experimental system.

All of my current and former colleagues at AME: Thank you for all of your assistance. Your contributions, in all forms, have taught me much and improved my work.

Yinpeng Chen: You motivated me to work with programming and think about system design, which excited me to pursue both. Thank you for all of your guidance and mentorship.

My parents: Thank you doesn't seem like enough here. You have always been there for me. You've celebrated the highs and supported me through the lows. I love you both and would not be where I am today without your love and support. What do you think, should I get another degree? Just kidding…

Melody: Your care, affection, and support have kept me smiling through the whole dissertation process and beyond. You inspire me everyday to be excited about the next adventures in life. I hope to continually share that sense of excitement and laughter with you. I love you and I am so grateful to have you in my life. Go Team M&M!

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

APPENDIX

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

CHAPTER 1

INTRODUCTION AND PROBLEM STATEMENT

Stroke is a leading cause of disability in the United States. On average, every 40 seconds, someone suffers a stroke in the US, leaving millions of people with chronic upper-extremity impairments.[1] The extent and severity of these impairments varies across stroke survivors, necessitating comprehensive rehabilitation systems that can provide meaningful experiences based on individual needs. However, repeated visits to receive clinical-based therapy can be costly to a stroke survivor, both financially and logistically.[2] Technology has the ability to provide an array of sensing, computation and data solutions that hold the potential to maximize the time of physical therapy as well as extend therapy to home based environments. Therefore, the application of technology in physical therapy has become a burgeoning field.

Interactive neurorehabilitation (INR) systems represent a core group of technologies that have been applied to rehabilitation physical therapy. [3-5] While some implementation details may vary per particular system, at their core they feature similar capabilities, as seen in Figure 1. The user of the system, the stroke survivor, performs a physical activity. INR systems use various hardware and software solutions to track and measure this physical activity, which serves as the primary input. From this input, the INR system can extract various assessments and evaluations of the activity and provide feedback back to the user based on these evaluations. The form of this feedback can range across audio, visual and tactile.

The user then utilizes this information to correct their movement either in real time or upon reflection of feedback provided. The therapist is also a user of INR systems. Depending on the application and environment of the system, the therapist may also be observing the user's interaction of the system and can use the resulting evaluations and feedback to provide their own verbal or physical feedback to the patient. In addition, the therapist can make adaptations to the system or protocol to adapt the challenge or sequence of the activities.

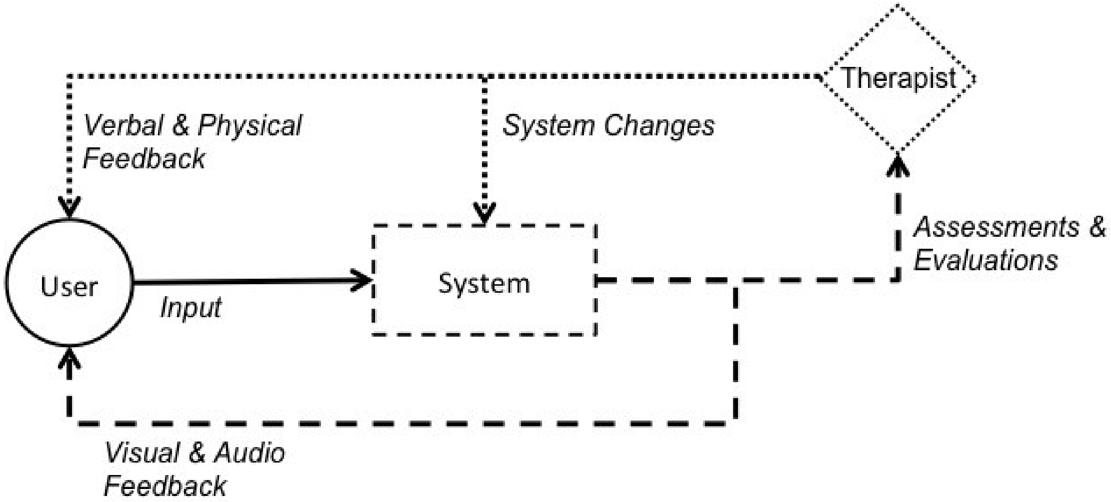


Figure 1 - The core model of INR systems can be represented as a basic feedback loop between the user (a stroke survivor), the system and a physical therapist

INR systems for stroke rehabilitation are beginning to show promise leading to practice-dependent improvement in motor function of the affected arm[6] and also have demonstrated greater improvements in limb function in comparison to conventional therapy alone[7], though the extent to which INR is more effective than traditional therapy is still under investigation.

Currently, the approaches to INR design range widely from robotic devices (where an apparatus actively drives or momentarily assists a movement)[8-10], virtual reality environments (where a task is completely recontextualized away from a physical environment)[11], or mixed reality (where digital media and physical elements are hybridized such that an activity can be adaptively recontextualized with varying amounts of media).[12]

Robotics can have a range of capabilities from fully driving a movement, to only actively assisting at determined moments.[8] Robotic systems that can be classified as INR systems will couple this with the display of feedback.[8-10,13] The main limitation to robotic systems is that they limit active learning. They do not train users to understand their movement in terms of an aggregation of motor elements necessary for self-assessment. Other limitations to robotic INR include degrees of freedom and range of motion, and variety of visual feedback games. Many existing games only provide a visual representation of the end point in relation to the context of a game, rather than within the context of a functional task.

Another approach to interactive rehabilitation is through the use of virtual reality.[7,11,14] For INR purposes this means that the tracked actions of the patient do not engage the physical world. They only engage in virtual tasks and the results are fully represented in virtual reality. Virtual environments provide the benefit of utilizing augmented feedback to provide detailed information about performance that may not otherwise be available in a real world context, such as direct information about spatial and speed aspects of movement. Virtual reality also provides flexibility in designing tasks, such that they can support generalization to activities of daily living, as well as adapting the challenge level to a patient's needs. However, VR INR systems have a key limitation. Because tasks exist in a near exclusively virtual environment this impedes the transference of gains made in VR to ADLs in the physical world[12,15,16].

Mixed reality integrates virtual environments[11] with physical objects to manipulate or navigate, and has the potential to facilitate training that can transfer to other contexts[12], such as activities of daily living (ADLs). Increasing the amount of digital feedback dissociates from the physical task by changing the context in which the task is performed while decreasing or eliminating the presence of digital feedback requires that the patient move more independently. Dynamically adjusting the amount of digital feedback helps the patient connect learning in the virtual domain to physical action. Methodologies within mixed reality INR can be effective in both continuously supervised clinical and minimally supervised environments.

As a result of these various approaches, a coherent methodology for how INR systems are developed, implemented, and assessed along functional outcomes and patient progress, is lacking. The main problem is that currently no large-scale clinical data validating the optimal approach to INR is available, due to challenges of recruiting participant populations for experimental, unproven systems in significant amounts in a timely fashion. However, before even beginning to build this body of data, consistent implementation of well-developed INR across a number of studies is required.

While stroke rehabilitation science can provide some answers to appropriately defining "well-developed INR", fields beyond rehabilitation have produced significant data for best practices in areas such as mediated interactive learning that can, and should, be incorporated. While these fields may not have arisen from stroke neurorehabilitation studies, they can provide well-tested ideas and concepts that share commonalities with neurorehabilitation. For example, the arts have for centuries studied and constructed complex displays for context-aware self-reflection.[17] Learning through creative practice has formed the basis of constructivist learning methodologies[18,19] that are prevalent in 21st century mediated learning. Rapidly evolving applications of interactive media (from mobile apps to interactive data visualizations) also rely heavily on the integration of arts, computing and mediated learning knowledge.[20] However there are many challenges that need to be addressed when integrating this information and applying the integrated approaches to neurorehabilitation.

INR systems are inherently very complex because of the merging of seemingly disparate aspects: INR systems have to merge physical with digital and merge people with computations. (Figure 2) While hybridizing these aspects is a challenge in and of itself, the dynamics of stroke rehabilitation offer further challenges:

- Stroke can affect patients differently, and thus the prospective population of users for and INR system will range in ability. In addition, for a given patient, ability will change over time while engaged in a therapy program. Thus challenges and training goals that were applicable early on in training may no longer be relevant in later training.
- Just as patients are dynamic, the technology available for an INR system is also dynamic. At the moment of designing and implementing an INR system, a particular solution may arise in the future that offers better capabilities than a particular aspect currently integrated in the system. Furthermore, a current design and implementation impossibility may become possible with future technological advancements.
- The implementation of an INR system needs to be cognizant of the space in which it will be utilized. For example, clinical spaces typically offer the ability for long term, more permanent installation of components. In addition, therapist supervision may be more frequent. However, in a home environment, the same assumptions may not be applicable. Therefore, the space of an INR system will greatly inform its design.
- While INR systems are primarily designed for the improvement of patient ability and rehabilitation, physical therapists need to also be considered as primary uses of the system, as previously discussed. INR systems are designed to assist therapists to meet their goals for a particular patient. Therefore, INR systems need to empower the therapist to make the best

assessments and resulting therapy decisions as possible with the given resources they have available.

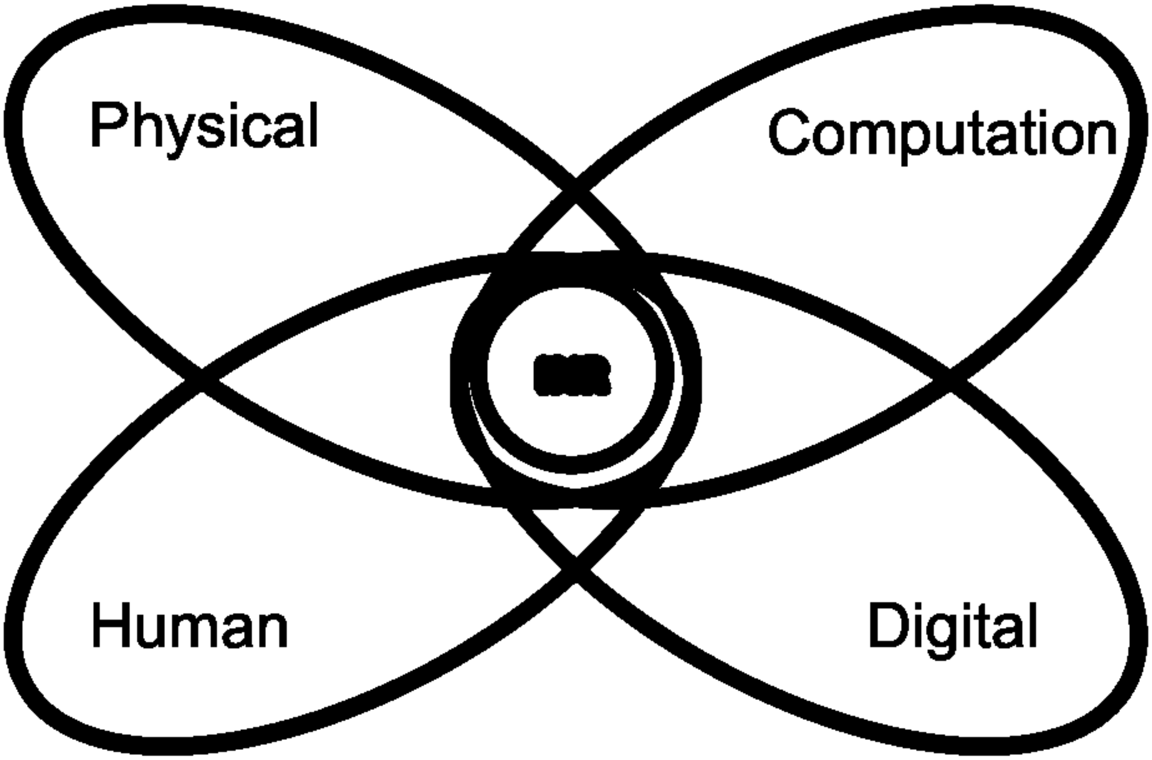


Figure 2 - The complexity of an INR design and implementation exist within a space that needs to connect variable elements of human and digital as well as physical and computation.

As can be seen a design of a one-size-fits-all solution, both for different patients and environments, is unlikely. Therefore, the possibility opens up that multiple INR systems will be required, each maximizing a solution to a particular combination of the complex space previously discussed. This is one of the primary reasons for the variety in INR approaches previously introduced. However, if multiple systems are to be designed and implemented, an emphasis on continuity design across systems is required:

- The training needs to be continuous across systems. Training from one system should transfer and to training in another system. In addition, this training needs to transfer to every day activities.
- To support the continuous connection, there needs to be a consistency in the data that each INR system can provide. Evaluating patient progress is one of the key metrics for determining if an INR system has any benefits. This requires that all systems can provide a continuity of data, such that patient progress can be assessed accurately in multiple contexts to provide a comprehensive view of progress.
- There also needs to be a continuity of user experience. Commercial design provides examples of this idea, such as the community of Apple products. These products are designed in such a way that they all share similar features and characteristics. Thus, while a user may not know all the intricate details of how to interface with a particular product, their experience with other Apple products can carry over and give them a starting point. The same needs to be true for INR systems. A patient should not have to relearn functionally different feedback environments or user interfaces. The experience of using and interfacing with one system should smoothly transition to another system.

This dissertation proposes that comprehensive INR design, an inherently complex problem, requires the hybridization of multiple knowledge sources, each with their own inherent levels of uncertainty. A definition of comprehensive INR design should be created before reaching for large scale clinical studies as this will help increase the likelihood of returning significant data and avoid trying to answer questions within INR that have already been answered in other fields. The intent of this dissertation is to define guidelines of complex system design, within the specific context of INR, to integrate interdisciplinary knowledge through applied design approaches, which can generalize to other integrated system design.

A proposed model for overall INR complex system design can be seen in Figure 3. The first stage is to assess contextual research. In the case of INR design, this requires reviewing core field research in neurorehabilitation, but also considering relevant interdisciplinary fields as well for best practices that may not exist comprehensively yet within neurorehabilitation. This interdisciplinary contextual research will directly inform the design of INR systems by providing constraints for what core functions are required as well as how proper assessments can be made to evaluate patient progress, and as a result, evaluate the validity of the particular INR implementation methodology. Once these constraints are identified, their implementation requires the used of a hybrid of existing design approaches. Multiple cycles of iterative design and testing, at various time scales, are required to test system components from early prototypes to a more fully functioning system. Once a full system is tested, the results form new questions for further contextual research or provide new insights for these domains as well.

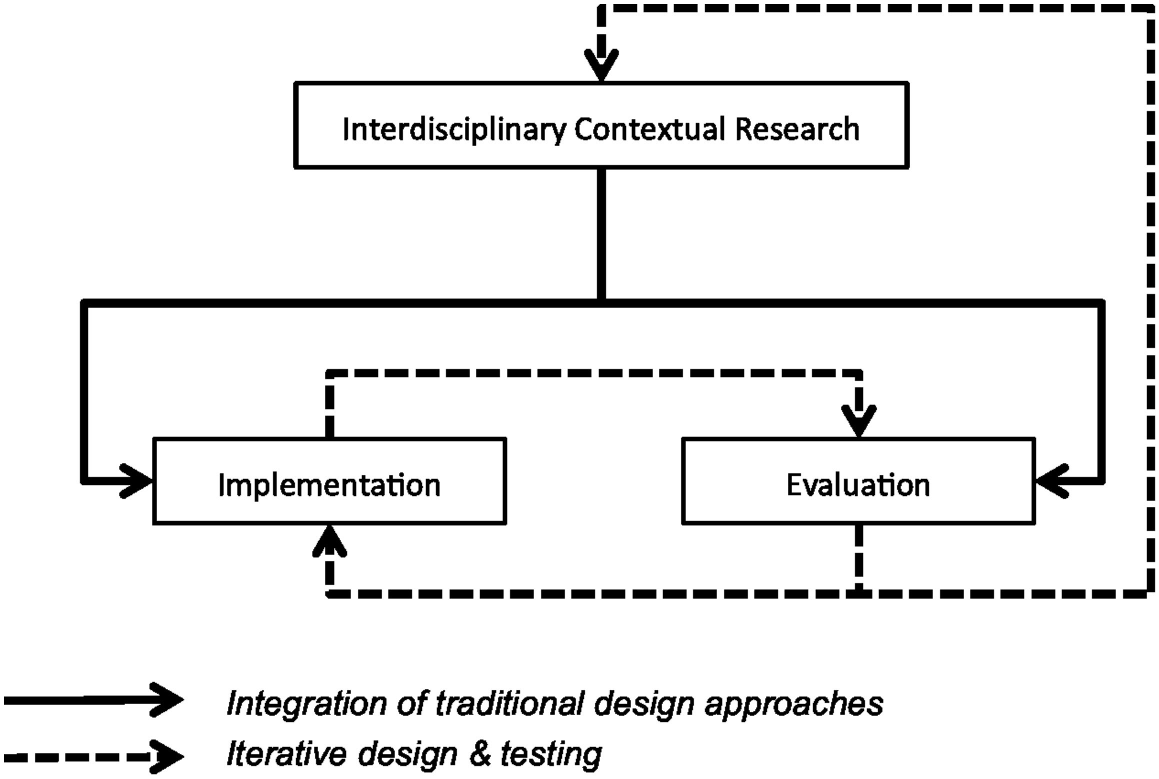


Figure 3 - Effective INR design should leverage interdisciplinary research to create system design comprehensive requirements and metrics that should be tested in multiple cycles of iterative design.

The proposed model is very much based in existing thought on experiential systems design[21] as well as the ideas contained within the Universal Principles for Design[22] and therefore is not unique in and of itself. However in this dissertation, I propose and demonstrate the unique application of these concepts and models within an INR context, thereby highlighting their utility as well as promoting their importance for consideration within the field of neurorehabilitation. My process of applying this model has led to numerous conclusions on the utility of the model within INR as well as more generalized thoughts on its overall utility across applications.

The intent of this dissertation is to show how this system design model was applied within an INR system design and implementation cycle, from contextual research to implementation, with reflections on considerations for more optimal implementation. Chapter 2 reviews utilizing contextual research to establish therapy task and patient evaluation constraints. Chapter 3 reviews approaches to design and how they are applicable to complex INR system design. Chapter 4 reviews the implementation of the task and evaluation design constraints within a fully developed and tested INR system. Chapter 5 reviews evaluations of this system across patient specific criteria as well overall system stability and use observations. Chapter 6 reviews future directions for extending the example INR system as well as overall reflections on the practicalities of the application of the system design model with a collaborative design team. Chapter 7 presents a summarized model for INR development that can extended towards other interactive system design contexts. Please note that some small portions of content presented in this dissertation were simultaneously authored by me and accepted for publication elsewhere[23] and have been reprinted here with permission (Appendix D).

CHAPTER 2

ASSESSING THE COMPLEX SPACE OF INR RESEARCH AND DESIGN

INR exists within a very complex space complete with multiple dimensions that need to be addressed in varying ways depending on the specific application of the system. This chapter outlines some of the key considerations that are necessary in the design of mixed reality INR systems. Section 2.1 reviews key contextual research from motor learning as well as other interdisciplinary fields that are crucial to comprehensive task experience and patient assessment design. Section 2.2 identifies key design constraints that arise from the contextual research. Section 2.3 concludes with a clarification of where in the complex space of INR my design work fits.

2.1 Key Contextual Research

2.1.1 Motor Learning

There has been significant research in areas of motor learning and developmental learning. Overall, learning occurs within highly individualized contexts (e.g. each individual constructs his own knowledge from first person experience), and the path to achieving knowledge can be non-linear and vary across learners. The accumulation of experiences contributes to a complex internal model that continuously evolves with more experience.[24] Motor learning and other statistical learning[25] research suggests that internal models are formed based on a collection of perceived rules, associations, and probabilities that are distilled from a variety of experiences in which people physically engage with their environment.[20]

Efforts to understand underlying mechanisms that foster stroke recovery suggest that active engagement, challenge, and independent discovery can facilitate neuroplasticity.[26] In stroke rehabilitation, problem solving is critical to enhance motor learning.[27] Looking beyond the field of rehabilitation science, the arts provide an exemplary medium for implicitly shaping the individual's experience without explicit instruction[28] so that the individual can independently draw conclusions about his experience. For example, the painting tradition capitalizes on our inherent perceptual sensitivities to patterns of value and color to achieve an intuitive and meaningful visual experience.[29] The arts also provide guidance on achieving long-term engagement over a sequence of separate sessions: consider how each chapter in a novel leads to anticipation of culmination.[30]

2.1.2 Constructivist Learning and Reductionist Hierarchies

Educational models grounded in constructionist theory emphasize learning through knowledge structures[31], or component-level understanding of the phenomenon being studied. Modular or structural learning, gained through active engagement within a diverse set of problem-solving contexts, allows for more meaningful and sustained learning that generalizes to real world applications.[32] Similarly, within a motor learning context, the motor control system relies on structural, generalizable learning for skill acquisition.[33,34] For example, some kinematic features are invariant across different types of multi-joint movement[35,36]and should be focused upon within training, as they are the most easily generalizable across different types of tasks.[37] A critical aspect of this formed generalizable knowledge is that not only can knowledge structures be reduced back down to component levels, but also higher concepts of quality can be established, which is a key feature of reductionist hierarchy models. This can be further demonstrated in the following example.

2.1.2.1 Example Learning Application: Musical Instruments

An area of learning that incorporates motor learning and constructivist learning ideas is musical instrument performance instruction. The Dalcroze[38], Kodaly[39], Orff[40] and Suzuki[41] methods are historically accepted approaches to musical instrument instruction. All of these methodologies approach music instruction from the viewpoint that anyone can gain an appreciation for music, and form amateur performance skills at a minimum. The key for these methods is to introduce musical concepts within a framework of understanding that the student already possesses and experience the music directly. Each method also structures the introduction of topics in a clear hierarchy that always tries to make small steps up in difficulty and complexity. Kodaly tried to form his instruction to mirror how a child's development occurs. For example, he found that children had a natural ability to discern specific note ranges and intervals, and thus began the instruction of relative pitch from that foundation.[39] The Kodaly method follows a structure of starting with what the student knows and connecting musical experience to that context. Then, in small steps, new concepts are introduced in which the student needs to use their current musical framework to see how the new information fits and, as a result, what needs to be updated within their conceptual framework. Within the Suzuki method, the process of learning typically utilizes a set ordered repertoire that teach certain skills in a meaningful sequence for student development. The Suzuki method has a focus on the "development of ability" such that as soon as the student can play a song, they do so. Then, the immediate next step is to continue to work on making the song technically and musically perfect.

These approaches to musical instrument instruction establish reductionist hierarchies. Underlying fundamental components to a song are practiced in isolation as well as integrated together within the context of the song. While this demonstrates to a student how such concepts as scales and chord techniques (both understanding their place in music theory as well as physical performance) can aggregate to performing a song, it also builds an understanding for the ability to break down larger forms to their constituent components. Thus, if a reductionist hierarchy of musical performance is formed, when a performer notes an error in the performance of a song, they are able to break the error down to its core components in order to isolate the specific detailed aspect of the performance to correct.

The goals of these models of instruction are to get a student performing songs, increase their music appreciation and, resultantly, improve their daily life experience. This is where the true value and motivation of musical performance lies. The reward of learning all of the techniques is to perform full songs and create expressive experiences.

Connecting back to neurorehabilitation therapy, patients want the ability to perform complex tasks that are part of every day life. However, they need to build motor learning skills that allow them to self-analyze and break down error in movements to its constituent components. Therefore, patients need to be continually practicing individual components of a movement as well as how they aggregate in complex tasks. They should also be afforded the ability, through the tasks, to isolate individual components within the performance of a complex task. INR system should reward the user through continuous connection to complex task accomplishment.

2.2 Resulting Design Constraints

As the contextual research suggests, INR systems need to focus on active motor learning that can provide the appropriate level of challenge and show how individual motor elements integrate within a complex task. From this, we can begin to compose a series of constraints for the design of the tasks of the system that the patient will perform.

2.2.1 Task Experience Design Constraints

2.2.1.1 Tasks Should Generalize to Activities of Daily Living (ADLs)

The training tasks of an INR system should focus on components of ADLs and train generalizable functional movements. As people learn and aggregate learning over multiple experiences, motor skills gained in use of INR should be easily applicable and reinforced in a patient's daily life. Similar to the example of musical instrument instruction, the true value in physical therapy is the performance of complex tasks, not just isolated practice of motor elements. INR systems should be able to take a complex task encountered in daily life, and help a patient break down the task into components and demonstrate how the components aggregate.

2.2.1.2 Integrate Component and Complex Task Training

INR therapy protocols should aim to integrate training of movement components within a goal-directed, functional context.[42] The action goal should be the primary focus[43], while the secondary focus should be on the contribution of movement components to the action goal and their interrelationships. When possible, tasks should be unassisted to reinforce patient problem solving, within an appropriate challenge level at a given moment in time.[44] Training sequences should move continually up and down the task hierarchy (from components to complex task) and the corresponding feedback hierarchy. This process promotes synthesis of components and decomposition of complex tasks into components.

2.2.1.3 Tasks Should Scale in Difficulty

Each patient will have a different starting point in ability, but the end goal of training a complex, functional task encountered in daily life remains the same. Therefore, the range of tasks needs to cover a distribution of abilities, while also scaffolding together to build towards complex, functional tasks and competency in ADLs.

2.2.1.4 Adapt Task Complexity and Challenge Level Across Pertinent Dimensions

The dimensions of task complexity and challenge level need to be adaptable, as each patient will learn differently and have different abilities. INR design should identify key dimensions of adaptability that ensure a distribution of patients can receive similar benefits from use of the system.

2.2.1.5 Balance Repetition and Variation

Tasks need to support repetitive training, since the amount of motor improvement correlates with the amount of practice.[37] However, while blocked repetitive tasks may show short-term benefits, long-term generalization and retention is better supported through introducing variability in a training protocol.[43] The exact balance between repetition and variation is not yet clear. Related research on how learning aggregates with repetition and variation is being explored in machine learning[45,46]and constructivist learning.[19,31]

2.2.1.6 Vary Level and Type of Supervision

While coaching is crucial to helping the patient during training, INR systems should be designed to maximize the limited time and resources of the physical therapist by gradually reducing the need for constant supervision during training. A primary role of the therapist within an INR system is to help prioritize focus on specific experiences before the patient has had the chance to develop internal models of how to best use the system to better perform the task. The amount and nature of the coaching can change over time and context. For example, inquiry based coaching has the therapist ask the patient questions to promote critical thinking (e.g. "What do you think causes the visual feedback to look like that?"), while prescriptive based coaching can be verbal (e.g. "Rotate your wrist less in the beginning of your reach") or physical (e.g. the therapist actively assists the patient to extend his elbow when reaching towards a target). Prescriptive coaching may be necessary initially in specific instances to avoid frustration. However type of supervision (e.g. decreasing prescriptive coaching and increasing inquiry based coaching) and amount of supervision (e.g. from the presence of the therapist at every session to visiting only every third session) need to fade as therapy progresses to encourage the patient to develop self-assessment strategies.

2.2.2 Assessment Design Constraints

One of the primary focuses in validating INR design is to evaluate patient progress. If a patient is not improving, an INR system has little to offer from a practical perspective. Therefore it becomes important to consider throughout the design of an INR system how patient progress can be accurately assessed within the complexity of the system. These evaluations need to reflect current understanding of motor learning and movement quality as well as current standard clinical measures that are used in daily physical therapy practice.

2.2.2.1 Assessments for INR Need to Integrate Functionality and Movement Quality

When assessing a patient's movement, both task completion and its quality require evaluation. Assessing the functionality of completing the task alone is not sufficient. Other aspects of the movement, such as amount of torso compensation, degree of elbow extension, degree of wrist rotation and other kinematic measurements, help distinguish between recovery of pre-stroke movement patterns and compensatory movements.[47]

2.2.2.2 Integrative Assessments Should be Achieved Through Combining Kinematics, Therapist Ratings, and Validation by Clinical Measures

Individual kinematics may not provide appropriate evaluations for variable, highly individualized multiphase movements.[48] Kinematics should be contextualized by collecting several references (e.g. determining the allowable degree of torso compensation), which is pragmatically problematic. Therapist ratings of movement quality can be variable, as each therapist will observe patients differently.[49] Clinical measures, while validated, provide only a higher level, category-based resolution of evaluation. However, an integrated evaluation of movement quality and functional improvement could be achieved through a synthesis of kinematics, ratings and clinical measures. While it may be desirable to move away from the more qualitative measures, they have a breadth of knowledge and research that should be leveraged for the initial validation of experimental assessments. Currently members of the INR team at ASU are looking at computational frameworks for movement quality assessment using a decision tree model that can provide measures that correlate with therapist ratings.[50]

2.2.2.3 Assessments Need to be Associated with ADLs

Assessing progress in a therapy protocol needs to correlate through evidence to improvements in ADLs. Traditional clinical approaches to assessing ADLs through questionnaires (e.g. Motor Activity Log (MAL)[51]) rely on self-reporting and can be subjective. Current studies are assessing ADL more objectively through promising ideas such as embedded sensing[52] and repurposing consumer technology.[53] Promising development work is underway to provide feedback on ADL evaluations.[54] Applications for portable devices, such as tablets, hold promise for facilitating more accurate personal documentation on daily activity and reflection.

2.3 Defining My Own Contributions Within a Specific Context of INR Space

As has been shown, INR design deals with a very complex, multi-dimensional space, where dimensions of multiple users, technology and learning must be addressed (each with their own specific considerations). When I joined the MMR team at ASU, they had just completed the design of a mixed reality rehabilitation system and were about to begin testing the system with stroke survivors. One of the future goals was to evolve the training of this system beyond the scope of a clinical space and move mixed reality rehabilitation to the home.

In the following chapters, a description of the design, implementation and evaluation of a minimally supervised INR system will be presented. Therefore, it is crucial at this point to identify where my contributions bit within the INR design process. I collaborated significantly with Nicole Lehrer in the entire design process. When thinking about how to move the clinical system forward, two significant components were identified: 1) the design of a system (both hardware and software) along with the user experience of that system (therapy protocols and system use by therapists and system designers) and 2) the design of a compositional approach to feedback.

My work has been within the first identified component. I took the previously identified task experience and assessment design constraints and translated them into implementation designs of both hardware and software for INR use under minimal supervisions. I primarily focused on the design and implementation of the software of the system, but also designed and maintained the architecture that connected other feedback and custom sensing hardware components designed solely by other members of the team. In addition, I also took the previous contextual research ideas along with the specific capabilities of the system to design protocols that created a multi-week therapy experience for stroke survivors using the system.

Nicole Lehrer's work was within the second identified component of compositional approaches to INR feedback design. As a result, the feedback of INR will not be discussed here, as it will be covered in much more depth in Nicole Lehrer's dissertation. However, the components of system design, user experience and feedback design are highly interconnected and all required for comprehensive INR design.

CHAPTER 3

INTEGRATED DESIGN CONSTRAINTS AND THEIR APPLICABILITY TO INR

The integration of interdisciplinary concepts to find solutions within a very complex problem space requires comprehensive design approaches. Complex system design is not a new idea (in fact research has been conducted to try and model, as well as automate, complex system development[55-57]) and there are many traditional approaches within schools of design. I argue that INR design and implementation requires an integration of multiple design approaches within an overall iterative design model. The application of these principles was already begun in the Adaptive Mixed Reality Rehabilitation System (AMRR), however new considerations were necessary for designing a home-based mixed reality INR system. Section 3.1 reviews current design approaches that are important to consider for INR. Section 3.2 reviews iterative design and its importance. Section 3.3 identifies high-level design constraints that come from the integration of design approaches. Section 3.4 shows how these design constraints (as well as the guidelines previously identified from contextual research) were used to review the Adaptive Mixed Reality Rehabilitation system, which formed the basis for designs of the Home-base Adaptive Mixed Reality Rehabilitation (HAMRR) system. Finally Section 3.5 briefly summarizes the transition from AMRR to a home-based system.

3.1 Design Approaches

3.1.1 Engineering Design

Engineering design has been defined as "the set of decision making processes and activities used to determine the form of an object given the functions desired by the customer."[58] In this approach to design, an emphasis is placed on the functionality of a device, usually following a mechanical and industrial engineering specific approach. Design engineering prioritizes customer needs, and designs products that can meet those demands, deciding upon levels of performance within various constraints. Engineering design approaches suggest ways to stage designs, beginning from purely conceptual, all the way to detailed specifications that can be manufactured. This school of design also heavily considers the economics of not only designing and building a product, but how well it can scale to a larger manufacturing process. Engineering design is also focused on planning, with a continual eye on timeline and the dynamics of a design team, and in that way, looks at the very pragmatic consideration of project management.

3.1.2 Industrial Design

Industrial design has been defined as "the professional service of creating and developing concepts and specifications that optimize the function, value, and appearance of products and systems for the mutual benefit of both user and manufacturer."[59] While there is certainly overlap between engineering and industrial design, here the focus is weighted more towards form, and its influence on function. The range of industrial design influence reaches towards such considerations as ease of use and maintenance, required interactions for product function, product safety, as well as considerations such as product identity, and intangibles such as perceived value or importance.

3.1.3 Experience Design

Experience design, which is connected to interface and interaction design, explores more about the use and resulting experience of a system. It is also a very interdisciplinary field as it seeks to find a framework for understanding experience across multiple disciplines.[60] At its core, experience design also looks for more philosophic understandings of how to define experience and the many facets it entails. This includes an argument that the primary way in which people synthesize activity with a particular environment or setting is through experience.[61] Similar to the previously mentioned idea that people learn in a multitude of contexts and environments, experience design takes a step back and looks at such factors as "memory, desire, anticipation, relations with others, cultural patterns, bodily feelings, sights, smells and sounds."[61] In this way, the experience design approach is very user empathetic, with the focus put squarely on thinking about what the user's experience might be like when using a particular technology. One model to study user experience categorizes experience into product-centered, user-centered, and interaction-centered.[60] It further explores and values how human emotion can shape experiences with technology.[60] Experience design requires a really deep understanding of the target users and an understanding of the impact the technology might have on the user. Further, experience design understands that experience is non-linear and therefore evolves over time. Experiences will scaffold future experiences and thus impact how experiences are interpreted.[60] The value of experience and its ability to create unique and identifiable experience language among technologies can be seen in many current consumer technology developers, such as Apple.[62]

3.1.3.1 Experiential Systems Design

Experiential systems design is a specific type of experience design that focuses on the creation of systems to foster an experience. As a direct result, the field is very interested in studying the nature of experience and how design systems shape a particular perception of experience. Overall, experiential systems design views experience as the result of multiple interactions that exist purely within subjective points of view. Therefore, approaches are very sensitive to how system design can create certain perceptions by the user. The approach proposed[63] is defined along the following key questions:

1. Point of View: How does the selection of data to sample affect construction of experiences?

2. Gaps: What data is being filtered out so as to facilitate users' constructive activity?
3. Flow of experience: How can we structure the flow of interaction with data to support construction models?
4. Form & Style: How can presentation and manipulation of data by the user establish experiences?
5. Context: How do we consider and individuals background?

Across these points, the user's experience (both with and without the system) is studied in its relation to what aspects of the system directly and indirectly impact the user's experience.

3.1.3.2 Reflective Design

Another subset of experience design is reflective design, which as the title suggests, supports the reflection of designers and users in regards to the interactions with technology.[64] It challenges designers to think critically of their work, especially in regards to aspects of the experience that a system may be indirectly providing to the user based on design choices. As a result, it encourages systems to provide opportunities to both designers and users to reflect on aspects of the system's experience. However, it stresses that reflection should not be separated from the experience, and as a result, considered to be cognitively separate from the experience.[64] Rather, reflection should be an integral part to the experience itself.

3.1.4 Software Design

Similar to the previously mentioned types of design, software development also has established approaches to design and implementation. Typically, design of code architectures is aided through the use of design patterns. Design patterns provide partial templates that have been proven in multiple contexts that can be combined together to optimize solutions to a problem while leveraging previously established details of a programming language.[65] Software design, through design patterns, is similar to the previously mentioned design approaches in that it provides guidelines to functionally break down a larger problem into manageable components, which in this case, are translated to code. However, given the digital nature of code, a heavier emphasis is placed on adjustability and extensibility. The software design approach assumes that demands will not only change but also new demands that were not previously part of the original design considerations will appear. Therefore, software design emphasizes the need for flexible structures that can accommodate change quickly and reliably, without impacting the core experience.

While design patterns will provide a detailed level view of code solutions to problems, there are also higher-level designs that are used in design and workflows of code development.[66] Agile development places an emphasis on releasing code in regular, short-term releases, with regular feedback from customers and developing functional code over documentation.[67] This approach to development encourages designing for assured change. Some have critiqued this method for a lack of focus on architecture and that it many not scale well to larger projects.[67]

3.2 Iterative Design and Finding Solutions at the Intersection of Different Approaches

Iterative design, while not a separate type of design, is a process by which a system is developed in multiple iterations, in which the knowledge and results gained from each iteration are incorporated into the next version. This approach to design is used throughout engineering, industrial, experience and software design. A design cycle is composed of design, implementation and evaluation with a heavy emphasis, especially in software iterative design, placed on usability, as the iterations will regularly incorporate user testing.[68] This approach to design has become very integral to technological development as the pace of innovation can be incredibly fast, not allowing for serial development.[69] Similarly, short, iterative design cycles allow for quick adaptability to a range of factors, including technology advancement and evolving user demands. Many times this adaptability is important due to a lack of clarity of the requirements for the final system. Multiple user-focused iterative design cycles can elucidate these requirements.[69] In addition, multiple iterations and testing of the iterations can inherently create checks and balances of components along the way.[70] Typically, iterative design will function such that components are developed in parallel, but conforming to an overall architecture that is established before intensive iterative design begins, therefore parallel components can have a common, established, and stable manner for integration. Iterative design puts a heavy emphasis on prototype development and a regular release schedule of new components, as iteration can continue even after the product is released, and thus the evaluation of system quality will evolve over time.[69] Due to the emphasis on quick release schedules, this design approach will also have an influence on the selection of tools used to build the system, as an emphasis is placed on components that can support quick development and integration.

One widely regarded approach to iterative design is the set-based design (SBD) method that began in Toyota development.[71] This method was created in contrast to point-based design, which divides design into small, repeatable linear steps with the goal of reducing time spent on bad or wasted designs. However, the SBD method allows for more flexibility and assessing multiple alternative design methods, which was found to help create overall better designs. The process is categorized in three steps:

1. Map the design space – Understand the space of alternatives that can be moved forward
2. Integrating by intersection – Find solutions at the intersection of sets of alternatives
3. Establishing feasibility before commitment – Maintain a consistency with the overall design

This approach has been found to have many benefits and has been studied in depth.[71]

One down side of the iterative design process is that due to the pressure of iterative cycles, maintenance is not an overall focus.[69] As a result system documentation, for example, will be lacking which can cause some problems in the late stage phases of the design. In addition, not all iterative design cycles are equal, and sometimes negative design iteration can result.

3.3 Integration of Design Approaches and Resulting Constraints

As previously described, INR systems are complex in nature due to the integration of physical and digital as well as human and computational elements. Therefore, it is argued that any one of the previously described design solutions will not work in isolation, but rather a synthesis of approaches is required. As the Toyota set-based design example demonstrates, successful solutions exist at the intersection of alternative ideas, and the same can be argued for alternative design approaches in comprehensive INR design.

For example, while traditional engineering design can help establish very reliable and manufacturable solutions, it does not offer much help when thinking about the user experience and context of using the system that industrial design and experience design can offer. However, INR systems in the face of all of the complexity need to be stable and reliable in order to offer benefits to the patients and physical therapists. Since INR relies so much on software solutions, and custom software solutions at that, software design patterns and models will be very helpful at quickly creating stable systems. However, this level of detail does not support or drive the design of larger themes of user experience.

In addition to these design approaches, iterative design approaches are critical. Testing with patients can be logistically and financially expensive, and thus needs to be efficiently optimized during the design of the system. However, this should not create a design culture of developing systems in series, only validated through patient testing. In addition, INR systems should not be only tested when there is exact certainty of how the system works. The evaluation, and thus design, of an INR system needs to be iterative. Both system components and integrated systems should be tested and developed in iterative cycles maximizing the appropriate types and amount of user testing. By doing this, the system can continue to develop and provide new information that can drive improvements in future iterations.

Integrating design approaches is not a new challenge, and as a result, there are examples of their result. One example composed by IDEO is an approach called Human Centered Design.[72] As suggested by the title, the approach is motivated to integrate the users of the future result of the design from the very beginning of the design process. Therefore, the end users are involved from the brainstorming stages all they way through multiple iterative design cycles, with the end goal of keeping the cost of the design process low. The first step of the process is for the design team to fully understand the problem they are trying to solve and what assistance the end users truly need. Then through multiple structured design and testing cycles a final viable product is delivered, all the while integrating the end user community. The methodology integrates concepts from engineering, industrial and experience design, and applies them through iterative testing cycles.

3.3.1 Hardware and Software Need to Track and Extract Necessary Features From Tasks

Before hardware and software solutions to INR design can be identified, a hierarchy of features to sense and extract from the patient's use of the system is required. As previously discussed, from neurorehabilitation and motor learning research, an understanding of what the system needs to facilitate in terms of tasks can be built. Then, in an approach similar to engineering design, the manners in which to measure those tasks need to be identified. By identifying these measurements, sensing solutions can be identified as well as clear metrics for how to evaluate the success of each hardware and software component. However, there are additional considerations when selecting hardware and software solutions for INR.

3.3.2 Hardware and Software Should be Designed in Reflection of Desired User Experiences

Similar to how contextual research will impact the task design (which will in turn impact the hardware and software solutions), desired user experiences should also factor into the design of both hardware and software components of the system. This is especially true for an INR system as accessibility constrains should be at the forefront of user experience design discussions. However, just as important is to consider what data may need to be collected to assess the user experience (and how reflections on the experience may impact the further design and refinement of the system).

3.3.3 Hardware and Software Should Avoid Proprietary Solutions

Proprietary hardware and software can be costly. INR systems should react to and incorporate commercial technologies to avoid high costs and obsolescence. The rise of the “internet of things” is producing consumer electronics, such as the Microsoft Kinect and immediate smart phone technology that can provide low cost solutions for INR as well as speed up the overall development of an INR system. This is crucial from two design perspectives. First, it supports faster iterative design cycles as many repurposed consumer technologies have significant technical support from both companies and user communities. Many times pre-existing, open-source solutions already exist to basic computational and sensing problems. Secondly, from an industrial and experience design perspective, the utilization of consumer technology leverages possible user familiarity with hardware. If the user is already familiar with how to use an iPad, as an example, the cost to learning how to use a new system implemented with an iPad will be arguably less, as the user may already be comfortable with the core interface and usage patterns.

3.3.4 Novel Hardware and Software Solutions Should Use Rapid Prototyping Methods and Open Source Communities

Development of novel solutions should leverage new manufacturing paradigms. Hardware design solutions of 3D printing[73] and use of simple sensors and processing solutions such as Arduino microcontrollers not only reduce total system cost, but also allow for rapid prototyping of ideas for quick design iterations. This is crucial from an interactive design perspective, especially within rehabilitation systems, as many times final hardware design will require multiple user testing sessions (both non-impaired and impaired). Responding to these test results quickly is crucial to keep the overall development time and development schedule reasonable. A software design emphasis should be placed on developing open source[74] so that others can continue to develop and stretch the code beyond its original purpose, while continually incorporating new functionalities. Again, this approach allows for faster design iterations, and the usage of knowledge and solutions that already exist. Plus designing for others so they can expand the code's functionality for new purposes helps to keep the design intelligible and readily useful.

3.3.5 Hardware and Software Components Should be Designed Iteratively, in a Modular Fashion, Within a Stable Architecture

As the best practices of INR design are still to be established, distilling design requirements from current interdisciplinary literature research and therapist and stoke patient system need reviews is still the key starting point to INR design. However, as research continues and technological solutions continue to improve, more optimal solutions will likely present themselves. INR system design needs to react quickly to these changes, and should not require the complete redesign of a system in order to do so. Given the requirements and research at a starting point of design, an overall architecture should be designed for how components will work together and integrate. At the same time, this architecture should be somewhat flexible to allow for parallel development and integration of new components or replacement of old ones without major redesign. That being stated, multiple iterations should be leveraged to better understand the architecture and components within. This is especially true in the burgeoning days of INR, as system design will still be heavily based in research and exploration. However, within very complex design challenges (such as INR) it is many times advantageous to seek a better design than wait for the ultimate design. Especially within an INR context, designing the best solution can be difficult, as it requires significant monetary and logistical investment in patient population recruitment and has to factor in the limits of current understanding (both technologic and contextual research). Therefore, iterations need to be tiered: What can be tested with unimpaired subjects? What then needs to be tested with impaired subjects? And, overall, how do research questions fit within these design cycles? Which specific questions can be answered first, and how do they fit within larger multifaceted research questions? Faster iterations can happen with unimpaired subjects, leading to advancement in redesigns. However, iterations with stroke users should not be overlooked and design-testing schedules should look to incorporate this type of user testing.

3.4 Assessment of Adaptive Mixed Reality Rehabilitation System Along Design Principles

When I joined the Mixed Reality Rehabilitation team, they were in the process of beginning a full study of the Adaptive Mixed Reality Rehabilitation System at Banner Baywood in Mesa, AZ. While I was not involved with the development of the system, I did observe many of the patient sessions and in doing so ran the system, which entailed making task and sensitivity adjustments to the experience based on physical therapist desires for the patient. This allowed me the opportunity to see a full experiential system in use and think about its design and implementation. What follow is a summary description of the system as well as examples of how the previously identified task, assessment and design principles can be applied to evaluate a system and its iterative design.

3.4.1 Overview of Adaptive Mixed Reality Rehabilitation System

Adaptive Mixed Reality Rehabilitation (AMRR) is a supervised training system that provides detailed evaluation information and interactive audiovisual feedback on the performance of reach to grasp and reach to touch tasks.[75,76] (Figure 4) It was developed under the same goal of applying interdisciplinary approaches, within mixed reality, to develop a comprehensive tool for INR. The system uses 11 infrared motion-sensing cameras to track 14 reflective markers worn by the patient on the arm and torso (Figure 5). Based on the three dimensional location of the markers, several key kinematic features are used for computational evaluation of the full arm and torso movement as well detailed, real time audio and visuals and a post reach visual summary.

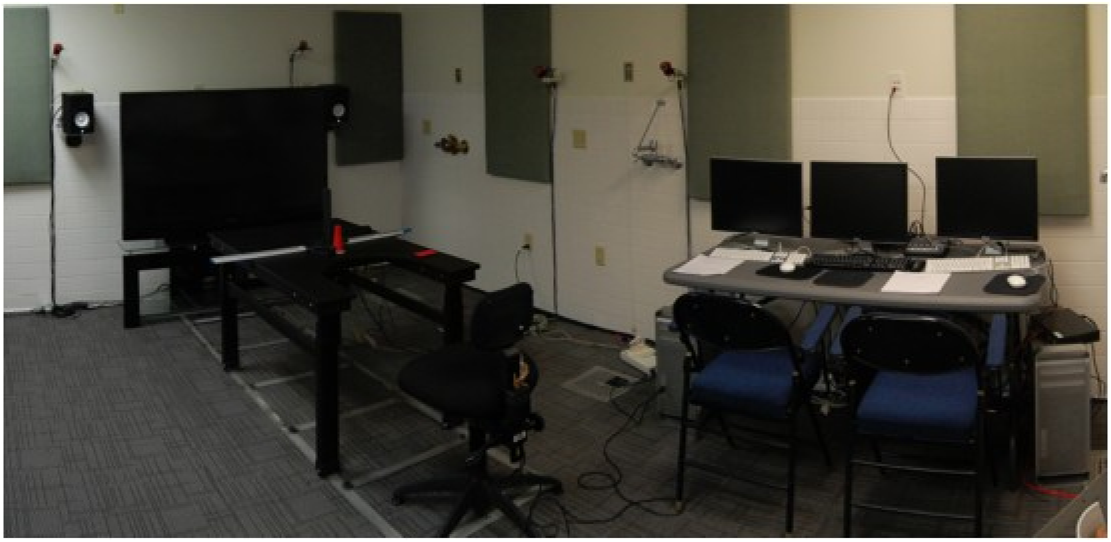


Figure 4 – The physical setup of AMRR

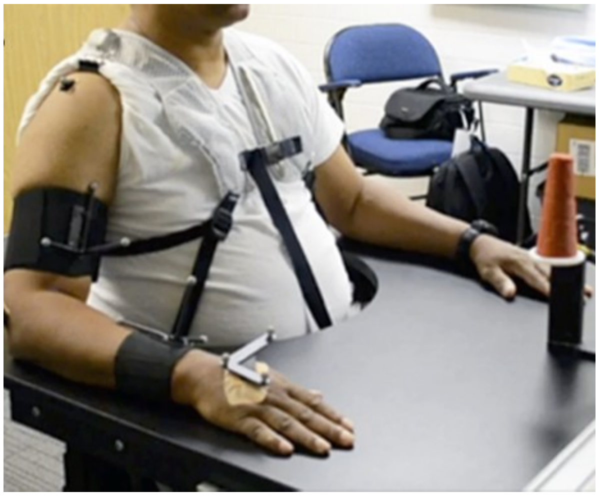


Figure 5 – The marker array worn by the patient using AMRR

The AMRR experience utilizes the presence of three users: patient, therapist, and system controller. The patient sits at a table, wearing a collection of markers along the arm and torso, and performs the tasks in sets of reaches. They generate and respond to the feedback during and after their sets of reaches. The therapist observes all of the patient's movements and provides some feedback to the patient through suggestions for correction as well as helping to interpret the feedback if there is a moment of confusion, especially early on in the training. The therapist also might provide an end of session summary to the patient by translating some of the kinematic analysis that AMRR generates. The therapist also provides feedback to the system by setting the sensitivities of the feedback streams, including determining if a particular stream should be on or off. The therapist also determines the amount of reaches spent on a particular task and the sequence of tasks that will compose a particular session of activity. The system controller acts as an intermediary between the therapist and the system by working with the therapist to translate their goals for the patient into the language of the system's parameters and controls. The controller will live debug issues based on observations of the patient and knowledge of how the system works. The controller is also the sole interface with the system to turn it on and configure it before each patient session. Therefore, the AMRR system assumes not only that the therapist and system controller are continually present during all of the patient sessions, but also that they both have knowledge of how the system works and its range of capabilities.

The hardware architecture of AMRR was designed to take in input from 11 infrared OptiTrack cameras, process the data to generate kinematic features, send those features to feedback engines, and finally archive the data. (Figure 6) This required two Mac computers, one to run the kinematic data extraction from the OptiTrack cameras (which can only occur on a Windows machine), and the other computer to process the kinematic data and generate the audio and visual feedback, as well as archive the data. There was also a separate laptop computer that was setup to exclusively run a web camera to capture video of the patient sessions for later analysis. All of these computers communicated data to one another through multicast communication on an internal network.

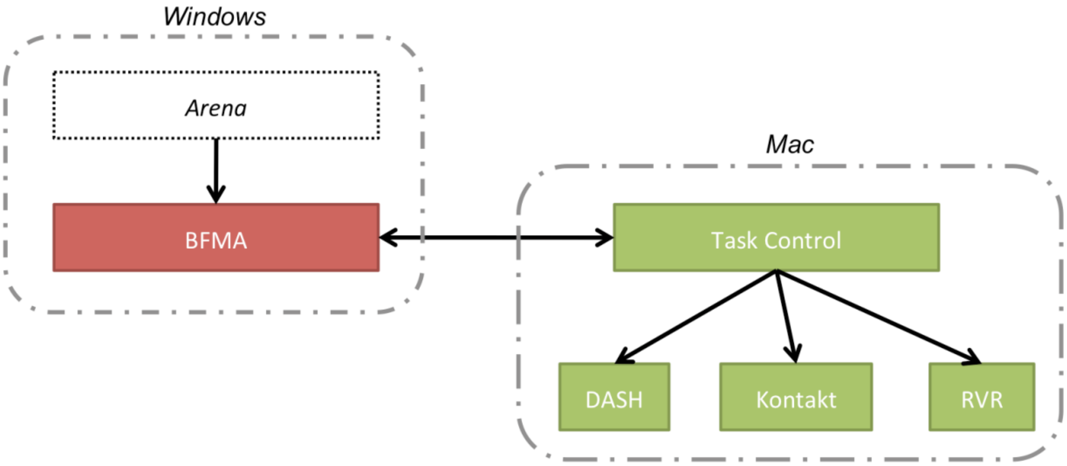


Figure 6 – The software of AMRR was spread across two computers, with the Windows machine running two programs to process incoming data, and the Mac machine generating all of the real time analysis and feedback.

The software of AMRR features a few main programs. On the OptiTrack camera computer, a single program called BFMA was written to generate the core kinematic features based on input from the cameras and some calibration information from Task Control. Task Control served as the master control program to the AMRR experience. It would receive kinematic features from BFMA and process these for analysis and feedback generation. Task Control had a main interface in which all the feedback mapping and task parameters could be adjusted before starting a set. This UI provided the manner in which therapist goals were translated into system functionality. All of the visual feedback was controlled through a DASH plugin, based on received features from Task Control. Audio feedback used a Kontakt interface that responded to commands from Task Control. After the completion of each set, the data was archived to binary files, which another program MRR Offline Tools, could open for further analysis such as the Kinematic Impairment Measure. This program was also the manner in which the therapist could see the raw kinematic data through a series of plots.

3.4.2 Assessment of AMRR Along Design Principles

AMRR was tested across 11 mild to severe stroke survivors within a one month long protocol. In addition, 10 additional stroke survivors comprised a control group that received traditional therapy for the same amount of time as the experimental group. It was found that both the experimental and control group improved significantly in clinical measure scores. However, only the experimental group improved consistently in kinematic measures. The results suggested that AMRR showed promise to improve both functionality and movement quality.[77]

3.4.2.1 Task Generalize to Activities of Daily Living

While improvement in movement quality and functionality improved for all of the participants, assessments of ADLs did not. This seemed to indicate that while participants were improving by clinical and kinematic measures, there was not a perceived direct connection for each of the subjects. The disconnect could be due to the participant having a different set of criteria for improvement than what the clinical and kinematic measures suggest. Relatedly, traditional ADL assessment measures are inherently subjective, and as a result, it was unknown whether more objective measures would have indicated a different story of impaired arm use after therapy.

3.4.2.2 Lack of Comprehensive Experience Design

The fact that perceived improvement did not connect to clinical scores and kinematic measures suggests that the system lacked from a comprehensive experience design approach. As INR systems are rehabilitation tools, the primary user is held to be the stroke patient with the primary goal being to improve the quality of movement and functionality. The main manners in which both of these qualities are assessed are though traditional measures. However, when looking at these measures, they many times require a clinician to interpret the results. For example, demonstrating how a shift in magnitude of the Wolf Motor Function Test (WMFT)[78] correlates to a particular physiological concept. However, to view the assessment from the patient's perspective, the evaluation is more likely in terms of subjective, functional goals, such as: "Can I brush my teeth? Can I reach the garage door controller?" There will likely be correlations between aspects of clinical measures and perceived performance in these specific tasks. However, a truly comprehensive INR design should begin to help make this connection.

The feedback and task design of AMRR was such that the subject's task of reaching to grasp an object was dissociated from the physical to a digital experience. This allows the patient to interact in a space with less constraints and leverage supplied augmented media feedback to better understand and self-assess limb dynamics in a very complicated task. While this dissociation is important in the short term training contexts, it may be providing a long-term disconnect between performance during therapy and during ADL. The system is providing, from a clinical perspective, functional improvement. However, comprehensive INR systems should fully understand the experience of a patient and their own goals, and how those can be realized within the context of a clinically viable system.

In addition, AMRR at its core was designed to focus on component level training rather than functional task training. As the kinematic results demonstrated, the system was very effective at component level training. The system did not address how component level training connected to functional tasks, which would be a crucial component to long-term home-based training.

3.4.2.3 Integrate Component and Complex Task Training

As previously discussed, the results from the AMRR study showed participant improvement in both movement quality and functionality. Results also showed that different movement quality components improved for each participant in similar amounts independent of varied training experiences.[77] This seems to indicate that not only were the designed task and feedback paradigms beneficial in providing component and complex task training, but it could be applied to ranging patient impairments.

The results of the study encouraged that future extensions of AMRR should push the complex task design even further. As previously described, AMRR's tasks were all single object reach to touch/grasp. However, many functional tasks in ADL require complex, multi-phase movement, such as when moving a cup across a table. Therefore, it became a design consideration for future work to take the core ideas of the validated task and feedback design and apply these to more complex task training.

3.4.2.4 Tasks Scale in Difficulty, Adapt in Challenge and Complexity, and Balance Repetition and Variation

These three categories are being discussed in combination because they were all the result of the therapist's supervision and use of provided system constraints:

- Start from simple tasks and move to complex ones to support engagement and active learning and reduce frustration.
- Introduce each new component first in a simple context before a complex context.

- Focus first on functional components (task completion) before addressing movement quality components (torso compensation)
- Repeat each set at least twice but no more than three times.
- Choose any overall path that follows these rules to fit a given patient's impairment profile and training needs.

Working within these constraints, the therapist was able to provide adaptable, beneficial therapy to each patient as shown in the AMRR study results. It was observed that a therapist, with training and time working with the patients, could come to a very good understanding of a complex media system and be able to adjust therapy protocols both in the language of physical task structure as well as the language of the feedback environments. It was also determined that 100 reaches and grouping reaches into blocks of sets, such that the total session did not last longer than an hour, supported patient compliance of the protocol and could be completed by patients of varying impairment levels. This validated the application of general motor learning principles within INR task design, and severed as a starting point for future work.

3.4.2.5 Vary Level and Type of Supervision

As was previously discussed, the environment of training is a very important consideration for INR design, and will directly impact system capabilities. The AMRR system was designed to be a fully supervised system. This design choice was made because it was valued more to test and validate interactive training and feedback paradigms first, before trying to see how these ideas could be automated. Following the concept of iterative design, a system with known constraints and limitations was developed and deployed in the short term, so that the next iteration could answer larger questions with some evidence of core ideas and implementations. The fact that functional and quality of movement improvement was seen in patients with varied types and levels of impairment validated the training paradigms and encouraged the designs to be iterated further for home-based use.

3.4.2.6 Assessments for INR Need to Integrate Functionality and Movement Quality

A kinematics based impairment measure (KIM)[79] was proposed and tested, which combines the key kinematic attributes of reach and grasp tasks into one normalized score. This impairment measure was found to align with WMFT scores.[77] It was also found that the therapist, over time, would use these scores and other kinematic data results to both adjust therapy protocols for each patient as well as provide the patient with evidence for and encouragement towards their progress.

3.4.2.7 Integrative Assessments Should be Achieved Through Combining Kinematics, Therapist Ratings, and Validation by Clinical Measures

The design of the KIM and determining the weights of components that comprise the total integrated score involved therapist input to model how a therapist rates relative importance of components within the overall performance of a task. As previously described, this measure was found to correlate with the Wolf Motor Function Test.

3.4.2.8 Assessments Need to be Associated with ADLs

While some patients in the AMRR studies improved in both kinematics and the WMFT, they did not improve in the MAL and Stroke Impact Scale[80] scores, which could result from the failure of clinic-based practice to translate into daily activity or from the patient's self-perception of limited improvement, despite indices within other measures.

3.4.2.9 Select Hardware and Software Solutions Based on Features to Track

The goal of AMRR was to focus on improving movement quality in tandem with functionality by giving the therapist the ability to focus on any movement quality features relevant to each patient. As a result, AMRR required sensing of up to 40 kinematic features as well as providing feedback on several of these features in any combination desired. Not only did these 40 features need to be sensed through hardware and software solutions, but also they needed to be tracked accurately and reliably to form the foundation of evidence for this style of therapy.

OptiTrack cameras provided reliable, millimeter-detailed tracking of patient movement. The team became familiar and comfortable with its calibration process, how to position the markers on a patient's arm for proper tracking, and how to process the resulting 3D position of the data to extract both low level (e.g. trajectory and speed) and high level (e.g. evaluation of the smoothness of reach speed). As a result, there was substantial existing, and validated, code to extract these features given 3D marker input. This knowledge would serve as a strong basis for future sensing design considerations.

In addition, the feedback engines, developed through a combination of DASH and audio instrument software, were validated. Both of these tools were selected due to their ease of use in creating feedback in response to movement features based on experience of the team. DASH, developed by team member Loren Olson, provided a manner to easily interface with OpenGL capabilities, and thus, served as the tool for all of the visual feedback development. Through the development and testing of these feedback designs, there was a significant knowledge base created for real time and post reach feedback implementation.

AMRR began to identify appropriate task and feedback parameters. These were a collection of adjustable parameters that defined how a movement could be characterized and mapped to feedback. The physical space of the movement was divided into spatial zones. The rest zone marked an area in which the patient would start each reach. The target zone identified an area around the target object in which the patient's wrist and hand should be within when interacting with the object. Between these two zones, a hull space, or sensitivity zone, was created. The sensitivity zone followed an idealized trajectory path that was complied from collecting reaching movements from non-impaired subjects. The hull space was divided into three regions. The zero zone marked variability seen in non-impaired subjects. Any deviation from a reference trajectory in this zone would not be displayed in the feedback. The next zone, between the zero zone and the outer hull, represented the first region of deviation in which feedback would be provided. This ranged from nearly no error feedback (near the zero zone) to maximum feedback (near the hull). Then the final zone, the area outside the hull, represented maximum deviation from the reference and thus maximum error feedback was provided. Similar zones were also setup for other aspects of the movement, and more details can be found elsewhere.[76] Therefore, the AMRR system established, tested and fine-tuned sensitivity zone paradigms that could be applied in future work.

Relatedly, the software design also validated the idea of dividing reaches into discrete states. These states time-divided a reach to help drive the patient's interaction with the system. These states defined a resting state (stop), a state to check if the patient was ready to start by having their hand in the rest zone (ready check), a state to see their reaction time to a go prompt (reaction), a state defined by reaching to the target (reaching), a state defined by successfully interacting with the object (grasp), and finally a state defined by returning from the object to the rest zone (return). The progression through these states provided a very stable interaction and thus informed how the flow of activity would be defined in the software of HAMRR.

3.4.2.10 Avoid Proprietary Solutions

For the task objects, a custom built button and cone were successfully instrumented to detect touch of the button or a grasp of the cone. These objects used capacitive touch and pressure sensors which reduced costs and also allowed for easier integration with the custom software, as opposed to more proprietary options. While these object sensor data streams were not fully real-time integrated with the marker data, a significant amount of sensor data from these objects was collected to validate instrumented objects in therapy tasks.

3.4.2.11 Use Rapid Prototyping and Open Source

Where possible, rapid prototyping and open source solutions were used in the development of AMRR to quicken development time and lower costs. As discussed, the task objects were custom built using low cost sensors. In addition, open source libraries were used for specific aspects of each software component module. However, there are many instances in which proprietary solutions cannot be avoided. OptiTrack cameras, while very expensive and proprietary in the nature of how the data can be interfaced with, were used because they provided reliable, detailed motion tracking information. At the time, neither the technology nor team knowledge existed to develop a more novel, low cost sensing solution. Relatedly, the goal of AMRR was not to design and implement novel computer vision and sensing solutions, but rather to test feedback and interactive training paradigms. Therefore, to iteratively design and test these feedback and task ideas, OptiTrack was selected as a short-term functional solution to provide the necessary data for validation.

3.4.2.12 Design a Modular Architecture

The software design began to show indications that modularity would be the best manner in which to progress in the future. As previously described, the software was divided into a motion analysis engine, a feature extraction and master control engine, and a feedback engine. The motion analysis engine was developed for Windows because Area and Tracking Tools (the motion capture software used to interface with OptiTrack cameras) is exclusively based in Windows. A program called BFMA was developed to listen to the OptiTrack camera data stream and output core motion analysis features to be used in later feedback and kinematic evaluation. However, communication flow of the program was not one directional, as it also responded to commands and some calibration information from the master control program. The feature extraction and master control engine, Task Control, served as the main GUI (Graphic User Interface) to control any component of the therapy experience as well as starting and stopping each component of the interactive therapy session. It was also the core piece that loaded and stored any key control data as well as patient data recordings and extracted features. The feedback engines were separated into the Dash-based visual feedback component and a Kontakt-based audio component. Each engine listened to a feature stream from Task Control and mapped that information to different aspects of the feedback content.

While some modularity was part of the overall AMRR design, the existing modularity was limited and did not allow for easy future extensions. Most likely, this modularity was due to leveraging exclusively the strengths of team members that built each component of the system. In addition, the lack of modularity was also due to a lack of applying software design principles that stress the need to always design for future unknown, yet inevitable, new features and extensions. Since OptiTrack cameras could only be interfaced with Windows software, a computer running Windows is required. It then followed that code to extract features from the data stream could be composed in Windows, further enforced by a team member's expertise in C++ programming. However, other members of the team were more experienced in feedback design and implementation in MacOS programming. Thus the control and feedback components all needed to run on a

Mac machine. This lead to some development bottlenecks as the majority of the team was more familiar with Mac development, and therefore had to rely on one particular person to implement the Windows-based feature extraction. Once this team member left, a lag was introduced in getting a new team member integrated with BFMA.

Similarly, one person also developed the control component exclusively, and as a result, I propose this led to a lack of modularity in the design of Task Control. Instead the program became a "catch-all" for any capabilities that were not generating feedback. All of the adaptation, control, motion analysis (especially for features that drive the live interaction), feedback feature extraction and archiving were all part of one program, with delineation between components that did not support easy isolated development or extension. Therefore, while the program was stable and validated many sensing and analysis components, it demonstrated that for future development, a more modular approach would be more ideal not only for short-term iterative development, but also for long-term research extensions that would only be apparent after future testing.

3.5 Transitioning AMRR to the Home

As has been shown, AMRR was a very successful system. Significant improvements were found in both functional and kinematic measures. The therapist, with some training and time using the system, was able to craft individual therapy sessions across a multi-week protocol. Even given the variations in starting impairments of each participant and the different paths used by the therapist, patient improvements were found.

The system, both hardware and software, were very stable and demonstrated a framework for taking motion capture data and translating the resulting motion analysis into auditory and visual feedback. AMRR also demonstrated the validity of many kinematic measures as well as key components to interactive mixed reality training.

The next goal was to transition these successes into the home environment for long-term training (with the ultimate goal to build a system that could eventually support multiple months of training). However, the focus of this next system would need to change. AMRR was designed specifically for component level training under constant supervision of the therapist with continual adjustments to the therapy completely under the control of the therapist. This model was not going to work for the home. The combination of long-term training as well as the constraint of reduced sensing in a home environment lent more to the goal of higher-level training that focused on complex tasks and encouraging regular therapy and confidence in using the impaired limb. While some of the principles and successes previously identified in AMRR would be helpful in the home, many aspects would need to be redesigned or newly created to address the different training, analysis and hardware environment that a home-based system would inhabit.

CHAPTER 4

DESIGN AND IMPLEMENTATION OF HOME ADAPTIVE MIXED REALITY REHABILITATION SYSTEM (HAMRR)

In the following chapter I describe the implementation of a Home-based Adaptive Mixed Reality Rehabilitation System. The goal was to transition what the design team had learned from the AMRR system into a form that could support long-term training with minimal to no supervision. This required not only rethinking key hardware and software components, but also the protocol needed to progress a patient through the full system experience. My work entailed taking the previously described contextual research and observations of the AMRR system as well as the new goals for a home based therapy system and translate these into a comprehensive system design and implementation. This involved designing and building the overall modular software architecture of the system as well as designing and building many of the individual plugin modules. I also designed a set of therapy protocols that used previously described motor learning and constructivist learning principles to create structured, automated therapy sessions. Section 4.1 provides further details on the problem statement behind the HAMRR design. Section 4.2 provides details on the design and implementation of the system. Section 4.3 reviews the experiment design, which tested the system with eight stroke survivors, each using the system for 15 sessions.

4.1 Problem Statement

The goal for HAMRR was to take the experiment tested knowledge and successes of AMRR and apply them to a device that could be setup in a minimally supervised environment, such as the home, and thus allow for longer durations of training due to the device's convenience. While there were direct connections to be made between the AMRR system and design thoughts of a home based device, there were a few big key differences that the design team did not completely have data driven evidence for their implementation. How should the physical system look and be built such that it could fit in someone's home, retain a similar data resolution, yet minimize the setup and interface required of a stroke survivor using the system on a regular basis? Also, how does the software need to be designed to support automated therapy progressions that utilize minimal user input yet, and at the same time, extract kinematic and user data real time and post activity and use analysis? In addition to these larger questions, there were also multiple more focused research questions from members of the team such as: How do we properly provide multi-layered feedback, both from feedback content design and motion data analysis perspectives? How can object interactions be sensed and evaluated within complex tasks? Overarching all of this is the consideration that the development of each individual component of the system will require varying degrees of testing, both in isolation and in connection with the system as a whole. How can the system design be progressed forward in the face of multiple components being iteratively tested themselves? Therefore, while the team felt fairly certain about some design decisions, many modifications had to be made for this new application.

4.2 Design and Implementation of HAMRR

4.2.1 Task Experience Design

4.2.1.1 Vary Level and Type of Supervision

Arguably, the most significant difference between AMRR and HAMRR was the change in training environment. This manifested itself in two ways: a more variable home environment and less therapist supervision.

In order to transition to the home environment from AMRR, the physical setup needed significant reductions. HAMRR was designed to be a system that could be setup in a space, such as the home, for a period of time and then moved elsewhere for another patient to use. These criteria required an experience design perspective to be adopted more heavily than in AMRR. The primary difference between AMRR and HAMRR was thinking about physical setup changes in terms of what was once permanently setup in a clinical space that would now have to be more contained or temporary in the home. How would the physical design reflect in the regular use of a physically impaired user within the variable space of a home living room?

It would no longer be feasible to rely on permanent installation of components in someone's home. In addition, because each home environment could vary, it was assumed that any required technology (TV, computer, internet connection, etc…) would need to be provided by the system in a self-contained manner. Therefore, the overall goal was to provide a table and media stand to the patient. (Figure 7) The media stand contained all of the computation, visual feedback display and optical tracking. The table was custom built with embedded electronics and places for smart objects to be placed for each therapy session. Finally, a specially designed chair was also provided, with embedded pressure sensors for coarse torso orientation detection. The chair was created as an attempt to move away from using reflective markers to detect torso movement, or at the very least, help filter noise from the torso marker data. While reflective markers can provide very accurate spatial measurements, they can be very susceptible to problems due to camera occlusion, which can be very likely with severe body compensation. More specific details of the implementation of these components will be discussed later.

Figure 7 – The HAMRR system physical setup

In addition to designing the physical setup, user experience was also important to consider for the change in therapist supervision between AMRR and HAMRR. Whereas the therapist was the main source of protocol modifications and adaptations, as well as the main interface between the patient and system for AMRR, in the home this regular presence would be eliminated. Therefore, the same aspects to the training that the therapist would provide during therapy would have to be replaced, or approximated as best possible, in a computational manner. For example, the system would need to drive the therapy (in terms of task sequence and dosage) and provide directions and examples for how to complete each task and interpret the feedback. In addition, the setup required to begin and run a therapy session would have to be limited to what was easily controllable by a patient with minimal outside assistance needed.

4.2.1.2 Task Generalize to Activities of Daily Living

As previously discussed, it was found in testing of AMRR that the connection of gains made in therapy to self-perceived changes in ADLs was not significant. In an effort to change this, it was decided to widen the range of complex tasks available for training in order to have training tasks that are more similar to complex, functional ADL tasks. (Figure 8) This resulted in the introduction of a transportation task, in addition to the previous AMRR tasks. The transportation task required the patient to grasp a cylindrical object and move it to a new location, either directly on the table or elevated off of the table. It was hoped that by introducing this new, more complex task and properly integrating it with the other AMRR tasks that a better connection of the training to ADLs might be possible.

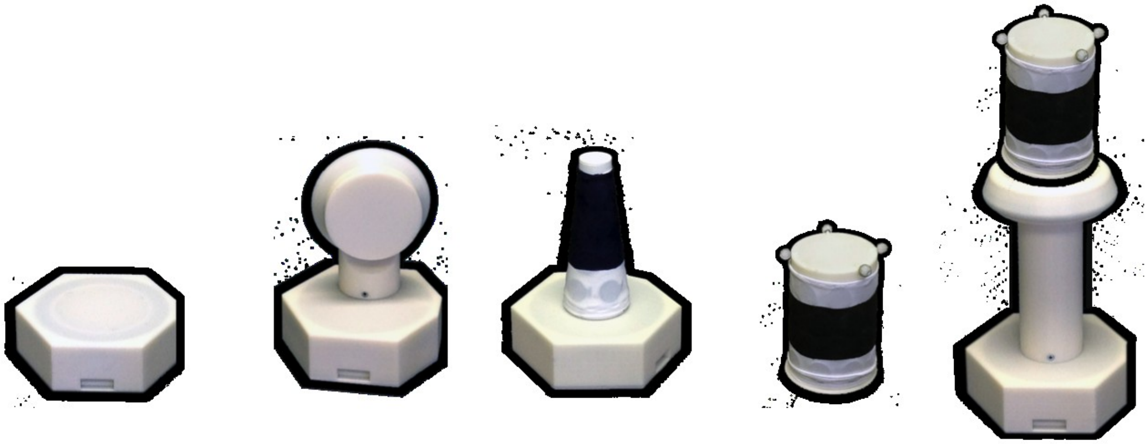


Figure 8 – HAMRR Objects (Designed by Margaret Duff) that range in complexity from: simple reach to touch, reach to slight elevated touch, reach to grasp a cone, reach to grasp a cylinder and transport (both supported and against gravity).

4.2.1.3 Tasks Integrate Component and Complex Training, Scale in Difficulty, Adapt in Challenge and Complexity and Balance Repetition and Variation

The integration of the new complex task in a training protocol was not an insignificant problem. In addition to meeting the previously identified criteria for task design, it also needed to be feasible in a minimally supervised environment. HAMRR was designed to train and provide unique feedback for three different levels of activity, referred to as interaction levels: concurrent and summary feedback per single task (supported and against gravity reach to touch and/or grasp and/or lift tasks), summary feedback per set of repetitive tasks, and summary feedback for complex tasks (transporting an object between two locations). (Figure 9) The details of the feedback structure and content design can be found in Nicole Lehrer's dissertation.

Further, given the more limited sensing capability and reduced supervision (when compared to AMRR), the tasks in HAMRR needed to emphasize more higher-level outcomes, by reducing emphasis on movement quality components and increasing emphasis on building the patient's confidence in using their affected limb. Thus, as previously discussed, tasks needed to emphasize training ADL activities, as well as long term engagement. The protocols that could build this high level confidence using the tasks and feedback available in the system would be very important.

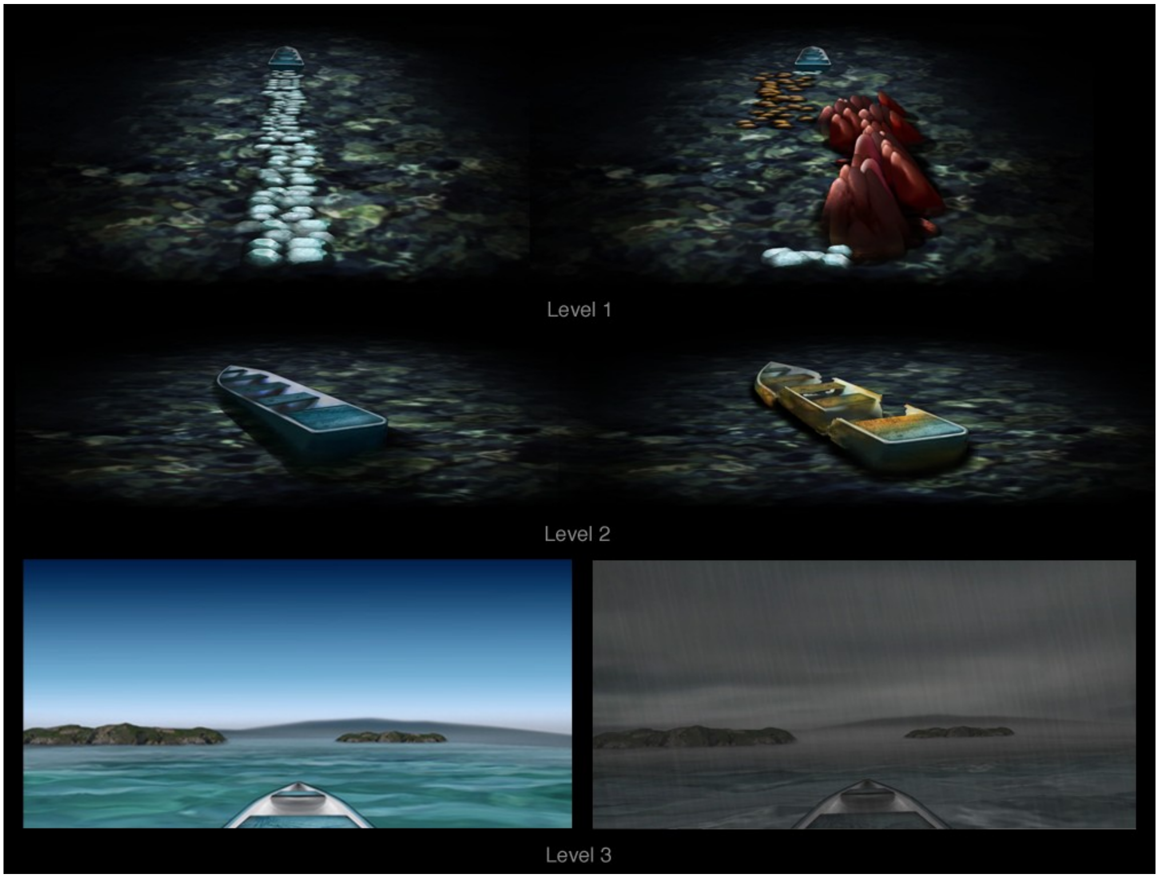


Figure 9 – The visual feedback levels

4.2.2 Therapy Protocol Composition Through Iterative Design

Since not enough information was known about how automated adaptation could be successfully implemented within HAMRR, pre-composed protocols (or scenario paths) were created for the study. From an iterative design perspective, this would allow for the collection of data and testing of semi-supervised training, which is important for forming the basis for future automated adaptation design. In order to have a basic choice in protocol difficulty (yet simultaneously limit the number of possible adaptation decisions), two scenario protocols, or paths, were created. The difference between paths was the weighting of time spent on against gravity transports. One path put a higher emphasis on training off table transports, while the other focused more on on-the-table transports. Thus, even if a patient could not complete the off-the-table transport tasks, the on-table transport focused path was composed to still give them an opportunity to attempt the task, but did not present a large amount of off-table transports in order to reduce frustration.

As will be shown, the exact design constraints for developing a protocol path were not known. Some knowledge gained from the AMRR study, as well as interdisciplinary research, formed a basis for many design decisions. However, these protocols were created as a first attempt at designing paths for unsupervised training using previously identified contextual research in motor learning and constructivist learning.

4.2.2.1 Establish a Flow Back and Forth Between Simple and Complex Tasks

Similar to the musical instruction models previously discussed, the path protocols were designed to train tasks in an overall order of simple to complex. In order to accomplish this, first the collection of tasks was ranked in terms of difficulty. The difficulty of a task can be defined by the combination of two factors: manipulation type and target location. The manipulation type can rank from simple reach to touch to grasp and transport. The location ranged from midline to far ipsilateral on the table, to midline to far ipsilateral off the table. In addition, because visual feedback is explicit and audio feedback is implicit as well as aggregate, sets of reaches with a focus on trajectory feedback were introduced before speed. And the integration of trajectory (Path) and speed (Flow) was introduced before body compensation. In addition, a general rule was established that when new task or feedback elements are introduced, they should be introduced in isolation and then incorporated with the already established feedback environments and tasks. Based on these larger structural considerations, the tasks could be ordered from simple to complex to provide an overall road map to composing a 15-session protocol.

However, within this overarching protocol, a flow was attempted balancing simple and complex movements. For example, in some sessions, a group of objects were used that included both a simple flat object and a more complex cone object. Therefore during a session, the patient could start practicing with a simple reach to touch object and then immediately practice transferring this knowledge to the more complex task of reaching to grasp a cone. Then the protocol would move back to the simple reach to touch object to reinforce this connection in a simpler context.

This hierarchy was also important for composing the condensed training sessions that comprised the first three sessions. These first three sessions were designed to train the patient and give an overview of the system. Since some training sessions were designed to heavily focus on only a few motor elements with one object, it was necessary to introduce the full breadth of the system early on so there was a feeling of experiences yet to be encountered after the early, more repetitive training stages of the protocol. In addition, when introducing someone to a new, complex system, some training time is expected. This training time is in addition to motor learning training and relates to understanding how to control the system, complete tasks and understand the content of feedback. Therefore, these first three sessions were designed to not only show all of the tasks and feedback environments, but also be supervised by a therapist so questions could be quickly answered.

4.2.2.2 Establish Repetition and Variability

Approaches to motor learning stress repetition to reinforce skill acquisition, however, variability is also important. Therefore, the protocol paths were composed attempting to balance these two constraints. Overall, a limit was placed that each set could only contain five reaches. This number was a modification of the setup of the AMRR system, which contained 10 reaches per set. The lower number was selected to allow for more opportunities for variability, as system parameters are changed in between sets. In addition, a given set (with a selected task type, object location and feedback environment) could only be repeated in sequence once. If any component of the task was changed (for example, the location was different but the feedback environment was the same), then the set could be repeated more times. In this way, repetition was represented in blocks of sets where the amount of change was minimal to reinforce practice and learning, but still provide differences where the patient has to apply newly acquired skills in multiple contexts.

4.2.2.3 Establish Reductionist Hierarchies

The ultimate goal of the training is for patients to be able to self-assess their movement and to be able to understand the dynamics of their limb during a reaching activity without the assistance of any additional audio or visual feedback. In order to accomplish this the patient needs to transition from a media-rich environment to a purely physical interaction space. There are a couple considerations for this. First, the frequency of provided feedback needs to fade over time such that the patient does not become overly reliant on the feedback. At the same time, feedback content needs to evolve such that less feedback is provided on individual motor elements of a movement, and more feedback is given on the overall quality of the movement. As previously discussed, it is important to engage the patient in problem solving to support motor learning. Thus, the patient needs to be encouraged to and learn how to self-assess their own movement. Thus the tasks and resulting protocols need to help build skills in which an understanding of movement quality during a complex task can be broken down into individual elements of the movement. Therefore, within the tasks, individual elements need to be shown in aggregation within a complex task, and conversely, the complex task needs to be broken down into constituent components. The feedback and tasks should be structured to establish this connection such that, over time, more and more feedback can be provided about the overall quality of the movement, and the patient can break down that information, through their own self-assessment, into individual motor elements to fix an error in a complex movement.

Therefore, much like the musical instruction models previously discussed, the final form (in this case a complex task) is always instructed along with the constituting elements, such that the aggregation of elements and break down of complex tasks can be demonstrated. As previously discussed, the integration of component and complex training is crucial in motor learning. Therefore, even if the patient initially can't complete a complex transport tasks, the protocols have the patient interact with two simple touch objects in sequence to see how motor elements connect to this simple task. Then, progressively more complex final forms are introduced. Over time, however, the transition within the protocol between low-level component feedback and higher-level quality of movement feedback needs to change. Initially, the protocols needs to be designed such that there is a focus on low level motor elements and how these elements can begin to connect together in progressively more complex tasks. Later in the training, less emphasis should be placed on component feedback and more towards quality of movement feedback. If the reductionist hierarchy has been established correctly, the patient will be able to use this feedback to correct their movement along individual elements.

4.2.2.4 Result of Applying Protocol Composition Principles

The first three sessions were envisioned as a training week. This training week represented the first encounters a patient would have with the system. It was also fully supervised by the therapist. The training week was designed to show most of the core interactions with the system, and as a result the breadth of feedback environments, and allow the patient to ask questions of the therapist. Figure 10 and Table 1 provide some overview information on the total composition of each protocol. Figure 11 shows an example of the non-linearity of training that occurs during the third week of the protocol.

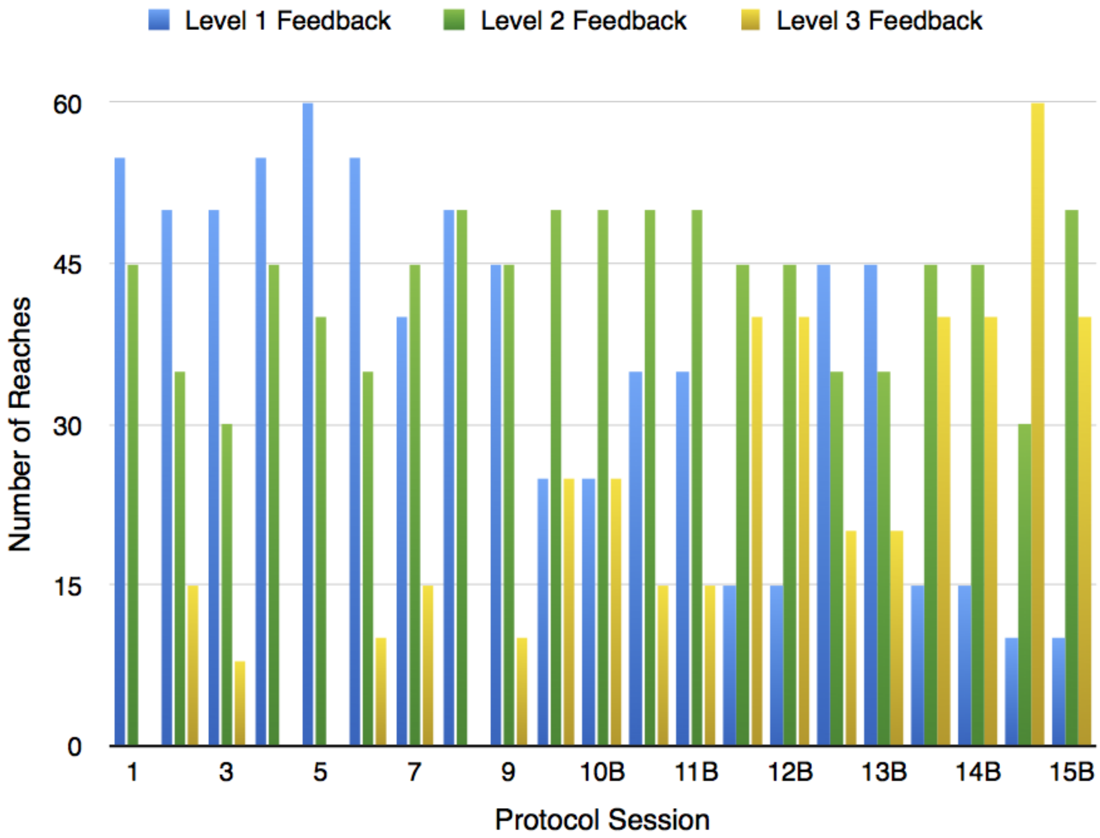


Figure 10 – The number of reaches in a particular feedback level changed as the sessions progressed, such that overall more time was spent in level 3 than level 1 towards the end of the protocol

Table 1. Scenario total reaches overview

Session	Level 1	Level 2	Level 3	Midline Target	Midline – Ipsilateral Target	Ipsilateral Target	Ipsilateral - Far Ipsilateral Target	Far Ipsilateral Target
1	55	45	0	50	0	50	0	0
2	50	35	15	35	15	25	0	25
3	50	30	8	25	8	55	12	0
4	55	45	0	35	0	30	0	35
5	60	40	0	45	0	30	0	25
6	55	35	10	40	10	10	0	40
7	40	45	15	35	0	40	15	10
8	50	50	0	55	0	0	0	45
9	45	45	10	45	0	40	10	5
10 A	25	50	25	35	25	10	0	30
10 B	25	50	25	35	25	10	0	30
11 A	35	50	15	40	0	30	15	15
11 B	35	50	15	40	0	30	15	15
12 A	15	45	40	25	30	35	10	0
12 B	15	45	40	25	0	35	40	0
13 A	45	35	20	45	20	35	0	0
13 B	45	35	20	45	0	35	20	0
14 A	15	45	40	25	40	35	0	0
14 B	15	45	40	25	20	35	20	0
15 A	10	30	60	0	0	30	60	10
15 B	10	50	40	0	0	40	40	20

Path A represents the transport heavy protocol. Path B had fewer transportation tasks against gravity.

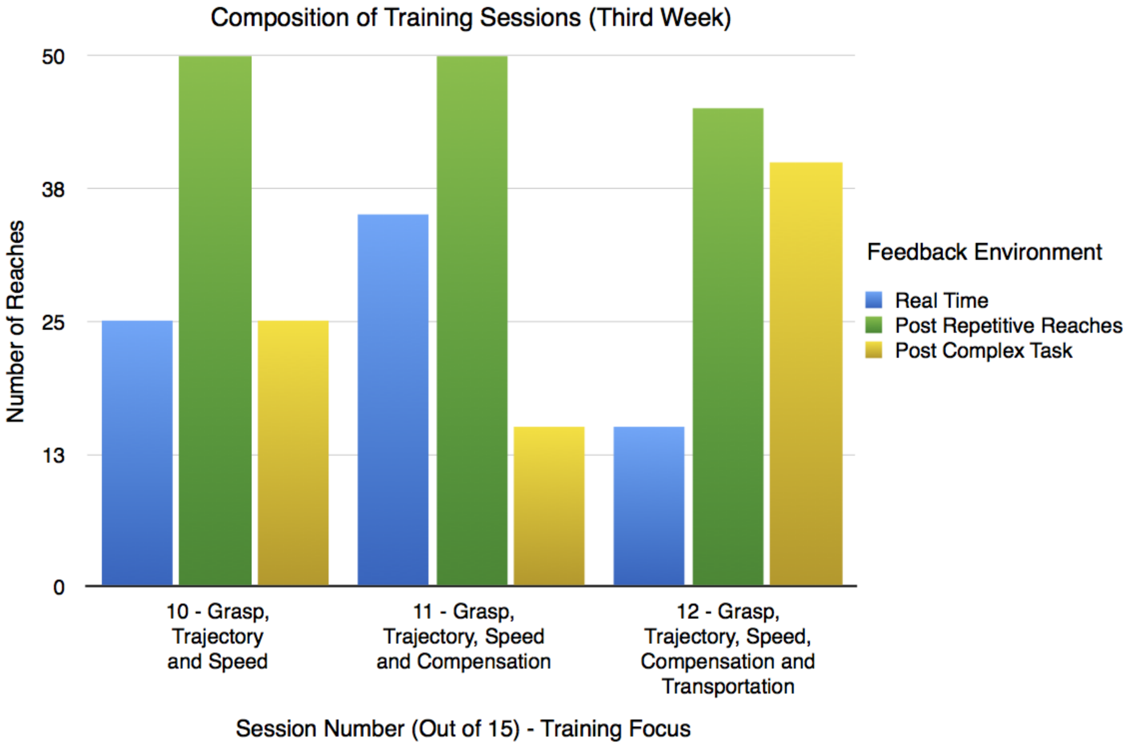


Figure 11 – The progression of feedback levels in the training was not linear. While the goal was to use complex task feedback (Level 3) progressively more, when introducing a new feedback stream (such as compensation in Day 11), real time feedback is increased to help the patient experience the new training

What follow is an overview description of each session in the protocol. Each session had an object interaction focus (Touch, Press, Grasp, Transport) as well a focus on feedback (Path – Trajectory, Flow – Speed, Compensation).

A & B Day 1 – Touch & Path & Flow

Day 1 was the first day the patient was using the system. Therefore the simplest objects were used in what was thought to be the more simple joint space target locations (midline and ipsilateral). The goal of this training session was to show the real time and post reach trajectory feedback and how they connected together by fading between the two. Then speed was introduced to show its connection within the visual feedback. Finally, Level 2 was introduced, mirroring the way in which the real time and post reach feedback was introduced: first Level 2 trajectory feedback, then Level 2 audio feedback, and finally Level 2 tasks with both feedback streams.

A & B Day 2 – Touch & Press & Grasp & Comp & Complex Task

The second day of the training week introduces two more complex stationary objects: the button and cone. The session begins with real time and post feedback as a reminder of the previous session. Next compensation is introduced in isolation, without any other feedback streams on. As previously discussed, introduction of components is not strictly linear, and new task and feedback components are introduced in isolation first. Next in the protocol, trajectory and compensation are focused on in real time and Level 2. Next, speed is reintroduced with trajectory and compensation in real time. This then progresses to Level 2 with the same feedback streams. Next Level 3 is introduced with two of the simpler objects (virtual object and button object). To conclude the set, the feedback fades back to real time, builds up to Level 2 and then concludes with Level 3.

A & B Day 3 – Against Gravity and Transport

As the final day of the training week, this session uses the transport cylinder exclusively and also introduces against gravity reaches. Since the task complexity has leaped dramatically compared to the last session, each feedback stream is reintroduced within the context of the transportable cylinder in a manner similar to the first day. First task completion, trajectory and compensation are shown in Level 1 and then in Level 2. Next, speed is introduced followed by a set of tasks in which the patient has to lift an object up to a set height and place it back. Then trajectory is reintroduced with the other feedback streams within the lifting task. Next a few Level 2 tasks are introduced to focus on compensation and lifting. This culminates in the first transport task of moving the object between two on-the-table locations.

The remainder of the session is used to introduce against gravity reaches and transport tasks. The sets begin with task completion, then trajectory and speed, and then introduce compensation in real time to a fixed off the table object. Then Level 2 is introduced for against gravity reaches followed by an against gravity transportation task.

A & B Day 4 – Touch and Path

Day 4 represents the first session of unsupervised training, as a result the task and feedback complexity is reduced significantly from Day 3. For this day, the goal is to spend time focusing on trajectory, both in real time, post reach and Level 2 for the simplest object: the virtual object. Feedback streams are introduced in real time feedback and fading to Level 2 is interspersed.

A & B Day 5 – Touch & Path & Flow

Day 5 followed a similar task design as Day 4, but introduced speed feedback and showed its connection to trajectory in real time. Interspersed were Level 2 sets in which speed was focused on exclusively.

A & B Day 6 – Press & Path & Flow & Compensation

Day 6 introduced a slightly more complex object: the button. The first goal of this session was to show the connection of trajectory and speed in real time feedback as well as Level 2. The second goal of the session was to introduce compensation feedback both in real time and Level 2 in conjunction with trajectory and speed feedback. After fading between real-time and Level 2 for a few sets, a Level 3 task between the button and virtual object was introduced for the last two sets of the session.

A & B Day 7 – Grasp-Cone & Path & Flow

Day 7 introduced the first intensive training day that required a grasp of a cone. Therefore, the initial sets just focused on task completion of grasping the cone, both in real time and in Level 2. This was done to train in real time what is required for a successful grasp, and immediately connect this to sets in which task completion feedback would not be given in real-time. Next trajectory was introduced within this task, both in real-time and Level 2. Then speed was introduced, both in real-time and Level 2. Then a more complex Level 3 task was introduced between a cone and virtual object. Once two sets of this Level 3 task were completed, a quick fade progression ended the session where a set of real time, Level 2 and Level 3 were utilized.

A & B Day 8 – Grasp-Cone & Path & Flow & Compensation

Day 8 continued with tasks that involved grasping the cone and introduced body compensation. This day primarily utilized Level 2 feedback with an initial block of sets at the beginning of the session with real time feedback to reintroduce the patient to the training and feedback streams.

A & B Day 9 – Grasp-Lift & Path & Flow

Day 9 increased the task completion complexity by utilizing the cylindrical lift object. However, since this was the first full session with this object, it was kept stationary. Also due to the task complexity jump compared to the last session, the real time feedback was reintroduced. The first sets focused on task completion exclusively, both in real time and Level 2. Then trajectory and speed were re-introduced, first in real time and then in Level 2. This fading was completed in two blocks, first at the ipsilateral location and then at the midline location. To conclude the set, an on-the-table transportation set was introduced, followed by a quick fade from real time, to Level 2, ending the session with another on-the-table transport task.

Day 10

Day 10 was the first point at which the two paths split in day-to-day differences.

Path A – Grasp-Lift & Path & Flow

Day 10 for Path A continues with the lift object and introduces a non-transport Level 3 task with the object. The first part of the session fades between real time and Level 2 feedback for both trajectory and speed to reestablish this connection in the more complex joint space of the far ipsilateral target. The second main part of the session fades between Level 2 and Level 3 with a non-transport task used in Level 3.

Path B – Grasp-Cone & Path & Flow

Day 10 for Path B is very similar to Path A, except it uses the cone in place of the transportable cylinder object. The order and selection of feedback environments is the same, only the object used is different. This was done as establishing fading between Level 2 and Level 3 for a more complex object interaction was still important. However, for the simpler task, a simpler grasp interaction with the cone is used.

Day 11

Path A – Grasp-Lift & Path & Flow & Compensation

Day 11 for Path A continued from Day 10 by keeping the transportable cylinder as the main task object, and this time introducing compensation feedback in all levels.

Path B – Grasp-Cone & Lift & Path & Flow & Compensation

Day 11 for Path B had the same task sequence and feedback environments as Path A for Day 11, however, one of the lift objects was replaced with a cone. That way during the session, a break could be provided from having to grasp the cylindrical object, but still use the overall relatively more complex cone object.

Day 12

Path A – Grasp-Lift & Path & Flow & Compensation & On-Table Transport

Day 12 for Path A was an on-the-table transport intensive day. The session begins with a few sets of real time feedback tasks. However, the session is mainly spent fading between Level 2 tasks and Level 3 transportation tasks.

Path B – Grasp-Cone & Lift & Path & Flow & Compensation

Day 12 for Path B is structured relatively similar to Day 12 for Path A except a cone object is used in place of the midline transportable cylinder. Also, where Path A had full on-the-table transport tasks, Path B has non-transport Level 3 tasks between a stationary cylinder object and a virtual object. However, the amount of time spent in each feedback environment is the same as Path A.

Day 13

Path A – Transport Off of the Table

Day 13 for Path A represents the first time off of the table transports have been introduced since the training week sessions. Therefore, first, trajectory and speed are focused on in some initial Level 1 and Level 2 tasks. Then, compensation and lifting tasks are introduced for both off table and on the table locations. The session ends with some transportation against gravity sets and a few break sets with on table lifting.

Path B – Transport On Table

Day 13 Path B has the same overall structure as Day 13 Path A, however instead of transporting against gravity, these tasks are replaced with transports on the table instead.

Day 14

Path A – Transport Off Table

Day 14 Path A begins with some sets of on the table lifting tasks, both in real time and in Level 2. The remainder of the session fades between transportation tasks against gravity and Level 2 against gravity tasks.

Path B – Transport On and Off Table

Day 14 Path B is structurally very similar to Day 14 Path A, however some of the against gravity transportation tasks are replaced with on the table transport tasks instead.

Day 15

Day 15 Path A is primarily comprised of transportation tasks against gravity between the midline and far ipsilateral location. The initial sets feature some real time and Level 2 feedback as warm up tasks, but the remainder of the session is primarily focused on transportation against gravity.

Day 15 Path B is structurally similar to Day 15 Path A, however some of the transport against gravity tasks are replaced with lifting tasks or Level 2 tasks.

4.2.3 Hardware Design

4.2.3.1 Select Hardware and Software Solutions Based on Features to Track

The core idea, from an input/output perspective, of AMRR and HAMRR was similar: track patient movement, extract kinematic features, and provide feedback based on those features.

In the early stages of the design, based on a review of AMRR, we knew the key components of a movement to track were endpoint location, torso orientation and object manipulation. Secondarily, if possible, it would be beneficial to consider the wrist rotation and elbow angle. Similar to AMRR, these features would need to be assessed in real time as well as over blocks of time. Since HAMRR needed to have a much smaller hardware installation footprint with an emphasis on easy system setup, the movement features captured by HAMRR were limited to the higher-level categories of hand spatial and temporal performance, torso compensation and object manipulation.

4.2.3.2 Avoid Proprietary Solutions

As previously discussed in the review of AMRR, OptiTrack, while capable of providing real-time detailed motion tracking, is very expensive. In fact, it is one of the largest financial costs of the system. In addition, from a more logistical perspective, it is challenging to setup and requires a knowledgeable person to calibrate the cameras. Therefore, some lower cost, more open sourced solutions were considered in HAMRR's design process. Initially the focus was spent on video camera tracking of a glyph worn on the hand of the user. However, the computer vision knowledge of the team was lacking and we quickly ran into many problems of video frame rate and occlusion, which did not provide the resolution of features we required for movement analysis. As a result, given the confidence of the team in reflective marker tracking and the desire to focus on testing unsupervised training (not novel tracking solutions), the physical setup was designed such that OptiTrack cameras could be mounted on a provided stand and the number of trackable markers was reduced down to a single marker worn on the wrist.

Similarly, coupled with the initial desire to remove OptiTrack sensing, the torso tracking was initially designed to be handled by a combination of the Kinect camera and a pressure sensor system embedded in the patient's chair.[81] However, as testing continued with the Kinect, it was found that the torso tracking was inconsistent in its noise and stability in tracking the upper body of someone sitting at the system. It was hypothesized that because the Kinect could not see the legs of the user (due to them being covered by the table), the Kinect had trouble tracking the body frame. Therefore, an iterative design decision had to be made late in the development cycle to remove the Kinect tracking as it was not reliably providing the resolution of data that would be required to successfully test the efficacy of unsupervised therapy. Again, based on team confidence and knowledge of OptiTrack, a marker-based solution was quickly substituted, that while slightly increasing the complexity of system setup (the stroke patient would not be able to put a rigid body marker plate on their left shoulder consistently without someone else's help), it did not require the addition of any more hardware: it could use what was already available and provide detailed data. (Figure 12) Again, due to the priority to test unsupervised training and not novel sensing solutions, this was deemed a reasonable design decision to make.

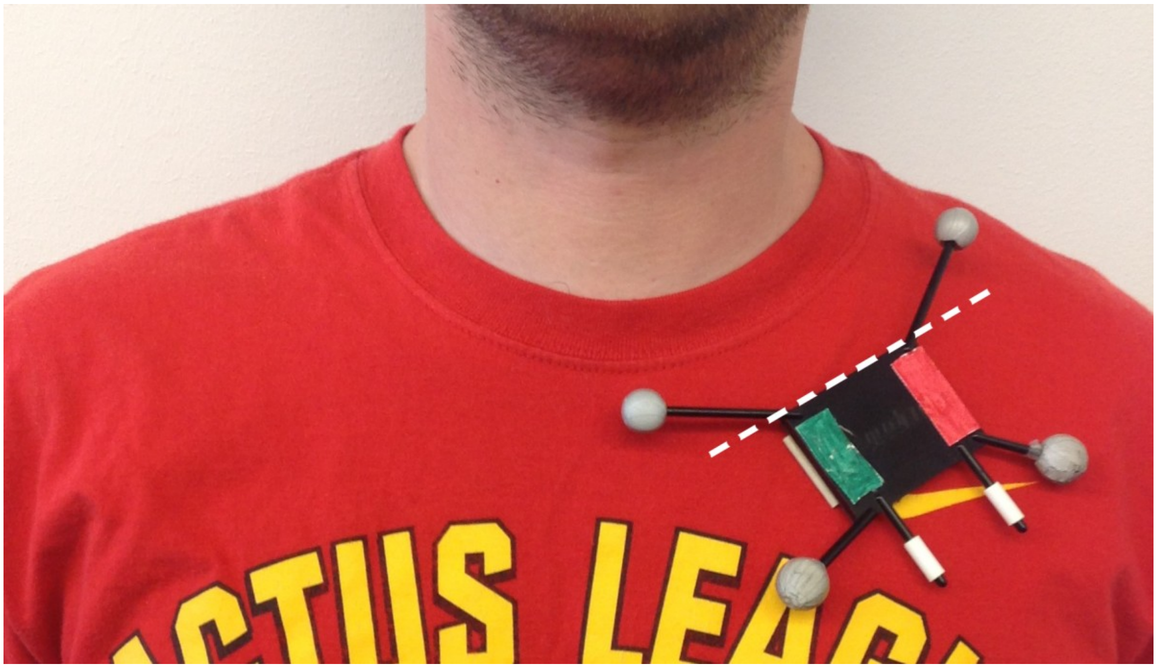


Figure 12 – Torso rigid body used in HAMRR system

4.2.3.3 Use Rapid Prototyping and Open Source

HAMRR further utilized rapid prototyping beyond what was seen in AMRR. Through the work of Margaret Duff, a collection of objects with similar core hardware and physical structural design components were created. (Figure 8) These objects could be used for simple reach to touch, reach to slight elevated touch, reach to grasp, reach to grasp and lift and reach to grasp and transport. Each object had a common structural base and electronics connection, and in conjunction with three custom object sockets arrayed in the table, any object could be swapped across positions for a variety of training scenarios. Each object was built with a combination of 3D printing and low cost sensor and microcontroller solutions. This allowed for quick prototypes to be mocked up in the design process as well as quick manufacturing of multiple copies of each object.

All of HAMRR's code, as will be discussed, was custom designed combing significant new code with verified code components of AMRR and open source libraries.

4.2.3.4 Design a Modular Architecture

In the AMRR system, two computers were utilized: one to run OptiTrack's software and generate motion analysis features, and the other to run the experience, generate high level evaluations and generate the feedback. However, to reduce the footprint, an iMac computer was selected for HAMRR that would run all of the tracking, analysis and feedback generation on the same machine. One of the hurdles that needed to be overcome in this solution is that OptiTrack software only runs on Windows, and all other software developed by the team is based in Mac OS. Initially, the OptiTrack cameras were run on a Windows virtual machine. However, many issues were found with this solution, which impacted all of the analysis and feedback generated by the system. Eventually, the OptiTrack system needed to be moved to an exclusive Windows machine. Therefore, a Mac Mini, boot camping directly to Windows, was selected due to its small size and relatively low cost addition. (Figure 13) Since such a heavy investment was placed on the OptiTrack sensing solution, a decision to ensure the data was clean was extremely important.

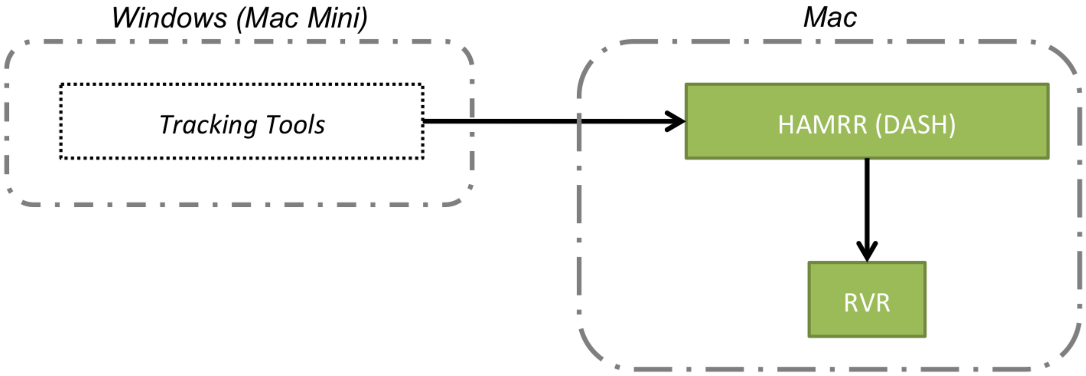


Figure 13 – HAMRR was originally designed to all run on one system. Eventually, motion capture was run on a Windows BootCamp on Mac. All other analysis and feedback was handled within modules of HAMRR, except the RVR video recording tool.

4.2.4 Design Modular Software Architecture and Plugin Components

Instead of taking the AMRR system code and purely modifying it for new needs, the code was designed with a new architecture in mind. The goal was to make the architecture modular. As seen in previous engineering design research, designing products to be modular allows for "economy of scale, increased feasibility of product/component change, increase product variety, reduced order lead-time, decoupling risk, ease of product diagnosis, maintenance, repair and disposal."[81]All of these have analogs in the HAMRR system design consideration. As previously described, all stroke patients are different and system designs need to be able to quickly react to unforeseen needs or limitations, in an iterative manner, without impacting the core experience. Similarly, when system bugs occur (which is an inevitability when testing new hardware and software in a variable environment), components can be isolated and repaired in parallel in a straightforward manner. The design of modular, iterative code with the expectancy of future change is in line with the software design principles previously discussed. In order to design a modular architecture, modules need to be identified based on "similarity between physical and functional architecture of the design and minimization of incidental interactions between physical components."[82]Therefore, a functional decomposition of the problem was done to identify which modular components could be created.

4.2.4.1 Create a Hierarchy Architecture with Established Fundamental System Flow

The functional decomposition was broken down first based on the flow of information: how information would be received, how it would be processed and finally how it would be output. (Figure 14)

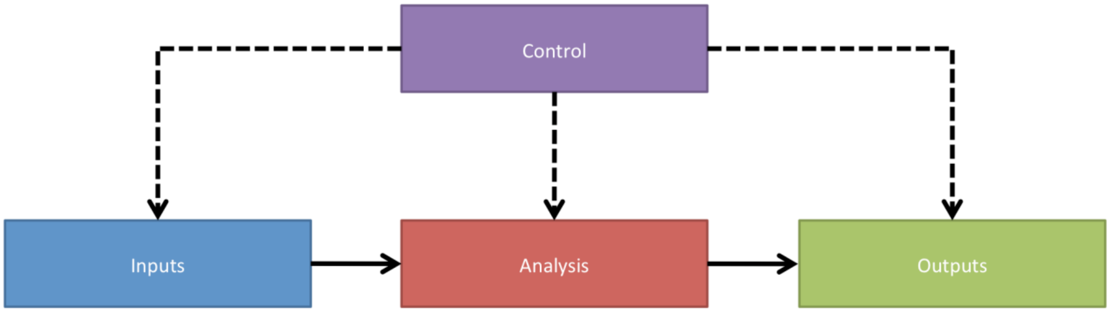


Figure 14 – General functional decomposition of HAMRR

Inputs

Inputs represent the information that is sensed from the environment, patient and therapist. In the case of HAMRR, the inputs were motion capture data from OptiTrack's TrackingTools, touch and pressure data from the tangible objects used in each task, pressure data from the array of sensors on the back of the chair, and touch data from the embedded buttons in the table.

Analysis

Analysis covers the operations that integrate the inputs in an appropriate, time synchronized fashion, and extract varying levels of features, based on the desired type of analysis and varying time resolutions.

Outputs

Outputs represent where the results of analysis need to go. This includes all of the information that is presented to the patient and possibly observing therapist through the audio and visual feedback, as well as the archiving of data for later analysis.

Control

Finally, there is a necessary component to control all of the other modules. It serves as the main interface to translate therapist goals as well as therapy protocols into system parameters that dictate the behavior of all other sub components. It controls the flow of input data, the manner in which data is processed and archived, and the sensitivity of feedback mappings. This module also provides the main user interface for a controller or therapist to reach all of the underlying system parameters and sensitivities to make changes on demand if needed.

4.2.4.2 Connect User Experience Design to Modular Code Design

In addition to the functional decomposition of the core, unique code modules that would require implementation, another layer that wraps on top of the previous four points needs to be considered: user experience. The previous four points primarily consider the system internal connections between hardware and software, but creating an appropriate user experience that fulfills the previously described design constraints is also important.

The need to integrate user experience considerations with software design is an example of where the Toyota set-based design approach can be very helpful in INR. A solution for the design of the software and experience using the system exists at the intersection of these two design criteria. They cannot be approached in isolation or in series, but rather should be iterated on as an integrated approach.

4.2.4.2.1 Design Required System Inputs for Controls so User Interactions are Minimal Yet Complete

It was a crucial design consideration to keep the patient required inputs or interactions to a minimum, due to the varied physical impairment of each participant. Therefore, the core idea was to have the patient adjust their own chair (which was light and had appropriate numerical markings on the floor for where the chair should be located), sit down in the chair, place on an elastic wristband with the single reflective marker (much like a wrist watch), receive assistance in placing a rigid body torso marker place below their left shoulder, and run through an entire automated therapy session. It was decided to provide the patient with assistance for the torso plate, as well as configuring the system before therapy because the primary goal was to collect clean data for answering the larger training research questions. Also it was assumed that in most cases, a patient would have the assistance of a spouse, family member, or friend to help with these steps if they were not automated in the future.

Given this core experience, the hardware and software had to be designed accordingly. In terms of inputs, this required that the system be able to constantly monitor the quality of the marker input and stop the interaction and notify the patient (with a prompt for correction) if the data was prohibitively noisy. This was also true for the transportable object. The system would need to be aware of where the object was in the space of the table and be able to notify the subject if the object was in the wrong location with instructions for correction. Similarly, at the start of each session, the system should check each object socket to determine if the objects for a given session were correctly configured, and provide any needed instructions for correction. Also, the patient should have full control of the ability to take a break between sets. Therefore the software would have to translate input from the two table buttons (Figure 15) to commands such as: start the next set, play the next demo or replay the previous demo. In this way, the required input was kept to a minimum, but full control was given to the patient.

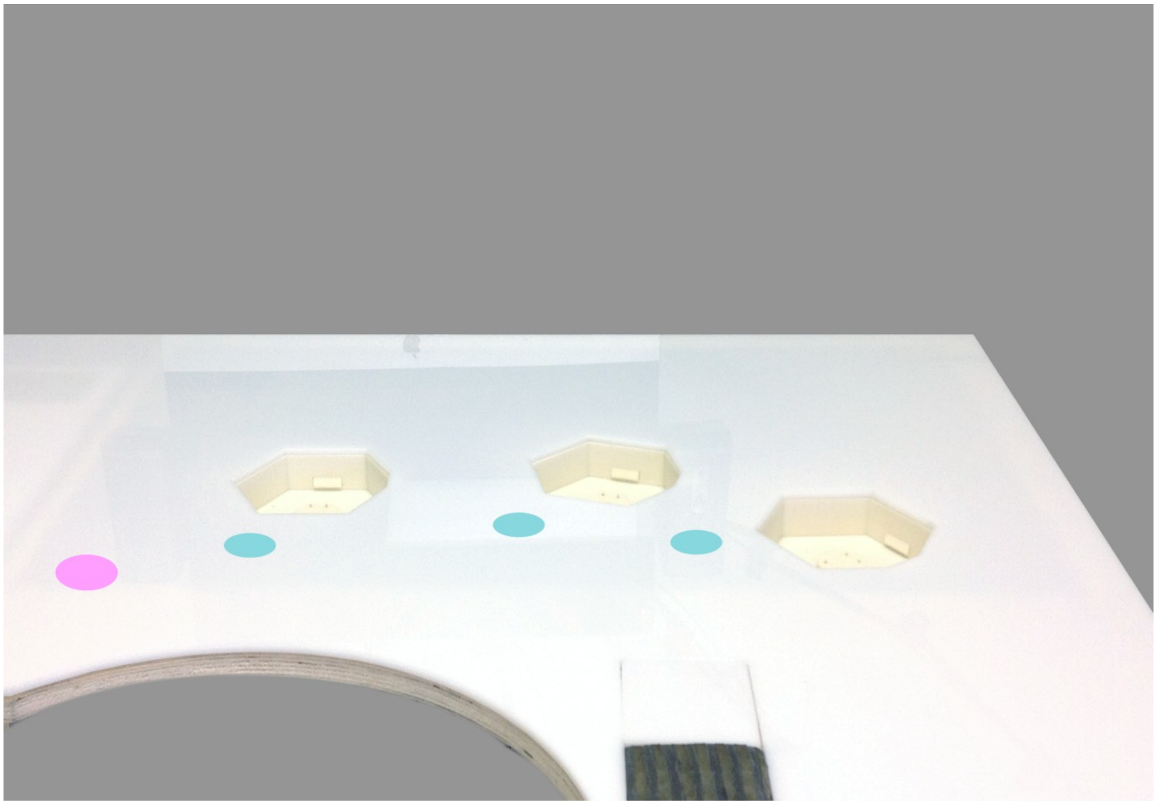


Figure 15 – Table buttons used within HAMMR to control progression through the onscreen instructions as well as to release objects from the table sockets

4.2.4.2.2 Design Feedback that Support and Engage Long-Term Constructivist Learning Experiences

The connection of the analysis and the user experience primarily provided constraints for the feedback design. As this was the work of Nicole Lehrer with Todd Ingalls, it will not be presented in detail here. However, the goal was to convey the results of analysis in an intuitive, constructivist manner as previously discussed.

4.2.4.2.3 Design System State Feedback for the User to Understand their Inputs and Choices

While the user experience outputs heavily focused on the design of the feedback, it also required considering what additional information would be important to give to the patient as they are progressing through therapy.

As previously discussed, the system would need to interrupt an interactive set to report to the subject when errors were encountered, with proposed solutions. In addition, given the variability of training across sessions, it was also important to provide the patient with some sense of the overall progression about how far along in a given session they were at the completion of a set.

4.2.4.2.4 Design a Control Component that Automates the Experience with Minimal Input

As previously discussed, the patient should have full control of the ability to take a break between sets. Conversely, there was no need for the patient to control any of the minute details of the system level parameters, and therefore these sensitivities were wrapped up as part of the system automated control.

4.2.4.3 Translate Functional and User Experience Design Decomposition into Modular Code Plugins

The high level functional decomposition begins to identify where dividing lines between modules exist. However, there are further functional differences that can be isolated. These functional differences need to be identified in order to support faster developed and more maintainable software solutions. By modularizing the code into plugins, functionally, the code can be more easily separated by the designer and thus extended into different directions. However, by utilizing a modular design and integrating software design principles that stress flexible, yet stable architectures, the code can be developed and tested faster as well as be more maintainable, which is crucial in iterative design cycles.

The application of software design approaches resulted in creating a series of plugins. (Figure 16) Due to the nature of the DASH interface, writing individual plugins that can be loaded on demand was a natural design direction and therefore supported the creation of functionally unique plugins. The DASH plugin structure was chosen based on team familiarity with Objective-C, and primarily because significant work had already been done to create visual feedback through DASH, and therefore to save development time, all of the modules were written as DASH plugins. For more detailed information on each plugin, including class-level descriptions of the code, refer to Appendix A.

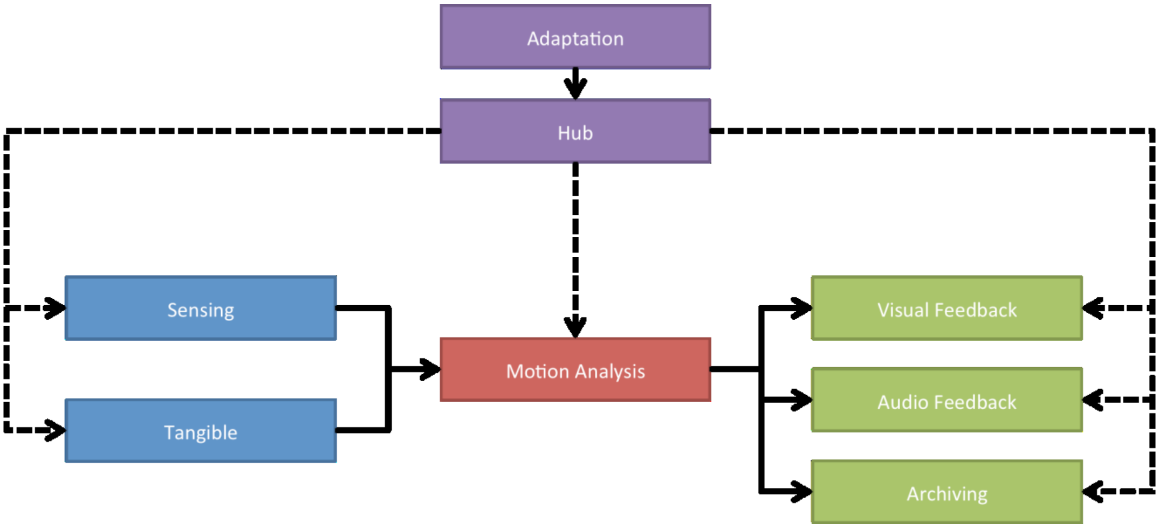


Figure 16 – Plugin Decomposition Diagram

Inputs

4.2.4.3.1 Sensing Plugin

Design and Implementation by Michael Baran, using an OptiTrack parsing library from Joseph Junker

The Sensing Plugin receives raw camera frames from OptiTrack and extracts the inside marker data for both individual markers and rigid bodies. It also receives the data from the chair sensor system and button selection input from both the system dialog buttons and object release buttons. All of the collected data gets stored into data objects that are sent out to other plugins and archiving.

4.2.4.3.2 Tangible Plugin

Design and Implementation by Michael Baran, using data parsing functions by Margaret Duff; All original tangible object code by Margaret Duff; Modifications by Assegid Kidane and Michael Baran

Similar to the Sensing Plugin, the Tangible Plugin is the main handler for all of the smart objects that are used in interactive therapy sessions. It has interfaces for all outgoing messages to the objects and parses incoming sensor data into data objects for other plugins to use. It also controls, through specific outgoing commands, all the cuing of visual feedback within the objects. The smart objects themselves are programmed separately through the Arduinos. The Tangible Plugin code is structured such that all of the smart objects types have similar interfaces so that modularity could exist within the Tangible Plugin, which would simplify future integration of new objects.

When testing the integration of marker data (from the Sensing Plugin) with the tangible data (from the Tangible Plugin) it was found that if the tangible data was sampled at the same rate as the cameras, oversampling would occur. Therefore, the Tangible Plugin was designed to respond to the Sensing Plugin, such that the tangible data is sampled at a fractional rate of the cameras.

Analysis

4.2.4.3.3 Motion Analysis Plugin

Design by Michael Baran

Implemented by: Rajaram Singaravelu and Michael Baran, using analysis functions from Yinpeng Chen and Long Cheng

The Motion Analysis plugin is the main data engine that performs all the analysis and feature extraction. It has various engines for analysis based on the type of task. Each engine is similar in that they use similar or the same feature calculations. However they differ in how each reach is divided into states. These reach states were based on the divisions of a reach previously tested in the AMRR system. However, some modifications needed to be made for the new types of tasks (blocks of reaches without feedback and tasks involving the movement of an object). There is an analysis engine for real time, post reach, post group of reaches and post transport of an object. There are also smaller variants of these specific for recording test data, either with a patient or non-impaired user, which removes the feedback interaction.

Within each of these engines, wrist and torso marker, chair pressure data and tangible object data is received. Frame level and task level kinematics and features are calculated, and these results are stored into a collection of data objects. The objects are then routed to the visual feedback, audio feedback and tangible plugins for all of the feedback to the user, as well as to the archiving plugin for storage.

The Motion Analysis plugin also handles all of the reference trajectories and speeds for a particular object location or transportation task. In addition, the Motion Analysis plugin collects the marker calibrations for both the wrist and transportable object location and has a separate GUI interface to control the collection of calibrations. The plugin loads the existing calibrations based on a subject ID and, within the GUI, shows the user which of the objects currently have existing calibrations, thus helping in identifying which calibrations need to be completed. As will be discussed later, this GUI was a later addition to the design of the Motion Analysis plugin to aide physical therapists running the system on their own.

Outputs

4.2.4.3.4 Audio Feedback Plugin

Design and Implementation by Todd Ingalls

The Audio Feedback plugin handles all of the audio presented to the user, except the narrated demos. It provides real time, post set, and post transport task feedback to the user based on the analysis received from the Motion Analysis plugin. The details of the code design will not be discussed here, as it was solely the work of Todd Ingalls.

4.2.4.3.5 Visual Feedback Plugin

Design and Implementation by Nicole Lehrer

The Visual Feedback plugin handles all of the visuals presented to the user, including on screen instructions and demos. It provides post task, post set and post transport tasks feedback to the user based on the analysis received from the Motion Analysis plugin. The details of the code design will not be discussed here, as it was solely the work of Nicole Lehrer.

4.2.4.3.6 Archiving Plugin

Design by Michael Baran

Implementation by Rajaram Singaravelu and Loren Olson

The Archiving Plugin saves all received data objects to an SQLite database. Initially, the Archiving Plugin was written to save data to binary data files as well as text files.

This was based off of the AMRR model for archiving data. While this was a good start, it required a dual file format save, as the binary file format required a specific program to open, where as the text file could be read more easily by other applications. To remove this duplication, an SQLite database format was adopted which would allow for structurally organized data tables that could be opened and previewed much faster. This helps in quickly looking at data during iterative testing and removes any necessary programming to open a binary file and parse the data.

The Archiving Plugin used the user, session and set id as the main index. It saves the marker calibrations, all data frames from real time and post task analysis, and the raw tangible data frames. It also queries an existing database file at the start of the program to determine which set to begin with for a given session Id. This ensures that if a session needs to be restarted no sets will be incorrectly repeated. This decision was made from an iterative design and software design perspective that the code is experimental and will encounter weeks of patient use that it had not previously. Thus unforeseen problems may be a possibility and In order to recover from and diagnose these problems quickly, the system needs to have a basic memory of the progression of the patient session, where the problem occurred along that progression, and where to resume the session upon startup.

Control

4.2.4.3.7 Adaptation Plugin

Design and Implementation by Michael Baran

The Adaptation Plugin serves as the main control for the system, both internally within the overall architecture and externally for the user though a GUI. It loads an XML file, which contains all the high level parameters and sensitivities settings for a given set within a session. Originally, similar to early Archiving Plugin design, these settings were saved in binary data files. As previously discussed, while this initial design worked, the binary files could not be easily or quickly edited. Therefore, the more transparent and better hierarchical structure XML file format was used for this application. Similar to the use of SQLite, XML could be opened by numerous applications and thus could be edited and created very easily.

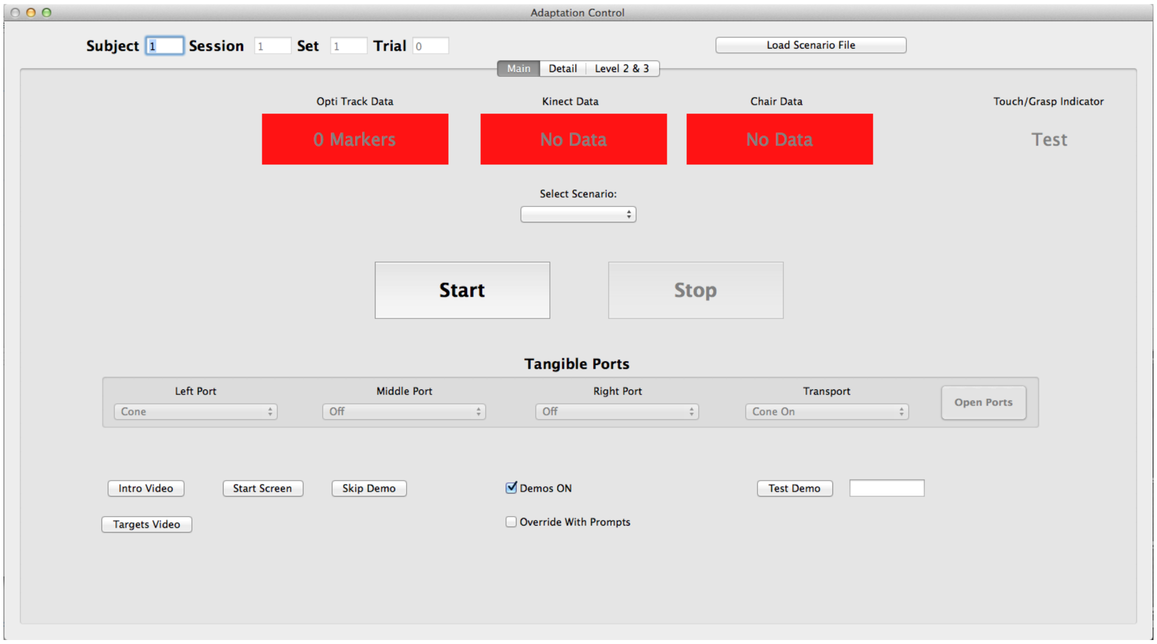
The Adaptation Plugin takes the XML file held sensitivity values and loads them into a data object and this information is broadcasted out to the plugins that need the information. For example, the sensitivity of the trajectory zones are loaded and sent to the Motion Analysis Plugin. In addition, it also serves as the main controller for the demos presented to the user. The plugin keeps track of which demos have been played to the user as well as when enough feedback attributes change between sequential sets that the user needs to be presented with a specific demo before the next set begins. The GUI allows for manual control of any aspect of the task and feedback, including sensitivity parameters for all feedback mappings, and contains a manual start and stop control.

Adaptation Plugin GUI

The Adaptation Plugin GUI serves as the main GUI for the use of HAMRR. It is divided into three main panels and a header.

Header

The header (Figure 17) shows an overview of which set is currently being run or about to run. It gives an indication of the Subject ID, Session ID, Set ID and Trial ID (during a set). This is provided in order to ensure the correct scenario XML file has been loaded as well as to know where exactly the patient currently is within a session and set.



Subject 1 Session 1 Set 1 Trial 0

Left Port: Cone
Middle Port: Off
Right Port: Off
Transport: Cone On

☑ Demos ON
☐ Override With Prompts

Figure 17 – Panel 1 of Adaptation GUI

The header also contains a button to load XML scenario files.

Panel 1: Main

Panel 1 is the main panel for controlling the system. At the top are three data state indicators for the OptiTrack data, Kinect data and Chair data. These provide color-coded status updates to a system controller on the state of each data steam. Also provided is a touch/grasp indicator. This indicator shows if the object is being touched or grasp. It was provided as a way to troubleshoot if the patient is interacting with the object correctly or if an object is not functioning correctly.

The Main panel also has a pull down tab to select a specific set from the scenario file. This provides manual control to skip or repeat a set as needed.

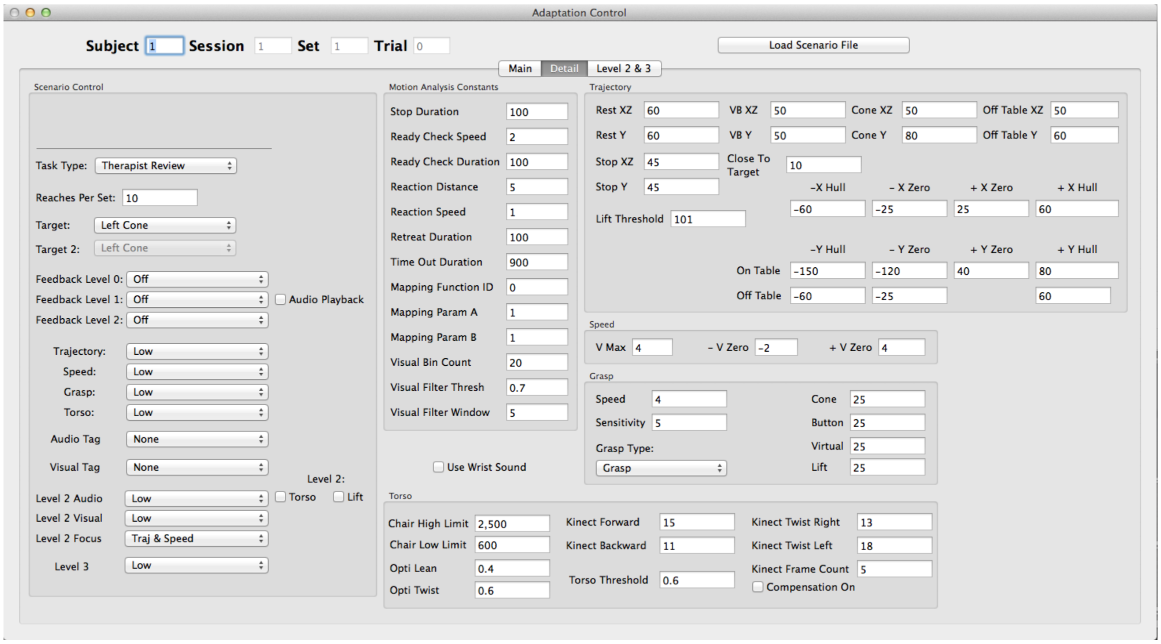
In the center of the main panel are the main start and stop buttons. These are manual controls that are used if the system controller wants to manually stop or start a set. Normally, the Adaptation Plugin automates this process.

Below the main Start and Stop buttons are pull down tabs for the tangible object ports (left, middle, right and transportable object receiver). These pull down tabs are not active, but rather show the ports that are currently in use for that particular session. As discussed in more detail in Appendix A, it was decided for port opening and closing safety to fix the port to a particular object once the XML session file was loaded. However, these pull down tabs were originally intended to provide the system controller with the ability to swap different objects at a particular location during the session without restarting HAMRR. A further discussion of this design will be discussed in a later section.

At the bottom of the Main panel are a series of one off controls. These buttons play the study intro video, play the target insertion demo video, and play the target location setup screen. There are also controls to skip or replay demos, duplicating the functionality of the table buttons the patient uses. Finally, there are two check boxes to turn all demos off or force all demos to use prompts. This was provided as a way to recover from a crash restart quickly. When the system restarts, it does not have a memory of which demos have been completed, therefore it will by default, play all of the full-length walkthrough videos. To skip this, the control to force only text demo prompts to be used can be selected.

Panel 2: Detail

Panel 2 (Figure 18) provides access to some of the detailed system parameters that are part of a selected set. All the controls are grouped into boxes of similar type and function.



Subject 1 Session 1 Set 1 Trial 0

Scenario Control

Task Type: Therapist Review
Reaches Per Set: 10
Target: Left Cone
Target 2: Left Cone
Feedback Level 0: Off
Feedback Level 1: Off ☐ Audio Playback
Feedback Level 2: Off
Trajectory: Low
Speed: Low
Grasp: Low
Torso: Low
Audio Tag None
Visual Tag None
Level 2 Audio Low
Level 2 Visual Low
Level 2 Focus Traj & Speed
Level 3 Low
Level 2: ☐ Torso ☐ Lift

Motion Analysis Constants

Stop Duration 100
Ready Check Speed 2
Ready Check Duration 100
Reaction Distance 5
Reaction Speed 1
Retreat Duration 100
Time Out Duration 900
Mapping Function ID 0
Mapping Param A 1
Mapping Param B 1
Visual Bin Count 20
Visual Filter Thresh 0.7
Visual Filter Window 5

☐ Use Wrist Sound

Trajectory

Rest XZ 60 VB XZ 50 Cone XZ 50 Off Table XZ 50
Rest Y 60 VB Y 50 Cone Y 80 Off Table Y 60
Stop XZ 45 Close To Target 10
Stop Y 45
Lift Threshold 101

	–X Hull	– X Zero	+ X Zero	+ X Hull
	-60	-25	25	60

	–Y Hull	– Y Zero	+ Y Zero	+ Y Hull
On Table	-150	-120	40	80
Off Table	-60	-25		60

Speed

V Max 4 – V Zero -2 + V Zero 4

Grasp

Speed 4
Sensitivity 5
Grasp Type: Grasp
Cone 25
Button 25
Virtual 25
Lift 25

Torso

Chair High Limit 2,500
Chair Low Limit 600
Opti Lean 0.4
Opti Twist 0.6
Kinect Forward 15
Kinect Backward 11
Torso Threshold 0.6
Kinect Twist Right 13
Kinect Twist Left 18
Kinect Frame Count 5
☐ Compensation On

Figure 18 – Panel 2 of Adaptation GUI

The Scenario Control box has all of the highest-level controls (they are also all of the features within the ADScenarioData object). Through these controls, task and target parameters can be adjusted as well as feedback streams and coarse sensitivities.

The Motion Analysis Constants box gives access to the parameters that are more constant and do not change across patients and sessions.

The Trajectory box shows all of the detailed sensitivity parameters that are related to spatial aspects of the tasks, including rest zone, grasp zone, and trajectory zero zones and hulls.

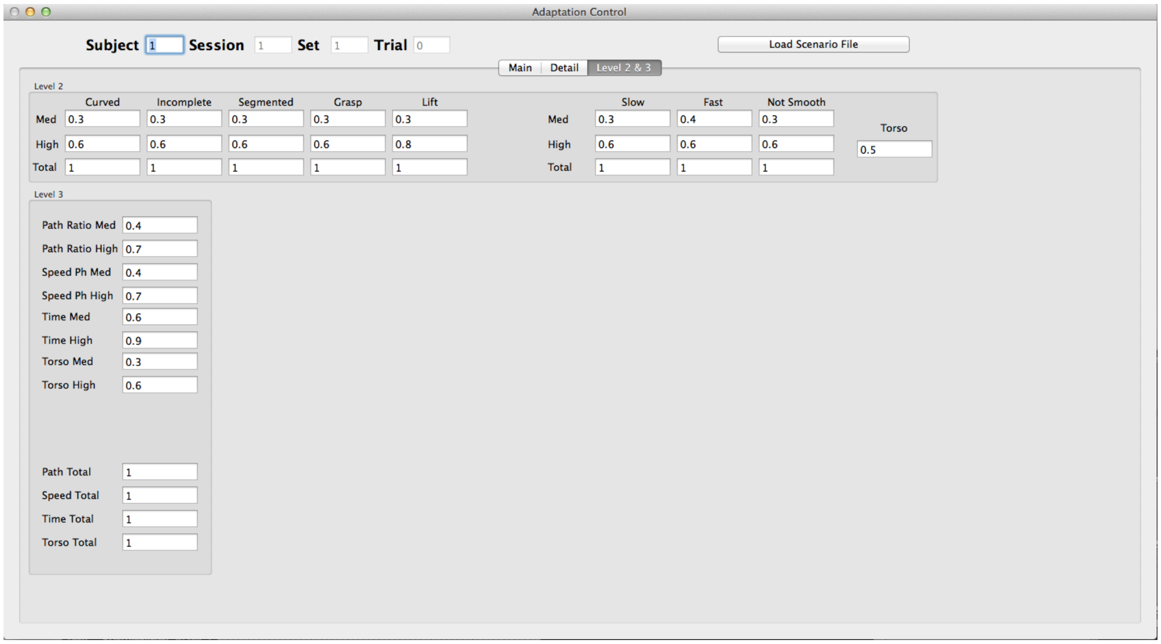
The Speed box shows all sensitivity parameters related to the speed aspects of the task.

The Grasp box shows all of the sensitivity parameters that are related to object interaction and determining if the object interaction is a success.

The Torso box shows all of the sensitivity parameters related to torso compensation analysis. It has sensitivities for interpreting both OptiTrack data as well as chair sensor data.

Panel 3: Level 2 and 3

This final panel (Figure 19) holds all of the sensitivity parameters for Level 2 and Level 3 classifier analysis. Within each level, the parameters are grouped into spatial and temporal features.



Subject 1 **Session** 1 **Set** 1 **Trial** 0

Level 2

	Curved	Incomplete	Segmented	Grasp	Lift
Med	0.3	0.3	0.3	0.3	0.3
High	0.6	0.6	0.6	0.6	0.8
Total	1	1	1	1	1

	Slow	Fast	Not Smooth
Med	0.3	0.4	0.3
High	0.6	0.6	0.6
Total	1	1	1

Torso 0.5

Level 3

Path Ratio Med 0.4
Path Ratio High 0.7
Speed Ph Med 0.4
Speed Ph High 0.7
Time Med 0.6
Time High 0.9
Torso Med 0.3
Torso High 0.6

Path Total 1
Speed Total 1
Time Total 1
Torso Total 1

Figure 19 – Panel 3 of Adaptation GUI

4.2.4.3.8 HAMRR Hub

Design by Loren Olson and Michael Baran

Implementation by Michael Baran

The hub acts as the main connection between the plugins so that the Adaptation Plugin's protocols can reference the appropriate plugins. It is also the last plugin to load, and therefore triggers further start up functionality of the Adaptation Plugin to begin a session.

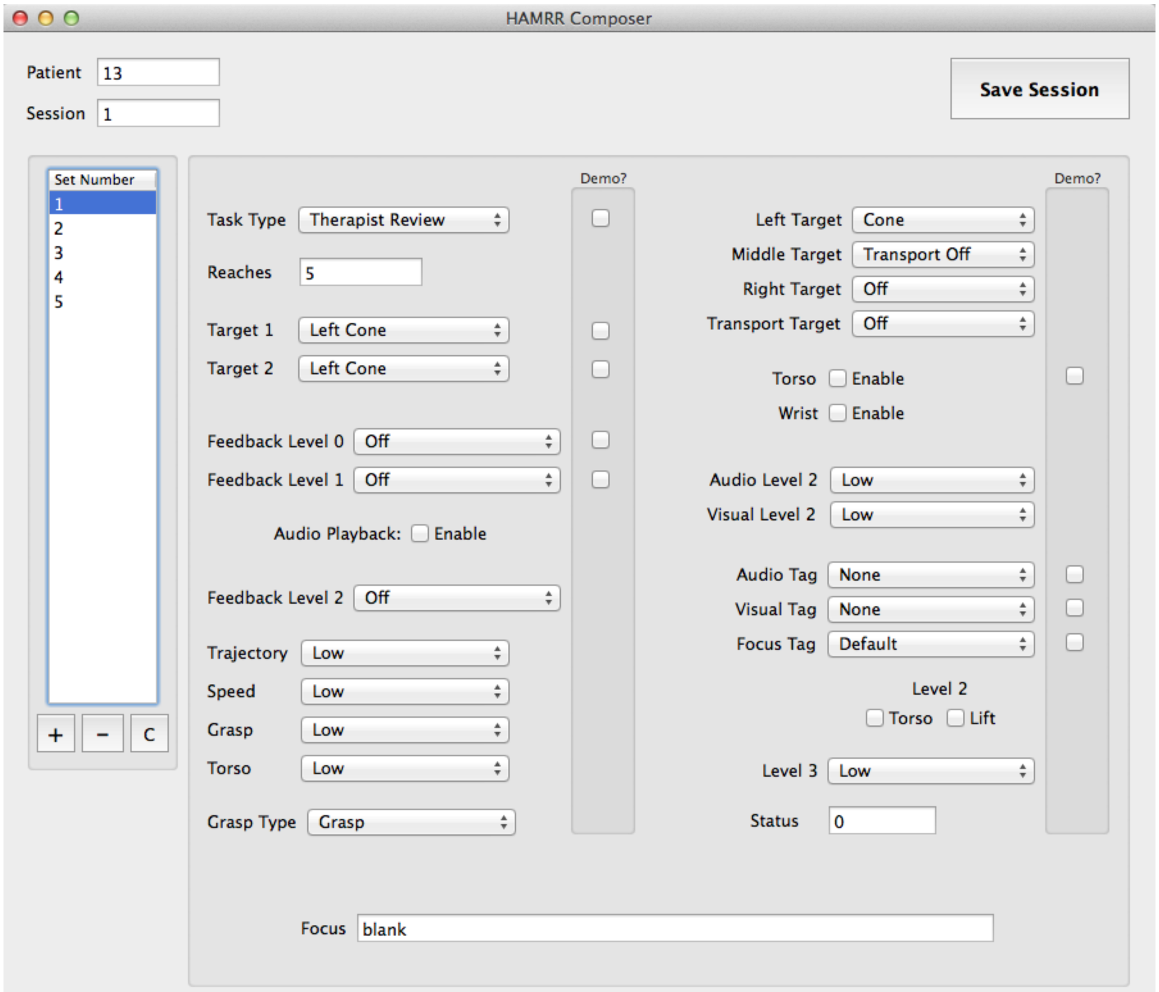
4.2.4.4 Design Tools for All Users of the System

In addition to the core HAMRR software architecture, additional tools were required for the other users of the system. As previously described, INR system are inherently complicated because they have multiple unique users with specific interests and correspondingly different required information. While the previously discussed plugins address primarily the needs of the patient in terms of their use of the system and receiving appropriate feedback based on their performance of a task, the therapist and system design also need to assess how the patient is progressing as well as the effectiveness of the system. Thus, in INR, modular components of the system should be identified to address the needs of both the therapist and system designer, which I identified as a means to compose therapy sessions and analyze recorded patient data.

4.2.4.4.1 HAMRR Composer

Design and Implementation by Michael Baran

As previously discussed, all of the HAMRR scenarios are composed as XML files. The HAMRR composer program was written as a simple interface to compose scenario files. (Figure 20) New sets can be added and undesired sets can be removed. Within each set, feedback streams, target types, and task sensitivities can be changed. Due to the modular nature of the sensitivity hierarchy structure, this program could easily be extended to offer suggestions and pre-compose parts of a scenario file based on certain rules or other inputs.



Patient 13
Session 1

Set Number: 1, 2, 3, 4, 5

Task Type: Therapist Review
Reaches: 5
Target 1: Left Cone ☐
Target 2: Left Cone ☐
Feedback Level 0: Off ☐
Feedback Level 1: Off ☐
Audio Playback: ☐ Enable
Feedback Level 2: Off
Trajectory: Low
Speed: Low
Grasp: Low
Torso: Low
Grasp Type: Grasp

Left Target: Cone
Middle Target: Transport Off
Right Target: Off
Transport Target: Off
Torso ☐ Enable ☐
Wrist ☐ Enable
Audio Level 2: Low
Visual Level 2: Low
Audio Tag: None ☐
Visual Tag: None ☐
Focus Tag: Default ☐
Level 2: ☐ Torso ☐ Lift
Level 3: Low
Status: 0

Focus: blank

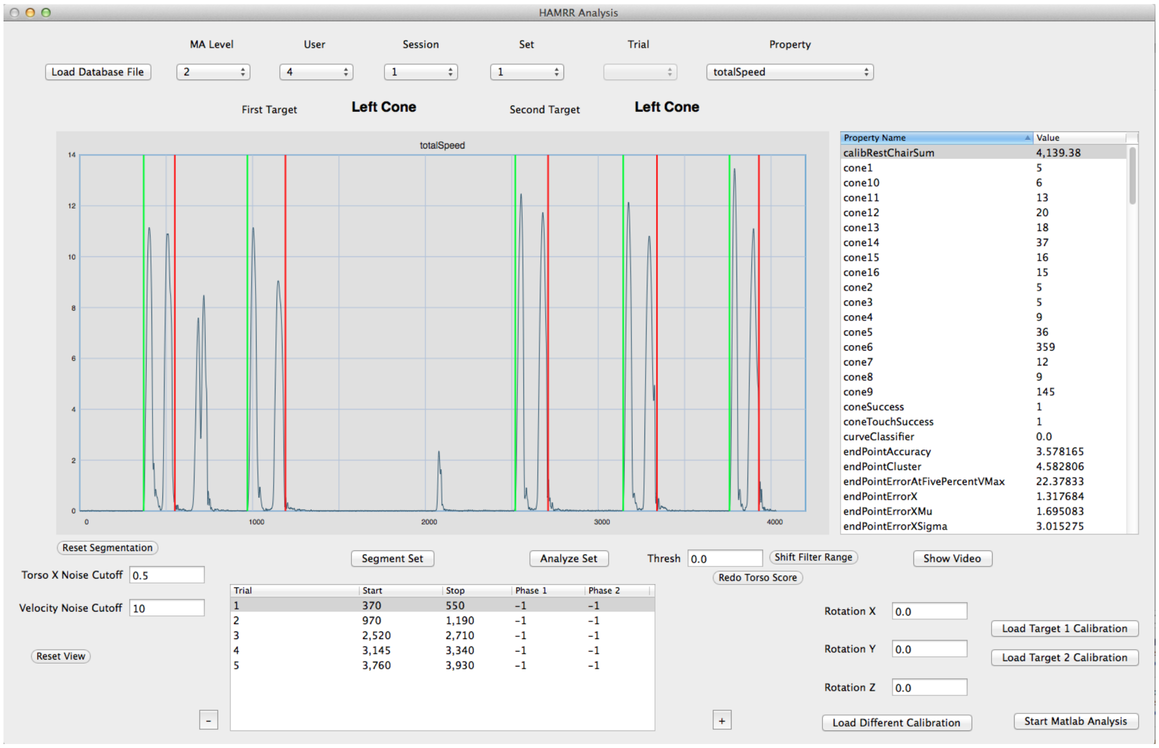
Figure 20 – HAMRR Composer Interface

4.2.4.4.2 HAMRR Analysis Program

Design by Michael Baran

Implementation by Michael Baran, using some functions by Rajaram Singaravelu and Yinpeng Chen

The HAMRR Analysis program was written to open data from the database and perform offline analysis. It allows for viewing of the data as well as filtering noise found in the wrist marker or torso marker data. (Figure 21) Each set of reaches can be segmented and passed through all of the computational analysis of HAMRR. Many of the functions are extensions of code written by Yinpeng Chen, but significant modifications were made to extend the code to HAMRR's data structure, as well as incorporate new calculations that were not completed in the AMRR study.



MA Level: 2
User: 4
Session: 1
Set: 1
Trial:
Property: totalSpeed

First Target **Left Cone**
Second Target **Left Cone**

Property Name	Value
calibRestChairSum	4,139.38
cone1	5
cone10	6
cone11	13
cone12	20
cone13	18
cone14	37
cone15	16
cone16	15
cone2	5
cone3	5
cone4	9
cone5	36
cone6	359
cone7	12
cone8	9
cone9	145
coneSuccess	1
coneTouchSuccess	1
curveClassifier	0.0
endPointAccuracy	3.578165
endPointCluster	4.582806
endPointErrorAtFivePercentVMax	22.37833
endPointErrorX	1.317684
endPointErrorXMu	1.695083
endPointErrorXSigma	3.015275

Torso X Noise Cutoff: 0.5
Velocity Noise Cutoff: 10
Thresh: 0.0

Trial	Start	Stop	Phase 1	Phase 2
1	370	550	-1	-1
2	970	1,190	-1	-1
3	2,520	2,710	-1	-1
4	3,145	3,340	-1	-1
5	3,760	3,930	-1	-1

Rotation X: 0.0
Rotation Y: 0.0
Rotation Z: 0.0

Figure 21 – HAMRR Analysis Interface

4.3 Design of the Evaluation of HAMRR

4.3.1 Patient Evaluation and Related Design Principles

4.3.1.1 Assessments for INR Need to Integrate Functionality and Movement Quality

Previous testing with the KIM only incorporated repetitive movements of reach to touch and reach to grasp in a very marker rich space with constant supervision. However, HAMRR was being designed and built to exist in a very different context, which would be much more variable and possibly noisy, due to reductions in supervision and detailed sensor based data measurements. Therefore, it was hypothesized from the beginning of the HAMRR design process that it would be more difficult to work on detailed lower level features in the home, compared to the hospital.

However, it was still of primary importance to focus on the connection between hospital based training and ADLs, and seeing where the HAMRR system could fit in that ecology. As previously discussed, due to the type of sensing and supervision, HAMRR was designed to place emphasis on higher-level outcomes, such as confidence in using the limb and long-term engagement with therapy. Many of the kinematic measures from AMRR were used again in HAMRR, including those that measured end point and torso information. However, new measures needed to be created for the assessments of groups of reaches and complex tasks that HAMRR introduced and utilized heavily in training sessions.

4.3.1.2 Integrative Assessments Should be Achieved Through Combining Kinematics, Therapist Ratings and Validation by Clinical Measures

Preliminary spatial and temporal category based classifiers were created for kinematic based assessment of groups of tasks as well as complex tasks. The categories were identified based on therapist input on whether the categories were discrete enough and therefore did not significantly overlap in terms of their evaluation meaning. [82] Again, from an iterative design perspective, since the validity and efficacy of these new classifier-based approaches are not known, they were designed with previously discussed motor learning principles in mind and tested among the design team. However, part of the experiment would be to see how successful the classifiers were in assessing patient movement by assessing correlations with clinical results and therapist ratings. If these assessments were found to have strong correlations with established clinical scores and therapist ratings, then they become very valuable tools for automated assessments and protocol adaptation.

In order to collect therapist ratings, Nicole Lehrer developed an iPad app to allow the therapist to record video of the patient using the system, and rate them (according to provided instructions) at their convenience. In addition to recording the video, the therapist would also select a few videos during to create a weekly narrative to review with the patient their progress during the study.

4.3.1.3 Assessments Need to be Associated with ADLs

At this stage in the development of HAMRR, complete focus was given to designing the main system and ensuring that the system was usable by both patients and therapists. Therefore, home based sensing of ADLs was not explored. However the iPad app had the therapist walk through a questionnaire with the patient to ascertain feedback on how the patient perceived any effect on the use of their impaired limb. While this is still not the ultimately objective measure desired, the questionnaire would still provide a basic sense of the patient's perspective on using the system and its possible benefit.

4.3.1.4 Tasks Need to be Designed for Regular, Isolated Assessment of Progress (Both Patient and System)

Using the previously identified principles, it is important to design components of the system that can gather regular assessments for both the patient, therapist, and system designers. Expanding upon the previously discussed design for the scenario protocol paths, a few sessions were also included within each protocol for purely evaluation purposes. These sessions collected both clinical score and kinematic measures.

1. Pre-Test: The pre-test was administered first at the very start of the study
2. Post-Test: The post-test was administered at the completion of the 15-session path
3. One Month Follow Up Test: The one-month follow up test was administered within 1 month of the post-test.

During each of these tests, the following tasks were conducted:

1. Reach to grasp and midline cone (a trained task)
2. Reach to grasp and transport a cylindrical object over a barrier between the midline and far ipsilateral location (a semi-trained task)
3. Reach to grasp and rotated an elevated key at the midline (a non-trained task)

In addition, in order to collect therapist ratings and help correlate these ratings with the data, a special group of assessments-based protocol sessions were created. After every third session of the protocol (excluding the week of training) a training monitoring session was administered before beginning the main training session for the day. During this session, three tasks were administered:

1. Reach to grasp a midline cone (same as the cone tasks provided during the pre, post and one-month follow up
2. Reach to touch a surface elevated at the ipsilateral location

3. Reach to grasp and transport a cylindrical object with and against gravity between an elevated midline location and an on-the-table ipsilateral location.

During these sessions the therapist would use the iPad app to both record videos, as well as review the videos with the patient.

As previously stated, these sessions are not only important for assessment of the patient using the system, but they were also crucial to assess the effectiveness of the therapy protocols. The training monitoring sessions provide regular glimpses into the overall progression of the patient to provide a better sense of how the therapy protocol may be having an effect, which is crucial information in the iterative design process of composing ultimately automated therapy protocols.

4.3.2 Iterative Testing and Application of Design Principles

In the process of developing HAMRR, there were many design iterations that involved feedback from members outside of the direct development team. These were required before the system was to be installed on site and patient recruitment was to begin. Therefore, the following section identifies some principles to consider for the very crucial testing phases of iterative design within INR.

4.3.2.1 Test System Experiences with Unimpaired Subjects

As previously identified, impaired participants can be difficult to recruit on a large scale. However, system testing should not be held off until substantial impaired participants are recruited for a study. Non-impaired subjects should be recruited regularly throughout the design process to test components and provide feedback that the design team may not be able to provide.

As an example, during the design of HAMRR a study with non-impaired subjects was conducted to gauge the feasibility and clarity of the visual and audio feedback environments. As a result, some changes were made to the feedback designs to further strengthen their semantic metaphors.

In addition, this was the first opportunity to test a series of interactive demos that an early prototype of the system facilitated. The initial desire was to have interactive task demos where the system would walk the user through each step of the task. This included instructions on when to start a reach and how to determine if a reach task had been completed. The user was instructed to mimic the task instructions with the system. It was determined during the study that the interactive demos were prohibitively complex and poorly designed. Therefore, narrated videos demonstrating and describing the task were created as a more direct way to convey the task instructions. This was a crucial problem to identify and resolve as it was unknown how instructions should be presented in an unsupervised training environment, where clarity would be of the utmost importance.

4.3.2.2 Include Small Iterative Tests with Impaired Participants

While some more universal aspects of the user experience can be assessed with unimpaired users, stroke survivors will be able to provide input and feedback that cannot be gained elsewhere. Therefore it is very important to include stroke survivors, when possible, in the iterative testing process.

As an example, a short three-session study was conducted late in the development of HAMRR with one of the stroke patients from the AMRR study to go through a set of example sessions in order for the team to observe a stroke patient using the system as well as get feedback on aspects of the system from a stroke patient's perspective, including the perceived difficulty of the tasks. This was an opportunity to make fine tune adjustments to aspects of the system. It also served as a final test of the torso orientation sensing Kinect system when it was realized that it would be too complex for a stroke patient to setup on their own.

4.3.2.3 Search for Feedback from Domain Experts, Even Outside of Interactive System Testing

In addition to these more formal, yet short, studies, interviews are important ways of gaining feedback on aspects of the system design. These can be especially helpful early on in the design process as a physical, working prototype may not be available for interactive testing. During the HAMRR development process, feedback was gained from therapists, clinicians and patients in the identification of proper computation algorithms (such as kinematic categories for qualitative analysis) in addition to identification of designs for components of the overall experience (such as how demo instructions are presented and how the scenario paths should be composed).

4.3.3 Pilot Study Phase I – Iterations Within Pilot Study

At the end of the development cycle, the system was installed in a clinical space in both the Rehabilitation Institute of Chicago and Emory University for testing with six patients (three at RIC and three and Emory) each across 15 training sessions. The goals were to see how multiple stroke patients engaged with the system, both in terms of usability and functional gains. The system was also monitored for any stability issues. The results of this study will be presented in a later section.

While the importance of regular feedback from multiple sources was previously stressed during the main design phase of INR system design, iterative design should not be overlooked during the pilot study. While iterations at this level will need to be more reserved as the study will be designed to determine the efficacy of different aspects or variables of the system experience, they should not be ignored either. Observations of system use and initial review of evaluation results are crucial during the pilot study to identify areas that need iteration.

After six subjects completed the protocol, initial review of the data (to be presented later) showed very inconsistent to minimal overall change in kinematic measures. At the time, the main question about these results was how much the recruited patient population's impairment level may have played in the results. The six subjects that participated in the study were of mild impairment, and it was questioned if these subjects may have a floor effect, in which the system could not address their levels of impairment. This initial assessment of results was also supported by observations of the subjects using the system and seeing that the majority of the subjects were able to complete the tasks with little difficulty or perceived challenge.

4.3.3.1 Include Small Iterative Tests with Impaired Participants

Just as previously described, the value of smaller tests within a larger iterative design cycle is once again crucial here. The iterative tests can serve the same purpose of identifying strengths and weaknesses in the new design before full integrated testing with the larger system experience.

Before beginning any redesign of HAMMR, the team needed to understand the current state of the system under the use of a more impaired participant. Therefore as a simple test, a one-day protocol was created to test two more severely impaired patients to see how they would react to the feedback environments and the tasks. This one-day protocol represented a condensation of the three sessions used at the beginning of the main study protocol. What was discovered is that the more impaired patients could not successfully open their hands wide enough to grasp the transportable cylinder object. If a patient could not complete tasks with this object, it would effectively eliminate a significant portion of HAMRR's protocol as well as eliminate the primary goal of testing HAMRR in the context of complex tasks and higher-level evaluations and feedback.

4.3.3.2 Use Iterative Testing with a Modular Architecture to Quickly Integrate a New Solution

As previously shown, small tests can identify key areas for redesign. Once those key areas have been identified, a solution (especially during the pilot study) needs to be identified as quickly as possible. The ability to quickly integrate new solutions further emphasizes the value of a modular architecture that is designed with flexibility and extensibility in mind.

The solution to the identified problem of the transportable object was to create a smaller object that could be grasped with a smaller hand aperture. The new object was able to be implemented quickly as the new design used parts from the cone object and transportable object, and the Tangible Plugin was designed in such a way that adding a new object to the code was a simple class extension.

However, iterative testing is still crucial, as it was found in testing in the lab that a new wireless data setup utilized was impeding smooth interaction with the object during full interactive system testing, and thus a compromise needed to be made to remove any data output from the new object. Given that this redesign was a temporary solution, this compromise was viewed as acceptable for the scope of the pilot study. For more details on the implementation of this new object, please refer to Appendix B1.

4.3.3.3 Include All Users of the System in Iterative Design

As previously discussed, there are multiple users of an INR system, and when looking at observations of the system as well as evaluation data, it is equally important to assess where iterations in the design are required for the other users of the system outside of the patient.

In order to ultimately test the system with more impaired participants, it was required that a physical therapist would have to run the system, not a system designer as was the case for the first six participants. From observations of the system in use, it became very clear that the interface was not easy to use for someone who had no previous knowledge of the system or how it worked on a technical level. Therefore, it was crucial to identify which aspects of the main GUI (in the Adaptation Plugin) could be redesigned in order to help the therapist run the sessions. Again, during a pilot study there is not adequate time for a full redesign, so prioritized aspects of the GUI to adjust had to be identified. This resulted in adding two new visual indicators on the main window: one to identify which targets needed calibration and one to describe the state of the incoming data (both marker and tangible object). For more details on the implementation of the new GUI components, please refer to Appendix B2.

CHAPTER 5

EVALUATION OF HAMRR

In order to test the efficacy of the previously identified design constraints and the resulting system, HAMRR was tested with eight patients. The initial six patients were split across two sites at Rehabilitation Institute of Chicago and Emory University. These sessions were completed under minimal supervision by Nicole Lehrer and myself. The final two subjects were recruited at Emory exclusively and the sessions were conducted in person by a physical therapist on site with Nicole Lehrer and myself observing remotely via Skype. All subjects were able to complete the protocol, but the final two subjects required more assistance than the initial six. Section 5.1 presents results of the patient assessments. Section 5.2 reviews system stability and usability results. Section 5.3 summarizes the larger conclusions that the results present.

5.1 Patient Assessments

5.1.1 Patient Demographics

Eight total subjects were recruited for the study. The first stage of the study was comprised of six subjects, three at Emory and three at RIC. After the completion of this phase of the study, two additional subjects were recruited with more moderate impairment (in contrast to the mostly mild subjects recruited with the first phase of the study. Each subject's demographics can be seen in Table 2.

Table 2. Patient Demographics

Subject	Age	Months Post Stroke
1	63	14
2	69	44
3	65	31
4	47	26
5	56	28
6	49	18
7	50	15
8	44	13

All subjects had a right-sided hemiparesis stroke (and were right hand dominant pre-stroke) between six months to five years before the study. Prior to entrance into the study, each subject was screened along the Montreal Cognitive Assessment[83] (score >25), Geriatric Depression Scale[84] (score <10), range of motion assessments (Shoulder: at least 45 degrees flexion, abduction, and rotation; Elbow: ROM at least 90 degrees of flexion/extension; Forearm: at least 20 degrees of pronation or supination from neutral position; Wrist: at least 20 degrees extensions at any point in range with forearm pronated and supported and wrist in full flexion; Fingers: at least 10 degrees active extension of metacarpophalangeal and interphalangeal joint of the thumb and any two fingers) a sensory perception test (testing for color, shape, pitch and timbre perception) and the seated portion of the Upper Extremity Fugl Meyer Assessment (FMA)[85] (score between 30 and 56).

5.1.2 Clinical Assessments

Before (the Pre Test), immediately after (the Post Test) and one month after (the One Month Follow Up) the protocol, a series of clinical measurements were assessed. The Wolf Motor Function Test was used to assess movement at the shoulder, elbow and hand. The sitting portion of the Upper Extremity Fugl Meyer Assessment was used to assess reflexes, sensation, pain and motor function of the affected upper extremity. The Quality of Movement section of the Motor Activity Log was used to assess how well the subject was using the affected upper extremity during the activities of daily living. The measurements of each of these features can be found in Table 3. A two-tailed T-Test, with a confidence interval of 95% was utilized to assess group changes in clinical scores between Pre and Post as well as Pre and One Month Follow Up. The results of these tests can also be seen in Table 3. Subject 6 was unable to attend the one-month follow up session, and thus does not have data from this time point.

Table 3. Pre-Post Clinical Score Comparisons

Subject	Pre FMA (/66)	Post FMA (/66)	1 mo. FMA (/66)	Pre WMFT FA (/5)	Post WMFT FA (/5)	1 mo. WMFT FA (/5)	Pre WMFT TT (s)	Post WMFT TT (s)	1 mo. WMFT TT (s)	Pre MAL (/5)	Post MAL (/5)	1 mo. MAL (/5)
1	37	42	44	3.73	3.93	3.93	47.40	42.86	43.08	3.64	3.07	3.32
2	50	58	57	3.67	3.87	3.80	59.10	47.10	44.08	2.44	3.02	2.96
3	47	56	62	3.93	4.47	4.73	41.34	37.51	36.86	4.38	4.71	4.41
4	47	52	55	4.60	4.80	4.87	26.61	26.98	23.79	3.88	3.63	4.04
5	44	50	46	4.33	4.87	4.20	32.26	31.66	32.54	2.48	2.96	3.15
6	37	43	NA	3.50	3.40	NA	246.87	117.23	NA	2.96	3.50	NA
7	30	34	32	2.73	2.93	3.07	692.75	590.76	576.82	1.18	2.43	2.73
8	29	32	29	2.73	2.87	2.80	986.46	917.15	973.74	0.98	1.13	1.13
tValue		8.205	3.050		3.215	2.191		-2.167	-1.404		1.600	1.721
pValue		*0.0000	*0.023		*0.015	t0.071		t0.067	0.210		t0.154	t0.135

(*)Significant scores (p<0.05); (t)trending scores (p<0.2)

The Fugl Meyer Assessment and Functional Ability portion of the Wolf Motor Test both improved (when comparing the pre test to the post test) at the group level with p values less than 0.05. The group level improvement for the Time portion of the Wolf Motor Functional Ability Test (between the pre and post test) was not significant, but found to be trending with a p value less than 0.1. The change in the Motor Activity Log was not found to be significant, but slightly trending with a p value less than 0.2

Overall, the comparison of the clinical scores between the pre test and the one-month follow up do not show the same significant changes. The change in Fugl Meyer Assessment scores was found to still be significant with a p value of 0.023. The difference in the Functional Ability Test of the Wolf Motor Function Test was found to be trending with a p value of 0.071. The change in the Total Time portion of the Wolf Motor Function Test was not found to be significant or trending. The

Motor Activity Log showed similar trending changes (p value less than 0.2) as the pre versus post-test comparison.

5.1.3 Kinematic Assessments

In addition to the clinical assessments, a series of kinematic measurements were made. All of these measurements were primarily from the analysis of the wrist and torso markers, with some minimal help from the chair pressure sensors. The chair sensors, as will be described in more detail later, were somewhat helpful in identifying when participants were starting a reach with their back fully supported by the chair (a basic indication of minimal to no compensation at the start of the reach). These kinematic measurements were made while the subject performed a series of tasks during the Pre, Post and One Month Follow Up, as well as during all of the Training Monitoring sessions. The design and selection of tasks was chosen to represent a distribution of system trained and untrained tasks.

5.1.3.1 Task Descriptions

Task 1 – Key Turn / Untrained / Pre, Post and One Month Follow Up

For this task, the Wolf Motor Test key turn object was placed at the subject's midline, but rotated 90 degrees such that the key resembled the motion required to start a car. The subject was instructed to reach to grasp with key (with a pincer grasp) and rotate the key away from their body with the furthest rotation possible (the key would start perpendicular to the plane of the table and end up parallel to the table). The subject performed three of these reaches, with the therapist resetting the key after each reach.

Task 2 – Transport Over a Barrier / Semi-Trained / Pre, Post and One Month Follow Up

For this task, a barrier (a paper towel roll) was placed at the middle table target location and the patient was instructed to transport the cylindrical transport object from the midline location to the far ipsilateral location, moving the object over the barrier. The subject was instructed to perform five round trips (the object was moved in each direction five times). For analysis purposes these round trips were divided into two groups based on the direction of the movement: Forward (Midline to Far Ipsilateral) and Reverse (Far Ipsilateral to Midline)

Task 3 – Transport Against Gravity / Trained / Training Monitoring

For this task, the subject was instructed to transport the cylindrical object between an off the table platform at the midline table location to an on the table platform at the ipsilateral table location, resting at the rest pad in between sequential reaches. The subject was allowed to complete the tasks as many times as needed, but the task was stopped if an allotted amount of time was taken or the subject knocked or dropped the cylinder to a location that the subject was not able to reach without assistance. The subject was instructed to complete two complete transports (round trips). For analysis purposes these round trips were divided into two groups based on the direction of the movement: Up (Ipsilateral on the table to Midline off of the table) and Down (Midline off of the table to Ipsilateral on the table).

Task 4 – Reach to Grasp a Cone / Trained / Pre, Training Monitoring, Post, One Month Follow Up

For this task, a cone was setup at the midline target location. The subject was instructed to reach and grasp the cone 10 times. This task was performed at the beginning of every training session.

Task 5 – Reach to Touch and Elevated Surface / Semi-Trained / Training Monitoring

For this task, a flat surface (resembling a button) was affixed to the top of an off of the table target receptacle, which was placed at the ipsilateral target location. The touch required was similar to that of the training button object, but the surface was elevated to a height similar to the cylinder's height when using the off of the table target platform. The subject was instructed to reach to touch the button surface with any fingers, excluding the thumb. This task was completed five times.

5.1.3.2 Feature Descriptions

Across these features, a series of kinematic measurements were made by the system as well as some made by Nicole Lehrer and myself from patient video observations (for features that could not be assessed through the raw data alone). What follows is a description of these features (Shorthand numerical labels are provided for brevity in subsequent data tables). Table 4 shows the tasks from which these features were evaluated.

Table 4. Feature Analysis Summary

	Key	**Transport Over a Barrier**	**Transport Against/With Gravity**	**Cone**	**Reach to Touch**
F1	X				
F2	X				
F3	X				
F4	X			X	
F5	X	X	X		
F6	X	X	X	X	
F7	X				
F8	X	X	X	X	
F9		X			
F10		X	X		
F11			X		
F12				X	
F13				X	
F14				X	
F15				X	
F16				X	
F17				X	

An “X” represents that the task was analyzed for that feature. Grey features were removed from analysis for this discussion.

Key Completion (F1)

This feature was a qualitative evaluation made by myself and Nicole Lehrer when reviewing the videos of each subject completing the key turn task in order to determine if the key turn task was completed. This measure was necessary as the key object was not instrumented with any sensors and thus the grasp and complete rotation of the key could not be detected through the raw data alone.

Time Until Grasping the Key (F2)

This feature measured the total time from the moment the subject's wrist began to move from the starting position to the moment the subject grasped the key. Because of the lack of sensing on the key, the measurement was determined visually by watching a recorded video of the subject completing the task. The camera frame rate was used to estimate the value of this time measurement.

Time Until Releasing the Key (F3)

This feature measured the total time from the moment the subject's wrist left the rest pad to the moment that the subject released their hand from a grasp of the key. As with feature F2, due to a lack of sensing on the key itself, the timing information was determined visually by watching a recorded video of the subject completing the task.

Time Until 5% Velocity Max (F4)

This feature was predominantly used as the main measure of the time required to reach for an object. This feature was also used in the AMRR system study. The feature measures the total time from the start of the reach (the subject's wrist leaving the rest pad) to the moment at which the total velocity reaches 5% of the peak velocity achieved during the reach. This measurement was determined by identifying the motion capture frames in which the reach started and concluded (with 5% of the maximum velocity) and converting the difference into time given the fairly constant frame rate of the motion capture cameras.

Path Ratio (F5)

As has been previously discussed, studying the dynamics of multiphase movement can be very complex as even non-impaired subjects will perform the tasks with variability. Therefore, path ratio was used as a way to compare trajectory efficiency during a reach that was part of a complex multi-phase transport task. This measure integrated the total speed of the reach over the entire reach in order to determine the total distance of the path taken by the wrist marker during the task. This total path distance was then compared to the straight-line distance (determined using the calibrated wrist marker location recorded for each location) as a ratio. As the ratio nears one, the total path taken more resembles the straight-line shortest distance path between targets.

Number of Speed Phases until 5% Velocity Max (F6)

The feature was also used in AMRR. It is a measurement of the smoothness of the velocity profile that counts the number of local minima found in the velocity profile between the location of the maximum velocity and the end of the reach. If a person were to reach in one smooth movement, then the number of additional phases would be zero.

Number of Speed Phases until Key Grasp (F7)

The determination of the number of extra speed phases for the key turn task was determined slightly differently than for other tasks. The core idea is the same in counting the number of local minima in the speed profile between the maximum of the velocity peak and the end of the reach. However, in this case the end of the reach was determined visually by watching the video of the participant completing the task. This was done because the key object was not instrumented, and therefore the tangible grasp of the key could not be detected from the data alone. Therefore, the point of grasp was located in the video, and this point was converted to a camera frame number to match with the motion capture data. This conversion could be done because the start of the data archiving and the start of the video recording were the same. However, some issues were encountered with this feature since dropping frames of data would lead to a mismatch between the video time and the data frame. More will be discussed on this issue later.

Torso Movement Sum (F8)

This measure provides a basic evaluation of compensation during a reach by summing the distance traveled (the distance between the location of the centroid of the rigid body from the centroid location of the rigid body at the start of the reach) for each frame of the reach. This measure was selected as a more stable alternative to angle orientation measurements, which had some noise problems (which will be discussed later).

In order to filter some of the results, cases in which the subject was already compensating at the beginning of the reach needed to be removed from analysis as they would not provide an accurate measure of the compensation. Therefore, the pressure sensitive chair data information was used. During each rest calibration, in addition to the location of the wrist marker, the applied pressure to each chair sensor is recorded. During each rest calibration the subject is instructed to, as best as possible, sit up straight with their back against the chair. It was presumed that most forms of compensation would require leaning away from the chair, and therefore reduce the total amount of pressure applied to the chair. In order to determine if the subject was not compensating at the start of the reach, it was ensured that the starting chair pressure was at least 50% of the calibrated rest chair pressure. In addition, the starting chair pressure total had to be greater than 1000 (if no pressure is applied to the chair, the total would be 552). These thresholds were determined from analyzing the movements of non-impaired subjects using the system. If the subject's chair data passed both of these thresholds, then the torso movement was analyzed, otherwise the analysis was not performed on that reach.

Transport Over Barrier Completion Rating (F9)

Similar to F1, this feature as a rating determined by Nicole Lehrer and myself upon watching each subject complete the transport over barrier task. While the transportable cylinder had built in sensors, the grasp analysis was only binary and wouldn't provide information on how well the task was completed.

Total Time Until Completion (F10)

This time feature was only used for the transportation tasks. This feature measures the time from the start of the reach to the deposit of the object. In order to determine where the object deposit occurred some additional reach segmentation was conducted. For each reach, the velocity profile was analyzed to find the main phase of the reach towards the object as well as the main phase of moving the object to its second location. From this second overall phase, the location of the "5% Velocity Max" was found and marked as the end of the reach. Therefore, the total time until completion measures the time from the start of the movement until 5% of the velocity max of the second overall phase.

Transport Against and With Gravity Completion Rating (F11)

This feature is similar to F1 and F9, except it is applied to the training monitoring task in which the subject transports an object between an off the table and on the table location.

Cone Grasp Completion Rating (F12)

This feature is similar to F1, F9, and F11 except it is applied to the task of the subject reaching to grasp a cone.

Maximum Horizontal Trajectory Error (F13)

For each single object task (as well as the object approach portion of a transportation task), a reference trajectory was collected. This reference trajectory represented the average trajectory path of the wrist marker if a non-impaired subject were to complete the task. These references were collected from a group of non-impaired, age-matched subjects. This feature finds the maximum horizontal deviation from the reference that the subject had during a reach, as a measure of overall spatial efficiency of the reach. This feature originated from AMRR. For the purposes of analysis, this feature was only used with single object tasks, where spatial efficiency was evaluated in complex tasks through path ratio.

Maximum Vertical Trajectory Error (F14)

This feature is very similar to F13, except this feature measures the maximum vertical error during a reach.

Peak Speed (F15)

This feature measured the maximum speed of the velocity profile created during a reach. This feature is important to analyze as different attributes of patient impairment can come across in the speed, such as non-controlled, ballistic movement (seen with very high peak speeds) or difficulty completing the task (seen with lower peak speeds). Similar to the trajectory error features, for the purposes of analysis, this feature was only used with single object tasks. For complex tasks, total completion was used as a higher-level measure of the speed of the task. This feature was also originally part of the AMRR system.

Bellness Normalized Area (F16)

This feature was also taken from the AMRR system. The feature measures the "bellness" of the velocity profile, which provides an indication of the overall smoothness of the reach. This feature is coupled with the number of additional phases feature. While the phase number feature provides a measure of the number of additional phases in the reach, it does not provide any information on the magnitude of the additional phases. This feature is a ratio of the distance (integral of the velocity profile) from the end of the first phase to the end of the reach to the distance from the peak speed to the end of the reach. This value will range from 0 to 1, with zero representing a bell curve without additional phases.

Jerkiness (F17)

This feature, also used in AMRR, determines the jerk cost of the reach, which is a further measure of the smoothness. With this feature one concerning issue was found.

Upon analysis of the data, it was found that the result of the jerkiness analysis (the same methodology and analysis as the AMRR system) was very sensitive to peak speed. During the pre test, Subject 4 attempted to complete the cone reaches as fast as possible. While very fast, these subject's reaches were smooth and completed in one phase. However the jerkiness results for this task for this subject were the highest seen in the entire study. Therefore in the future, a new jerkiness measure may be required, but for the scope of this study, Subject 4's jerkiness (and peak speed) data was removed from analysis.

5.1.3.3 Task and Feature Selection for Analysis

From the entire set of previously described tasks and kinematic features, a subset of tasks and features were selected for analysis. Overall, since the AMRR system had already verified many of the mixed reality approaches to single object tasks, the focus of the HAMRR analysis was to look at higher level features of the complex tasks and see how these related to clinical score progression. The cone tasks were also kept for analysis as this task was used throughout AMRR and would provide a general understanding of progression in low-level kinematics.

Removal of Key Turn Task

The key turn task turned out to provide too much noisy data for reasonable analysis. Initially, the desire was to measure the amount of wrist rotation during the key turn task. Therefore, a small four marker rigid body was used in place of the single wrist marker, so that angle orientation information could be extracted. However, the use of this second rigid body (coupled with multiple sources of occlusion of the torso marker, the key turn box obstructing the wrist markers, and the wrist rotation sometimes rotating too many markers away from view of the cameras) led to markers being dropped and swapped. This resulted in all of Subject 1 and Subject 8's key turn task data being unusable, and numerous unusable reaches from the other subjects as well. While this did not impact the kinematic analysis that was done by observing the video, it was difficult to connect these results to the raw data due to gaps of data missing during marker occlusion. Therefore, it was noted that a new sensing solution is needed for this task, but for the scope of this study, the all data (motion capture and video-based) would be removed from analysis.

Removal of Task Completion Ratings

As previously discussed, Nicole Lehrer and I completed a series of task completion ratings for the tasks in order to determine if each subject completed the task. However, it was noticed that this feature did not provide much information as most subjects completed the tasks. In addition, trained therapists did not complete these ratings, and thus the value of these ratings is questionable.

Removal of Reach To Touch Task

The reach to touch task was designed as a semi-trained task that could be included in the training monitoring sessions. It represented a hybrid of tasks that the subject saw in training. However, in practice, the task was not as helpful in assessing progress. The main issue with this task was the distance between the rest pad and the target surface (the effective distance the subject would reach over) was very small. For subjects with larger hands, the wrist marker moved a very small distance during the reach. Therefore, these short distances provided predominantly unusable data as every reach looked very efficient (since the amount of data collected during a reach was so small).

5.1.3.4 Kinematic Analysis Results

The first analysis completed to look at group level changes in kinematics was to run a paired p-test for each task. Tables 5-7 present the p values for group level changes in each feature of the task comparing between the pre and post test as well as the pre and one month follow up test.

Table 5. Group Level Kinematic Changes for Transport Over a Barrier Task Forward (Moving Midline to Far Ipsilateral)

	F5	**F6**	**F8**	**F9**	**F10**
Pre vs Post	0.5327432	0.3903811	0.1887622	0.16082208	0.3526443
Pre vs 1 Mo.	0.3142627	0.5037976	0.2920075	0.07558682	0.6022861

Table 6. Group Level Kinematic Changes for Transport Over a Barrier Task Reverse (Moving Far Ipsilateral to Midline)

	F5	**F6**	**F8**	**F9**	**F10**
Pre vs Post	0.7561348	0.6215233	0.4932472	0.08527752	0.9756226
Pre vs 1 Mo.	0.04789909	0.117275	0.227445	0.36321747	0.3157943

Table 7. Group Level Kinematic Changes for Cone Task

	F4	**F6**	**F8**	**F13**	**F14**	**F15**	**F16**	**F17**
Pre vs Post	0.823903	0.781255	0.140980	0.228067	0.145943	0.183444	0.508163	0.0877553
Pre vs 1 Mo.	0.210963	0.325946	0.323446	0.708468	0.182413	0.231653	0.506570	0.2571927

As can be seen in the table, there were no significant changes found in the Transport Over a Barrier task or Cone Grasp task. This is true for both the pre/post comparisons and pre/one-month-follow up comparisons. However, some features were found to have trending changes. The torso movement sum was found to have a trending decrease with a p value less than 0.2. Similarly, vertical trajectory error was found to have a trending decrease (both between pre and post as well as pre and one month follow up). Jerkiness was also found to have a decreasing trend (with a p value less than 0.1), however, this result may be skewed by the performance of Subject 4, who completed the pre-test very quickly and thus had a very high jerkiness value.

Floor Effect with Subjects 3, 4 and 5

One of the possible reasons why the kinematics did not show significant group changes was because a subset of the subjects had good functionality at the beginning of the study, and were not provided enough of a challenge by the system and therefore a floor effect was found in the their kinematic data.

Subjects 3, 4 and 5's pre test clinical scores show that each entered the study with good functionality (Their Wolf Motor Function Test Functional Ability scores were above about 4 and their Fugl Meyer Assessment scores were greater than 44). Each subject's good functional ability at the start of the study is also reflected in their kinematics when comparing their measurements with non-impaired subjects.

As seen in Table 8, the majority of the pre test measurements for Subjects 3, 4 and 5 are within one standard deviation of the average value recorded when non-impaired subjects completed the task. However, when looking at Table 9 it can be seen that the other subjects, while having some overlap with the non-impaired population, have more features that begin outside of the range of non-impaired measurements, and thus, have more room from improvement in kinematic values.

Table 8. Pre Test Values for Subjects 3, 4 and 5

	Control				
	Avg – Std Dev	**Avg + Std Dev**	**Subject 3 Pre**	**Subject 4 Pre**	**Subject 5 Pre**
Cone					
F4 (s)	0.8021	1.2075	1.105	0.616*	0.892
F6	0	1	0.375	0.2	0
F8 (mm)	0	1158.4763	2401.575	719.542	2545.99
F13 (mm)	0	24.1236	16.60175	13.22988	16.01672
F14 (mm)	0	30.3495	26.939925	30.4144	26.0203
F15 (m/s)	0.5693	0.8503	0.783055	1.24422*	0.923484**
F16	0	0.0718	0.054	0.0077046	0
F17 (mm^2/f^5)	0	0.0367	0.02339013	0.0818959*	0.0236854
Trans. Barr. Fwd.					
F5	1	1.468641	1.3608	1.376	1.2789
F6	0	1	0.2	0	0
F10 (s)	1.63562	2.43178	3.248	2.322	2.426
F8 (mm)	0	547.65	4091.5	984.63	2815.7
Trans. Barr. Rev.					
F5	1	1.401036	1.3359	1.415	1.373
F6	0	1	1.2	0	0.2
F10 (s)	1.81915	2.68125	2.752	1.794**	2.438
F8 (mm)	0	1200.04	3146.9	540.33	2826.8

Grey cells mark features that were measured within one standard deviation of non-impaired averages for that feature. (*) Subject 4 attempted to complete the task as possible. (**) While this feature is less than the non-impaired range, the therapists did not view reaching too fast as a detriment to the performance of the task.

Table 9. Pre Test Values for Subjects 1, 2, 6, 7 and 8

	Control						
	Avg – Std Dev	**Avg + Std Dev**	**Subject 1 Pre**	**Subject 2 Pre**	**Subject 6 Pre**	**Subject 7 Pre**	**Subject 8 Pre**
Cone							
F4 (s)	0.8021	1.2075	1.462	1.4255	2.146	2.408	3.6725
F6	0	1	1.1	1.1111	4.5	4.9	6.5
F8 (mm)	0	1158.4763	10668.84	3965.21	14303.41	47592.44	NA
F13 (mm)	0	24.1236	20.7201	22.63711	35.7324	43.8571	45.0115
F14 (mm)	0	30.3495	43.1093	29.2522	50.7228	108.206	183.2075
F15 (m/s)	0.5693	0.8503	0.631827	0.6417711	0.691672	0.662963	0.470585
F16	0	0.0718	0.134352	0.12073067	0.325525	0.439106	0.4146125
F17 (mm^2/f^5)	0	0.0367	0.0125567	0.01646311	0.0606767	0.0582684	0.0221745
Trans. Barr. Fwd.							
F5	1	1.468641	1.4418	1.4854	1.5835	1.5847	2.3195
F6	0	1	0.6	2.6	1.3333	6.8	8.75
F10 (s)	1.63562	2.43178	4	5.082	4.2067	9.202	10.373
F8 (mm)	0	547.65	14466	4915.8	6744.1	73671	NA
Trans. Barr. Rev.							
F5	1	1.401036	1.5685	1.4229	1.6243	1.8286	2.0456
F6	0	1	1.8	0.4	2.3333	8.2	6.75
F10 (s)	1.81915	2.68125	3.854	4.334	4.34	8.85	6.7625
F8 (mm)	0	1200.04	10345	2016.2	9513.2	66296	NA

Grey cells mark features that were measured within one standard deviation of non-impaired averages for that feature.

Removal of Subject 8 Data

Part of the second phase of testing of HAMRR involved the recruitment of two more impaired subjects with the previously discussed goal to see how the system could address their needs. However in the case of subject 8, the system was not able to accurately capture the performance data. This occurred mainly because of marker occlusion issues that would appear during excessive body compensation and would result in the loss of data frames and thus noisy data. In addition, the subject had difficulty in completing the transportation tasks, which resulted in a subset of the total data expected for the task. These two factors combined result in a shortage of data to look at for this subject, which suggests the need for better sensing solutions for this level of impairment as well as more adaptable, easier to complete transportation tasks. Therefore, for the purposes of the remaining analysis, Subject 8's data was removed from the discussion to focus on the remaining subjects.

Review of Subjects 1, 2, 6 and 7

As previously discussed, Subjects 3, 4 and 5 showed good functionality in their clinical scores and starting kinematic measurements that were within non-impaired variation. However, the starting kinematic measurements of the other subjects (excluding 8 as previously discussed) indicated possible room from improvement.

Therefore, a kinematics review was conducted for each of these subjects. Since looking at these subjects with the time samples collected would provide too small of a sample size to perform group level analysis with this specific subset of subjects, a more general analysis approach was taken to assess if any individual changes were seen in kinematic features.

Individual Subject Kinematics Analysis Approach

By looking at the kinematics on an individual level, the ability to run statistics to determine the significance of change between pre and post or the one-month follow up is eliminated because of the small sample size. Therefore the goal of this analysis was to identify any interesting trends in the data that might provide insight for future analysis and system design.

The first step was to identify a subset of the data in which the change in feature values between pre and post was worth further review. The approach was to look at the changes in the average value of a feature between pre and post and see if the average of the post value was beyond the bounds of the pre average value including its positive and negative standard deviation range. The features that displayed this type of change were selected for further analysis. Tables 10-13 show the percent change between pre and post for the features that had a decrease or increase beyond the variance of the pre test. In selecting the features to analyze both features that improved and worsened were considered.

The second step, once the previously described subset of data was identified, was to look at the standard deviation of the pre and post data. As an additional criterion for identifying interesting change between pre and post was for the standard deviation of the post to be less than that of the pre. If the standard deviation were to decrease from pre to post, this would indicate a more consistent change in the kinematic measure. With these two criteria combined, not only would the change represent movement beyond the variance of the pre test, but the posttest values would also be more consistent than the pre. Tables 10-13 display the features that decrease in variance between pre and post.

Table 10. Subject 1 Kinematic Results

			Subject 1		
	Pre/Post Change (%)	**Improve?**	**Decrease in Variance?**	**Training Monitoring Trend**	**Linear Model ($p<0.05$*)**
Cone					
F4	2.56				
F6	54.55	-			
F8	-45.23	++	Yes	Trans. Up	Linear*
				Cone	Quadratic*
F13	30.20	--		Cone	Linear
F14	24.73	+	Yes		
F15	6.30	-	Yes		
F16	8.80	+	Yes		
F17	23.15	-	Yes		
Trans. Barr. Fwd.					
F5	1.6604				
F6					
F10	-14.65	++		Trans. Down	Quadratic*
F8	-54.33	++	Yes	Trans. Down	Quadratic*
				Trans. Up	Linear*
				Cone	Linear*
Trans. Barr. Rev.					
F5	-1.39		Yes		
F6	-44.44	+	Yes		
F10	-21.07	++		Trans. Down	Cubic*
F8	-72.03	++	Yes	Trans. Down	Linear*
				Trans. Up	Quadratic
				Cone	Quadratic*

(++) Improve with post values outside of variance of pre test. (+) Improve, but post values are not outside of pre test variance. (-) Worsens, but not beyond variance of pre-test. (--) Worsens with post values outside of variance of pre test.

Table 11. Subject 2 Kinematic Results

	Subject 2				
	Pre/Post Change (%)	Improve?	Decrease in Variance?	Training Monitoring Trend	Linear Model (p<0.05*)
Cone					
F4	-10.84	+	Yes		
F6	-46	+	Yes		
F8	-53.97	++	Yes	Trans. Down Cone	Linear* Linear*
F13	-22.31	+	Yes		
F14	19.4425	-	Yes		
F15	-1.26		Yes		
F16	-55.34	+	Yes		
F17	-14.02	+	Yes		
Trans Barr Fwd					
F5	-8.43	++	Yes	Trans. Down	
F6	-53.85	++		Trans. Down Cone	Linear Linear*
F10	-25.7	++		Trans. Down Trans. Up	Linear* Linear*
F8	-41.05		Yes	Trans. Down Cone	Linear* Linear*
Trans Barr Rev					
F5	-3.81	++	Yes	Trans. Up	
F6	(0.4 to 0)	+	Yes		
F10	-14.58	++	Yes	Trans. Down Trans. Up	Quadratic* Linear*
F8	-33.01	++	Yes	Trans. Down Trans. Up Cone	Quadratic* Linear* Quadratic*

(++) Improve with post values outside of variance of pre test. (+) Improve, but post values are not outside of pre test variance. (-) Worsens, but not beyond variance of pre-test. (--) Worsens with post values outside of variance of pre test.

Table 12. Subject 6 Kinematic Results

	Subject 6				
	Pre/Post Change (%)	**Improve?**	**Decrease in Variance?**	**Training Monitoring Trend**	**Linear Model (p<0.05*)**
Cone					
F4	-52.84	++	Yes	Cone	Quadratic*
F6	-80	++	Yes	Cone	Linear*
F8	-67.31	++	Yes	Trans. Down Cone	Linear* Cubic*
F13	-23.69	+			
F14	-39.25	++	Yes	Cone	Linear*
F15	14.09	++	Yes	Cone	
F16	-59.24	++		Cone	Linear*
F17	-43.60	++		Cone	Cubic*
Trans Barr Fwd					
F5					
F6					
F10					
F8					
Trans Barr Rev					
F5	-6.92	++	Yes	Trans. Down Trans. Up	Cubic*
F6	-35.17	+	Yes		
F10	-45.85	++	Yes	Trans. Down Trans. Up	Linear* Linear*
F8	-57.60	++	Yes	Trans. Down Trans. Up	Linear* Linear*

(++) Improve with post values outside of variance of pre test. (+) Improve, but post values are not outside of pre test variance. (-) Worsens, but not beyond variance of pre-test. (--) Worsens with post values outside of variance of pre test.

Table 13. Subject 7 Kinematic Results

	Subject 7				
	Pre/Post Change (%)	**Improve?**	**Decrease in Variance?**	**Training Monitoring Trend**	**Linear Model (p<0.05*)**
Cone					
F4	20.35	--		Cone	
F6	34.69	--		Cone	Cubic*
F8	-60.21	++	Yes	Cone	Linear*
F13	-46.34	++	Yes	Cone	Quadratic*
F14	-41.86	++	Yes	Cone	Cubic
F15					
F16	-7.07	+			
F17	-43.53	++	Yes	Cone	Linear*
Trans Barr Fwd					
F5	2.77		Yes		
F6	17.65	-			
F10	4.65		Yes		
F8	-39.56	++	Yes		
Trans Barr Rev					
F5	-5.06	+	Yes		
F6	-39.02	++			
F10	27.63	++			
F8					

(++) Improve with post values outside of variance of pre test. (+) Improve, but post values are not outside of pre test variance. (-) Worsens, but not beyond variance of pre-test. (--) Worsens with post values outside of variance of pre test.

Results

Based on this type of analysis, it can be seen that Subjects 1, 2, 6 and 7 do have features that move beyond the variance of their pre test and further decrease in variance. Subjects 2 and 6 show the best results with almost all of their features showing improvement, and most improved beyond the variance of their pre-test with decrease post test variance. Subject 7, the most impaired subject in this group shows more inconsistent results with some features in the cone task worsening with decreasing variance. This inconsistency is likely related to the observations seen with Subject 8 in the system's current inability to adequately address moderate impairment users.

Inconsistencies can also be seen in Subject 1, however there may be extra factors affecting the results of Subject 1's pre versus post test results. About half way through the study, the subject began having personal issues that affected performance. In addition to emotional distress, the subject often reported being very tired from having to pack and move. Since the study already had a limited number of subjects, we were reluctant to remove this subject's results. However the inconsistent kinematics seen may be a result of these external factors.

Inconsistency can also be seen in Tables 14-16, which shows the results for Subject 3, 4, and 5. As previously shown, these subjects predominantly began the study with kinematic measures within the range of non-impaired measurements. The inconsistent changes in kinematic measures seen in Tables 14-16 could be an indication that HAMRR did not provide enough of a challenge for more mildly impaired participants. This is an important consideration for future work.

In addition to seeing many features improve for this subset of subjects, it can be seen that while there may be inconsistencies in the lower level features of the cone task, there is more consistent improvement seen in the higher-level features of the complex transport over a barrier task.

Analyzing Trends Within Subjects

In addition to assessing the pre post comparisons for individual subjects, we decided to look at the data sampled during the training monitoring sessions to see if the data overall trended in the direction indicated by the change in pre and post kinematic values.

Therefore, from the subset of Subjects 1, 2, 6, and 7's data (that had post data outside the variance of the pre test as well as decreasing variance) pre and posttest feature measurements were compared with the same feature in training monitoring tasks. The selected trends to analyze were chosen based on visual inspection of the data. If the overall trend of the training monitoring data seemed to fit the change between pre and post, then it was used in the subsequent analysis. In other words, the training monitoring task feature data had to visually look like it was increasing or decreasing with an increasing or decreasing pre to post change respectively. The training monitoring features that were compared with the pre and post features can be seen in Tables 10-13.

In order to determine the significance of the trend, linear models were applied to the raw data, with each feature being a function of the session number it was collected on (0 = Pre Test; 6 = First Training Monitoring Session; 9 = Second Training Monitoring Session; 12 = Third Training Monitoring Session; 15 = Fourth Training Monitoring Session; 16 = Post Test). Only a linear, quadratic and cubic model were applied as it was viewed that the limited amount of data would not support conclusions from higher order models. Each model was applied to the data, and the lowest order model that had a non-significant lack of fit ($p > 0.05$) was selected. Once the model was selected, the significance of the slope of the model was assessed ($p < 0.05$). The results of the model selection can be seen in Tables 10-13.

From this analysis, it can be shown that many of the pre to post comparisons do not seem to be aberrations, but in fact have trending data. Therefore, this seems to indicate for the features identified, that there is a possibility for these features to improve with training, given that the starting impairment level of the subject is closer to Subjects 1, 2, 6 and 7 rather than 3, 4 and 5. It also seems to indicate, given the decreases in variance, that the feedback has the capability to have a stylizing effect on lower level and higher-level features.

Due to the small sample size used to form the model, it is difficult to use the models as predictive tools to determine what effect more training may have had on the subject. However, because different models are found to fit the data significantly (and the distribution of possible trending improvement is varied), the training seems to provide different kinematic challenges for each subject.

Table 14. Subject 3 Kinematic Results

			Subject 3		
	Pre/Post Change (%)	**Improve?**	**Decrease in Variance?**	**Training Monitoring Trend**	**Linear Model (p<0.05*)**
Cone					
F4	9.77	--	Yes	Cone	Quadratic*
F6	-20	+	Yes		
F8	-12.48	+	Yes		
F13	7.94	-			
F14	-7.10	+	Yes		
F15	-11.82	++	Yes	Cone	Quadratic*
F16	17.97	-			
F17	-35.12	++	Yes	Cone	Quadratic*
Trans Barr Fwd					
F5	3.72	--	Yes	Trans. Up	
F6	100	-			
F10	-8.07	+	Yes		
F8	-26.50	++	Yes	Trans. Down	Linear*
				Trans. Up	Linear*
				Cone	Linear*
Trans Barr Rev					
F5	6.67	--		Trans. Up	
F6	-50	+	Yes		
F10	-1.09		Yes		
F8	-23.03	+	Yes		

(++) Improve with post values outside of variance of pre test. (+) Improve, but post values are not outside of pre test variance. (-) Worsens, but not beyond variance of pre-test. (--) Worsens with post values outside of variance of pre test.

Table 15. Subject 4 Kinematic Results

			Subject 4		
	Pre/Post Change (%)	**Improve?**	**Decrease in Variance?**	**Training Monitoring Trend**	**Linear Model (p<0.05*)**
Cone					
F4	**				
F6	(0.2 to 0.4)	-			
F8	10.71	-	Yes		
F13	12.67	-			
F14	-9.21	+	Yes		
F15	**				
F16	(0.01 to 0.03)	-		Cone	Cubic*
F17	**				
Trans Barr Fwd					
F5	7.65	--	Yes		
F6					
F10	-30.06	++	Yes	Trans. Up	Linear*
F8	-47.69	+	Yes		
Trans Barr Rev					
F5	9.12	--			
F6	(0 to 0.2)	-			
F10	12.60	--	Yes	Trans. Down	Linear
				Trans. Up	Cubic*
F8	-61.89	++	Yes	Trans. Down	Quadratic*
				Trans. Up	
				Cone	Quadratic*

(++) Improve with post values outside of variance of pre test. (+) Improve, but post values are not outside of pre test variance. (-) Worsens, but not beyond variance of pre-test. (--) Worsens with post values outside of variance of pre test. (**) Subject 4 attempted to complete the cone tasks as fast as possible and thus his data has been removed for time based features have been removed.

Table 16. Subject 5 Kinematic Results

	Subject 5				
	Pre/Post Change (%)	Improve?	Decrease in Variance?	Training Monitoring Trend	Linear Model (p<0.05*)
Cone					
F4	9.87	-	Yes		
F6	(0 to 1)	-			
F8	-29.36	++	Yes	Trans. Down Cone	Linear
F13	-11.47	+	Yes		
F14	12.17	-	Yes		
F15	-13.22	++	Yes	Cone	Cubic*
F16	(0 to 0.06)	-		Cone	
F17	-9.84	+			
Trans Barr Fwd					
F5	-1.19				
F6					
F10	6.60	-			
F8	-16.67	+	Yes		
Trans Barr Rev					
F5	-1.07		Yes		
F6	(0.2 to 0)	+	Yes		
F10	7.55	-	Yes		
F8	-56.67	++	Yes	Trans. Down Trans. Up Cone	Linear*

(++) Improve with post values outside of variance of pre test. (+) Improve, but post values are not outside of pre test variance. (-) Worsens, but not beyond variance of pre-test. (--) Worsens with post values outside of variance of pre test.

Kinematic Correlations with Clinical Scores

In addition to looking at both clinical scores and kinematics individually, it was desired to see if there were any correlations between them. This analysis was completed at the group level, both including all subjects as well as isolating subjects 1, 2, 6 and 7. Since the clinical scores only have one value per component and the features had multiple trials per pre and posttest, the average value of each feature was compared with each clinical score to assess correlation. This analysis was only completed for the complex task (transport over a barrier) as the previous HAMRR study had already shown correlation with low level kinematic features of the cone task with clinical scores. A Pearson correlation coefficient and significance was generated for each group comparison and can be found in Tables 17-28.

Table 17. Transport Over a Barrier Forward Pre Test Kinematics Correlations with Clinical Scores (All subjects)

	Torso Movement Sum		**Total Time**		**Speed Phases**		**Path Ratio**	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	-0.7241	0.0657	-0.8552	0.0068	-0.8827	0.0037	-0.6831	0.0618
FMA	-0.7891	0.0349	-0.7950	0.0184	-0.7613	0.0282	-0.6966	0.0549
WMFT TT	-0.9454	0.0013	0.9548	0.0002	0.9656	0.0001	0.8907	0.0030
WMFT FAS	-0.8251	0.0223	-0.9455	0.0004	-0.9032	0.0021	-0.7506	0.0319

Table 18. Transport Over a Barrier Forward Pre Test Kinematics Correlations with Clinical Scores (Subjects 1, 2, 6 and 7)

	Torso Movement Sum		**Total Time**		**Speed Phases**		**Path Ratio**	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	-0.8167	0.1833	-0.9480	0.0520	-0.9785	0.0216	-0.6864	0.3136
FMA	-0.7336	0.2664	-0.5269	0.4731	-0.4440	0.5560	-0.5261	0.4739
WMFT TT	0.9306	0.0694	0.9184	0.0816	0.9006	0.0994	0.8126	0.1874
WMFT FAS	-0.9547	0.0453	-0.9556	0.0444	-0.9394	0.0606	-0.7634	0.2367

Table 19. Transport Over a Barrier Forward Post Test Kinematics Correlations with Clinical Scores (All subjects)

	Torso Movement Sum		**Total Time**		**Speed Phases**		**Path Ratio**	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	-0.5613	0.2465	-0.5723	0.2353	-0.5623	0.2455	-0.2351	0.6538
FMA	-0.8304	0.0407	-0.7550	0.0827	-0.7562	0.0819	-0.7115	0.1129
WMFT TT	0.9952	0.0000	0.9732	0.0011	0.9936	0.0001	0.7671	0.0751
WMFT FAS	-0.8581	0.0288	-0.9187	0.0096	-0.8839	0.0194	-0.7177	0.1083

Table 20. Transport Over a Barrier Forward Post Test Kinematics Correlations with Clinical Scores (Subjects 1, 2, 6 and 7)

	Torso Movement Sum		**Total Time**		**Speed Phases**		**Path Ratio**	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	-0.9886	0.0961	-0.9998	0.0117	-1.0000	0.0018	-0.8909	0.3002
FMA	-0.8062	0.4031	-0.7210	0.4874	-0.7061	0.5009	-0.9516	0.1989
WMFT TT	0.9962	0.0556	0.9990	0.0288	0.9978	0.0422	0.9179	0.2597
WMFT FAS	-0.9910	0.0854	-1.0000	0.0010	-0.9998	0.0124	-0.8984	0.2895

Table 21. Transport Over a Barrier Reverse Pre Test Kinematics Correlations with Clinical Scores (All subjects)

	Torso Movement Sum		**Total Time**		**Speed Phases**		**Path Ratio**	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	-0.7316	0.0616	-0.8253	0.0116	-0.7734	0.0244	-0.7920	0.0191
FMA	-0.8121	0.0265	-0.8042	0.0161	-0.8967	0.0025	-0.9239	0.0010
WMFT TT	0.9671	0.0004	0.8498	0.0075	0.9283	0.0009	0.9598	0.0002
WMFT FAS	-0.8298	0.0209	-0.9512	0.0003	-0.9182	0.0013	-0.8811	0.0038

Table 22. Transport Over a Barrier Reverse Pre Test Kinematics Correlations with Clinical Scores (Subjects 1, 2, 6 and 7)

	Torso Movement Sum		**Total Time**		**Speed Phases**		**Path Ratio**	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	-0.8224	0.1776	-0.9183	0.0817	-0.7842	0.2158	-0.6071	0.3930
FMA	-0.7656	0.2344	-0.6408	0.3592	-0.8272	0.1728	-0.9432	0.0568
WMFT TT	0.9596	0.0404	0.9648	0.0352	0.9744	0.0256	0.9260	0.0740
WMFT FAS	-0.9745	0.0255	-0.9874	0.0126	-0.9754	0.0246	-0.8984	0.1016

Table 23. Transport Over a Barrier Reverse Post Test Kinematics Correlations with Clinical Scores (All subjects)

	Torso Movement Sum		**Total Time**		**Speed Phases**		**Path Ratio**	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	0.1384	0.7937	-0.8552	0.0068	-0.8146	0.0138	-0.7933	0.0188
FMA	-0.6142	0.1945	-0.7195	0.0442	-0.8310	0.0106	-0.8162	0.0135
WMFT TT	0.7971	0.0576	0.9640	0.0001	0.9869	0.0000	0.9423	0.0005
WMFT FAS	-0.7872	0.0631	-0.7300	0.0398	-0.7847	0.0211	-0.7120	0.0476

Table 24. Transport Over a Barrier Reverse Post Test Kinematics Correlations with Clinical Scores (Subjects 1, 2, 6 and 7)

	Torso Movement Sum		**Total Time**		**Speed Phases**		**Path Ratio**	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	0.8702	0.3280	-0.9680	0.0320	-0.7207	0.2793	-0.6455	0.3545
FMA	-0.8810	0.3138	-0.4601	0.5399	-0.8477	0.1523	-0.9565	0.0435
WMFT TT	0.7894	0.4208	0.9082	0.0918	0.9768	0.0233	0.8719	0.1281
WMFT FAS	-0.7559	0.4544	-0.6894	0.3107	-0.9248	0.0752	-0.8012	0.1988

Table 25. Transport Over a Barrier Forward Change in Kinematics Correlations with Change in Clinical Scores (All subjects)

	Torso Movement Sum		Total Time		Speed Phases		Path Ratio	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	-0.5820	0.2256	0.4778	0.3378	0.2823	0.5878	-0.2908	0.5762
FMA	0.6046	0.2036	-0.4268	0.3987	-0.5715	0.2361	-0.4318	0.3926
WMFT TT	0.9697	0.0014	-0.5638	0.2439	-0.6380	0.1728	-0.0966	0.8555
WMFT FAS	0.4165	0.4114	0.4038	0.4272	0.0934	0.8603	0.0376	0.9437

Table 26. Transport Over a Barrier Forward Change in Kinematics Correlations with Change in Clinical Scores (Subjects 1, 2, 6 and 7)

	Torso Movement Sum		Total Time		Speed Phases		Path Ratio	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	-0.6366	0.5607	0.4530	0.7007	0.3223	0.7911	-0.0429	0.9727
FMA	0.8261	0.3811	-0.9292	0.2410	-0.9721	0.1507	-0.9910	0.0856
WMFT TT	0.9625	0.1750	-0.8800	0.3150	-0.8040	0.4054	-0.5337	0.6416
WMFT FAS	Nan	Nan	Nan	Nan	Nan	Nan	Nan	Nan

Table 27. Transport Over a Barrier Reverse Change in Kinematics Correlations with Change in Clinical Scores (All subjects)

	Torso Movement Sum		Total Time		Speed Phases		Path Ratio	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	0.4145	0.4139	-0.3440	0.4041	-0.4766	0.2325	-0.5093	0.1973
FMA	0.5027	0.3095	-0.3600	0.3811	-0.2443	0.5598	-0.1788	0.6718
WMFT TT	0.4513	0.3690	0.1657	0.6949	0.1789	0.6716	0.4179	0.3028
WMFT FAS	0.5140	0.2969	0.0976	0.8182	-0.0435	0.9186	0.2849	0.4941

Table 28. Transport Over a Barrier Reverse Change in Kinematics Correlations with Change in Clinical Scores (Subjects 1, 2, 6 and 7)

	Torso Movement Sum		**Total Time**		**Speed Phases**		**Path Ratio**	
	Corr.	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**	**Corr.**	**pVal**
MAL	0.7450	0.4649	-0.6929	0.3071	-0.6481	0.3519	-0.7561	0.2439
FMA	0.9989	0.0300	0.6537	0.3463	0.7786	0.2214	0.1364	0.8636
WMFT TT	0.1830	0.8828	0.9160	0.0840	0.4883	0.5117	0.9587	0.0413
WMFT FAS	0.2350	0.8490	0.3914	0.6086	-0.2481	0.7519	0.6970	0.3030

The pre test values for both directions of the transport over a barrier task showed significant correlations with the pre test clinical scores when all subjects were considered. In addition, fewer correlations were found when comparing the posttest features and clinical scores of both directions of the transport over a barrier task. Even though these tasks are part of the same activity (each task represents a different direction in which the transportable object was moved), the correlations are inconsistent when compared between the posttest results of both directions of the transport over a barrier task. In addition, when changes in both directions of the transport over a barrier task are compared with changes in the clinical scores, minimal to no correlations are found. For each of these comparisons, when isolating Subjects 1, 2, 6 and 7, fewer correlations were found than when looking at the whole group of subjects. Therefore it is inconclusive whether these features were adequate measures of impairment. The ability of the system to capture these high level features in variable complex movements may not have been as clear or stable as was needed. Further, it is possible that these features are combined together in very complex ways to improve performance of complex tasks and this combination cannot be captured by single feature analysis.

Therapist Ratings

As previously discussed, during each training monitoring session, the therapist recorded videos of the subject performing a set of tasks, and then rated the videos along provided criteria. The goal in analyzing the ratings was to see if they correlated with any of the features of a complex task (which in the training monitoring sessions was transporting an object between an off the table and on the table location) as well as the cone grasp task (since this was used throughout training as well as previously discussed analysis).

For each trial rated, the first score was based on a modified FAS, which rated the therapist's overall initial impression. The range of values were such that a 1 represented that the subject could not complete the task and a 5 represented that the quality of movement appeared to be similar to non-impaired movement. In addition to the overall initial rating, component ratings were also completed. For the cone, these included: Trajectory (Accuracy of moving the hand from the start to end position during the reaching phase), Compensation (Excessive shoulder elevation or abduction and/or torso flexion, rotation, or lateral flexion), and Manipulation (Opposition between thumb and at least two fingers about the diameter of the object). The transport task also had these ratings, but in addition included: Transport (Translating an object from one location to another while carried by the hand) and Release (Extension of fingers and thumb to remove hand from object following deposit). All of these component ratings were rated on a scale of 1-4. Finally, each task was rated with a final overall rating. This used the same overall rating scale as the initial overview rating, but was completed after completing all of the component ratings for a particular task.

In order to assess correlations between the ratings and kinematic features, the ratings for a particular set of reaches were averaged and measurements for each kinematic feature were averaged as well. Pearson correlation coefficients were generated for these correlations and are show in Tables 29-31.

Table 24. Cone Task Kinematics Correlations with Therapist Ratings

	F4		**F13**		**F14**		**F6**		**F15**		**F16**		**F17**		**F8**	
	R^2	**pVal**	**R^2**	**pVal**	**R^2**	**pVal**	**R^2**	**pVal**	**R^2**	**pVal**	**R^2**	**pVal**	**R^2**	**pVal**	**R^2**	**pVal**
Initial Overall	-0.84	0.00	-0.63	0.00	-0.55	0.00	-0.89	0.00	0.67	0.00	-0.91	0.00	-0.56	0.00	-0.88	0.00
Trajectory	-0.80	0.00	-0.41	0.02	-0.37	0.04	-0.82	0.00	0.63	0.00	-0.83	0.00	-0.37	0.04	-0.73	0.00
Compensation	-0.88	0.00	-0.65	0.00	-0.55	0.00	-0.91	0.00	0.65	0.00	-0.85	0.00	-0.39	0.03	-0.95	0.00
Manipulation	-0.87	0.00	-0.55	0.00	-0.64	0.00	-0.92	0.00	0.64	0.00	-0.91	0.00	-0.49	0.00	-0.92	0.00
Final Overall	-0.84	0.00	-0.62	0.00	-0.56	0.00	-0.89	0.00	0.66	0.00	-0.91	0.00	-0.57	0.00	-0.88	0.00

Table 25. Transport Down Kinematics Correlations with Therapist Ratings

	F3		**F5**		**F6**		**F8**	
	R^2	**pVal**	**R^2**	**pVal**	**R^2**	**pVal**	**R^2**	**pVal**
Initial Overall	-0.851	0.000	-0.755	0.000	-0.844	0.000	-0.859	0.000
Trajectory	-0.691	0.000	-0.361	0.059	-0.602	0.001	-0.707	0.000
Compensation	-0.957	0.000	-0.733	0.000	-0.914	0.000	-0.918	0.000
Manipulation	-0.791	0.000	-0.711	0.000	-0.776	0.000	-0.757	0.000
Transport	-0.463	0.013	-0.367	0.055	-0.418	0.027	-0.515	0.005
Release	-0.665	0.000	-0.400	0.035	-0.621	0.000	-0.642	0.000
Final Overall	-0.851	0.000	-0.755	0.000	-0.844	0.000	-0.859	0.000

Table 26. Transport Up Kinematics Correlations with Therapist Ratings

	F3		F5		F6		F8	
	R^2	pVal	R^2	pVal	R^2	pVal	R^2	pVal
Initial Overall	-0.774	0.000	-0.591	0.001	-0.758	0.000	-0.754	0.000
Trajectory	-0.717	0.000	-0.445	0.018	-0.699	0.000	-0.642	0.000
Compensation	-0.852	0.000	-0.722	0.000	-0.875	0.000	-0.842	0.000
Manipulation	-0.744	0.000	-0.781	0.000	-0.696	0.000	-0.807	0.000
Transport	-0.589	0.001	-0.478	0.010	-0.563	0.002	-0.634	0.000
Release	-0.613	0.001	-0.660	0.000	-0.604	0.001	-0.744	0.000
Final Overall	-0.774	0.000	-0.591	0.001	-0.758	0.000	-0.754	0.000

Within the cone task, strong correlations (coefficient > 0.7 or < -0.7) are found between all of the ratings and reach time, number of speed phases, normalized area, and torso compensation. This could begin to indicate that these features correlate the most with how a therapist would visually rate a reach. In contrast, the features that measured changes in trajectory and peak speed (which do not show strong correlations) may have been features that the system observed but were not noticed, or deemed relatively important, by the therapist. One possibility is that the camera used to record the videos provided only one, profile view of the subject completing the reach, and therefore, it may have limited perspective to comprehensively see trajectory error. Further, when training the therapists with the system, it was noticed that in relation to the speed of a subject completing a task, each therapist would encourage the subject to complete the task quickly (they would not penalize a subject for completing the task too quickly), whereas the system had a range of speed it trained the subject within, and would provide negative feedback if the subject reached above this range. In addition, weak correlations were found with the jerkiness feature. This may be due to therapists using other indicators (such as number of phases and the magnitude of those phases) as metrics to assess unsmooth movement. Also it is likely difficult to see jerkiness in a video from afar. Therefore, it will require further investigation to better understand if the trajectory, peak speed and jerkiness features are too sensitive within the system or if they are truly characterizing aspects of the subject's performance that are too fine for the therapist to want to or be able to observe. In addition, it should be determined if the video capture is reducing the ability of the therapist to observe fine features. For example, a therapist may need to the capability to focus on fine features to do the rating instead of a video showing the whole movement.

Within the transportation tasks, it was noted that overall, the Transport and Release do not seem to correlate strongly with the kinematic features. This is interesting because the system was not able to provide detailed feedback about each of these aspects. The transport phase efficiency was only considered as a part of a larger overall feature such as path ratio, and no training was provided on the release of the object. Resultantly, the selected features that were used to assess the kinematics may not be adequately capturing the overall aspects that therapist can assess visually. When looking at the correlations with the path ratio feature, it is noted that there are limited, inconsistent correlations with the therapist ratings. This result may provide some uncertainty in regards to value of the path ratio feature in its current configuration within the system. It would be interesting to investigate this further to see if other features may correlate better with the ratings. However, the presence of strong correlations with other components of the ratings seem to suggest there is some connection between the existing kinematic features and how a therapist rates the performance of a subject, and therefore supports their utility moving forward with the design of the system. It is also possible that the therapist may be observing some combination of these features that simple features like path ratio may not capture adequately, which would call for more complex quantitative features.

5.1.4 Subject Survey Results

At the completion of each training monitoring session, the subject was asked a series of questions in regards to their experience using the system. These questions ranged in assessing general reactions to using the system and its perceived utility (as well as identifying components that may have been difficult to understand or interact with) to assessing their own progress. The responses were a combination of Likert scale and free response. All of the results were logged and saved as part of the other training monitoring data on the iPad app.

All subjects reported back that the amount of therapy was either too little or just enough, further highlighting the lack of challenge that could be provided with the existing protocol paths. This is reflected as well in the subjects' responses to a question about the challenge of the training tasks, in which all subjects (including the more impaired subjects) reported that the system challenge ranged from neutral to very easy. All subjects also reported liking using the system, with two neutral responses recorded at individual training monitoring sessions for Subjects 2 and 4. Most subjects reported using their affected arm more during the study, with only Subject 7 reporting consistent affected arm usage to that previous to using the system. Subjects 7 & 8 reported neutral to decreasing effects on their energy levels after using the system, which seems to suggest that the provided challenge many not have been perceived to be beneficial, but they were still physically demanding. Finally, all subjects reported neutral to better moods after using the system.

The conclusions that can be drawn from these questionnaire results are limited, however the system does show promise to encourage compliance and desire to use the system. It seems that the subjects overall did not feel that the system was able to provide them with a desired challenge level, regardless of starting impairment level even though some reported decreased energy after using the system. Therefore it seems that not enough challenge is being provided to higher functioning users and not the right challenge type is being provided to more impaired subjects.

5.2 System Stability and Usability Results

Overall the system was able to complete eight full protocols both under control of the system designers as well as physical therapists. Subjects were able to successfully setup the system on their own and control the flow of demos and tasks through the embedded table buttons. Some minimal assistance was needed to apply the torso rigid body below the shoulder, and some subjects needed assistance moving the chair in the correct position. These assistance steps were minimal and safely represented help that a caregiver or family member could easily provide. However, some stability and user experience issues were noted that will present good indicators for future work.

5.2.1 Marker Based Tracking of Complex Movement with Limited Cameras is not Adequate for Complex Task Interactive Training

In the design process of HAMRR, motion capture through OptiTrack provided the most expedient way to capture detailed information based on the existing code from AMRR and team experience. It was a beneficial decision to move ahead with this type of hardware sensing, however there were problems observed.

As has been previously discussed, the original torso tracking solution was to utilize the Kinect for marker-less tracking. However, due to the noise seen in early iterative tests this solution was removed. Due to the lack of time to test the replacement marker solution, occlusion and angle orientation estimation errors could not be assessed properly.

In addition, frequent marker swap was found during use of the transportable cylinder, which would cause the interactive training to stop (as the system was looking for the correct marker configuration). As subjects would lift the cylindrical object out of the view of the markers or any other occlusions would occur, there was a significant chance marker swap would occur. This was likely a problem due to using an older version of Tracking Tools. For more details on the marker-based problems, please refer to Appendix B3.

5.2.2 Tangible Objects Necessitate Sensing of Interaction

The tangible objects provided stable data to the system in regards to object manipulation. Occasionally an object's hardware would malfunction, but it could be quickly replaced in the middle of the session by a staff member or therapist by using the interface provided by the Sensing Plugin.

While most tangible object interactions were found to be stable, the transportable cone (which was designed as a quick solution to allowing subjects with more impaired had aperture to transport an object) proved to be problematic in use. Since the object did not have any active physical sensors, as previously discussed, the "grasp" of the object was detected by the speed and location of the wrist marker. However, with more impaired subjects, the reach was completed in two main stages: the patient got their hand near the object and then adjusted the hand to grasp. The system interpreted the completion of the first step as a grasp and thus misleading grasp success feedback was provided to the subject. This object and interaction will need to be reconsidered in the future.

5.2.3 HAMRR is Lacking a Manner to Report Errors

When the system was under the control of the two system designers, problems that arose could be diagnosed on the spot due to the nature of the knowledge of how the system worked and was built. However, this was not true of therapists using the system. Therefore, some documentation was created to explain the use of the system and source of common problems, but this was not enough as Nicole and I had to be on Skype throughout the sessions to debug problems live. This seems to indicate that the system would benefit from a standardized way of reporting errors to the controller along with clear directions for resolving them.

5.2.4 The Existing Interface is Not Easily Usable by Therapists

While the therapist was able to use the system interface on their own, it lead to a significant amount of time and documentation to explain how the interface should be used. This was not ideal for a situation in which the therapist needed to be quickly trained how to use the system so additional subjects could be recruited. Many times the interface led to confusion and frustration, which would detract from the time available to the patient and would highlight frustrations that the subject might be having with the training. This is a very important factor, which must be addressed in the future for anyone that will control the system. Therapist and patient time is extremely valuable and that time needs to be maximized with the system if HAMRR is to provide any benefit. Therefore rethinking the user experience (including that of the therapist) should be a top priority of the next system iteration.

5.3 Discussion of Results

From these results, it seems promising that unsupervised INR training can be feasible and has the potential to induce functional improvement. All of the subjects were able to complete the protocol (Subject 6 was not able to attend the One Month Follow Up session and some adaptations needed to be made to Subject 7 and Subject 8's protocols to accommodate their more moderate impairment). Also significant group level changes were observed in the clinical scores assessed before and after the study.

5.3.1 Inconclusive Connection of Training to ADLs

The retention of the improvements in clinical scores (assessed one month after the completion of the study) was not found to be as conclusive as the pre to post comparisons and will require further investigation. While there was not retention in clinical score assessed improvement, most subjects reported using their affected limb more during the study. The full scope of this increased use is hard to determine from the patient surveys, but may indicate some training benefit to increasing confidence in the use of the impaired limb. However, determining if this increase in confidence is due to HAMRR or just the increase in physical therapy will need to be determined.

5.3.2 Training Showed Promise to Integrate Component and Complex Training

The unsupervised training provided by HAMRR has shown some promise that movement quality improvements can be correlated with functional improvements. While this was not seen significantly at the group level, when looking at individual subjects (especially Subjects 1, 2, 6 and 7) there are promising trends in kinematic features that, with further exploration, may be found to correlate with functional improvement measures. The indications of integrating component and complex training with HAMRR also suggest the benefit of reductionist hierarchies and constructivist learning protocols to structuring interaction scenarios for physical therapy.

Previous work with AMRR has already shown how kinematics can be used to evaluate impairment for simple repetitive tasks. However, the presence of some strong trends within individual subjects may suggest that particular kinematic measures could serve as a good starting point to assess patient impairment of complex tasks through kinematics and thus inform individualized protocol composition.

5.3.3 Provided Protocols Need More Opportunity for Adaptations to Patient Performance

The variability of the individual subject kinematic trends do seem to indicate that INR training needs to be highly customized to patient needs, both in terms of tasks provided and how these tasks are trained over the course of a protocol. This stresses the need for physical adaptability of objects and system components as well as more adaptable training protocols. It is possible that with more adaptable protocols, stronger correlations between movement quality and functional improvement may be found, but this will require further investigation. The significance of the trends (through applied models) shows promise that individual progression may be able to be assessed throughout a training protocol within individual features in order to make more adaptive training decisions.

5.3.3.1 Accommodating More Impaired Subjects

While Subjects 7 and 8 were able to complete the overall protocol with some adjustments (mainly the introduction of the transportable cone object) the system was not able to consistently address their needs. While observing Subject 8's use of the system, it became clear that number of reaches within an individual day was too tiring, and thus the physical therapist needed to cut many of the sessions short. The physical therapist provided the feedback that the order and amount of transportable tasks should be reconsidered, as currently, there were too many and they were typically weighted towards the end of the session (where Subject 8 was becoming too fatigued to attempt the tasks). Future protocols should make sessions lengths more dynamic based on fatigue levels and the most challenging tasks should not be held until the end of the session.

5.3.3.2 Accommodating Mildly Impaired Subjects

As seen in the inconsistent results of Subjects 3, 4 and 5, it seems very likely that HAMRR was not able to provide an appropriate challenge for more mild impaired subjects. This issue is also confirmed by the reported challenge of the tasks in the patient survey results. Therefore, there is a need for future protocols to look for other dimensions of adaptability where additional challenge can be added.

5.3.4 Hybrid Kinematic and Therapist Evaluations are Possible

HAMRR, through the training monitoring sessions, has also shown that monitoring of therapy progress and related protocol adaptation can be achieved through therapist and computer hybrid evaluations. While these correlations will we require further investigation as previously discussed, the methodology by which they were collected and integrated was supported by this study. As mentioned previously, further investigation will be required to see if slopes of applied linear models and the correlation of these features with therapist ratings can provide adequate metrics to make adaptation decisions. Preliminary work at this time has already begun to show the possibility of correlating kinematics with therapist ratings to produce a decision tree model for rating component level performance of a simple task.[50]

5.3.5 Sensing Infrastructure Was Not Able to Adequately Capture Features in Some Complex Tasks

As previously discussed, the key turn task data was very noisy and became unusable in the overall analysis of HAMRR. The marker setup of rigid bodies in the camera space in conjunction with frequent marker occlusion of the torso or wrist markers (sometimes just due to the natural rotation of a key) created gaps in the data that eliminated significant portions of the reach. It is still very important for subjects to be assessed with an untrained task to see how elements of training may be transferring to other activities, but the current sensing infrastructure needs to be addressed first.

5.3.6 HAMRR Was Easy to Use by the Patients, but Not by the Therapist

Overall, patients were able to setup and use the system very easily. Each subject was able to follow onscreen instructions to setup the objects in the table at the start of each session. In addition, in nearly every case, the subjects skipped replaying demos very early on in the protocol. However, it was found that some instructions became over repetitive for the subjects.

In contrast, the system was not easily usable by the therapist for Subject 7 and 8's sessions. While some training and documentation was provided on the use of HAMRR, the interface was not easy to use or as helpful as it needed to be (even with the previously discussed improvements added before the second stage of the pilot study). The system did not provide sufficient information to the therapist on its current state or how to diagnose and recover from a system error. This led to frustration both for the therapist and subject. The control of HAMRR lacked from the application of experience design and in future work the control should be redesigned significantly.

5.3.7 The Modular Architecture was Able to Support New Additions to the System

The need to introduce a new transportable cone object to HAMRR was quickly supported by both modular code and hardware design. Changes to the Adaptation control GUI also benefitted from the modular design of the Adaptation Plugin code. The previously described problems with these components were not the result of the architecture (software or hardware) but rather a lack of experience design considerations (both for therapy and user control).

5.3.8 HAMRR Should be Progressed Forward with Identified Issues Addressed

Overall, it seems promising that INR systems that use a reductionist hierarchy approach to structuring feedback and interaction scenarios, combined with constructivist learning protocols focused on aggregate learning show promise to provide many key benefits. They can help maintain engagement over longer protocols as was evidence by the compliance and completion of HAMRR protocols. They can induce integrated active learning of movement components and related complex tasks as shown by the promising improvements clinical scores and individual trends in kinematic measures. Finally, given that these promising improvements were found in a minimally supervised environment with a fixed protocol, it suggests that the overall impact of unsupervised INR could be very beneficial with more reactive adaptation capabilities.

CHAPTER 6

FUTURE DIRECTIONS: SYSTEM REDESIGN, FUTURE ADDITIONS, AND PROJECT MANAGEMENT

With the desire to progress the system development further in the future, here are some considerations that reflect lessons learned in the application of the original design constraints through integrated design approaches for a complex INR system. These considerations represent ideas that should be incorporated with the previously presented design constraints to create an improved design methodology. Section 6.1 presents suggestions for code stability and extensibility improvements. Section 6.2 presents suggestions for improving and standardizing the system architecture. Section 6.3 presents suggestions for improving the hardware of the system. Section 6.4 presents user experience improvements. Section 6.5 presents specific user experience ideas for creating opportunities for the patients to express their own creativity. Section 6.6 presents suggested changes for the existing therapy protocols while Section 6.7 shows how automated adaptation may fit within the system. Section 6.8 provides reflections and suggestions for project management within an interdisciplinary team.

6.1 Code Stability and Extensibility Improvements

The code design and work should ultimately be released as an open source project. The sensing, analysis and feedback paradigms (as well as overall architecture) that have been thoroughly tested could prove to be of use to other neurorehabilitation applications as well as other interactive systems. Before this code is released, improvements should be made to improve the code's stability and allow future users to extend the code more easily.

6.1.1 A Modular Architecture Needs to be Supported by Clear Delineation of Plugin to Plugin Communication

As has been discussed in software design approaches, having a clear architecture can help support cohesive and fast, iterative development. As a result, considerable time was spent on determining how the plugins would be split functionally as part of the overall architecture design. Once the functionally separate plugins were identified, the next stage of the design was to identify how the plugins would communicate with one another. In order to support modular components, the communication between plugins should be standardized so that new plugins can receive and send data using tested and integrated communication methods.

The two main communication methods used by HAMRR were protocols and notifications. Protocols allow methods to be implemented by a different class than the caller. Notifications are real time events that get posted, and any class can subscribe to the notifications to respond to the event. Currently, the protocols are used primarily for one-off commands (such as control commands from the adaptation plugin) while notifications are used for real-time, low-level communication (such sending frames of camera data). This structure was a great start and allowed plugins to easily send and access needed information.

In order to further support the modular architecture, protocols should be used further to access specific information from other plugins that is currently sent as data objects. What I propose is that the data object sent via NSNotification should just notify the receiving class that a new frame of data is ready. Then the receiving class calls a protocol (implemented by the sender class) to access specific data. Currently the "observer class" has to import external header files to parse the data object on it's own.

As an example of this implementation, I present the example of the Sensing Plugin sending updated data to the Motion Analysis Plugin. In the updated implementation, the Sensing Plugin would broadcast a notification that a new frame of data is ready (once it parsed the incoming multicast data from OptiTrack). The Motion Analysis Plugin, being a subscriber of this notification, would call a series of functions within a Camera Data Frame protocol, implemented by the Sensing Plugin, in response. The functions would act as "getters" and return specific values for the X, Y and Z location of the wrist marker (among other data from the camera data frame as well). (Figure 22) In this way, the Motion Analysis plugin doesn't have to know anything of the data structure of the marker data. It only needs to know when a new frame of data is available and a means to access specific parts of that new data.

A similar methodology should also be applied to how the Adaptation Plugin sends commands to other plugins. Please refer to Appendix B4 for more details.

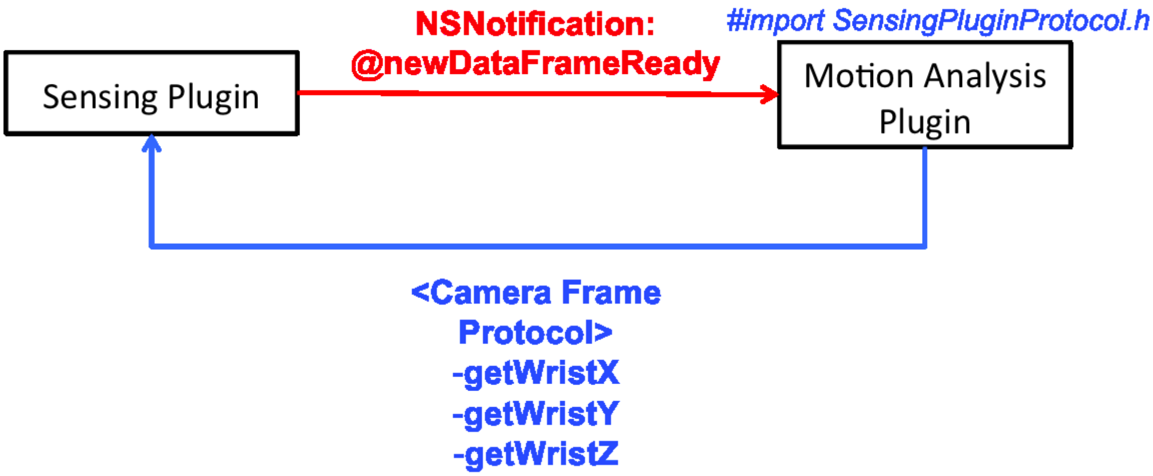


Figure 22 – Revised data communication between plugins.

6.1.2 Tangible Plugin Should Implement Flexible Interfaces for Increased Modularity

Modularity is also important at the hardware level. Given the session-to-session changes in objects used for the therapy protocol, both the system and therapist may want to interchange or replace objects frequently. The Tangible Plugin needs to support this flexibility. Currently, due to instabilities seen in initial testing, the Tangible Plugin was configured such that serial ports only open and close once at program open and quit respectively. As a result, once a port was opened it was configured for a particular object and could not be reconfigured for another object until the program was restarted. Not only is this limited from a modularity perspective, but it is also a bad user experience design. Therefore, in order to improve this design, interfaces should be flexible such that a port opens and a unique data parser can be assigned to the port. That way the manner in which the data is parsed is not linked to the opening or closing of a port, and thus, once a port is open, any sequence of data parsers can be attached to the port as needed. For more details on implementation details, see Appendix B5.

In the development of the system, many problems were encountered when developing the Tangible Plugin interface for receiving data from the tangible objects. If there was ever a crash from any of the plugins, the serial ports were not closed properly, and as a result, the computer would need to be forcefully shutdown and restarted. To avoid this problem at the time, it became necessary to run DASH as a compiled app and incorporate functions for a clean exit that could respond to thrown exceptions or commands to quit the program to safely close the ports. As a result of the problems seen in the stability of the serial port communication, the Tangible Plugin was configured such that only a fixed number of ports could be opened every time DASH was started. This ensured that a specific port could only be opened once and it would have to be safely closed via the port closing resulting from quitting HAMRR. While this created stability that reduced the number of crashes and also vastly improved the recovery time from a crash, it was not an ideal solution for the long term.

6.1.3 Existing Abstracted Interfaces Should be Utilized Further for Better Code Generalization

As I was writing the HAMRR Analysis program code, I was also exploring in detail the applicability of design pattern models to my work. As a result, the HAMRR Analysis code already shows an example of the benefit of abstracted interfaces.

When I was designing the code for the analysis program, I decided that the highest level of functional difference was a set of data. Since there were no features that were analyzed across sets, it made sense to only be concerned with sets of data at the highest level. As was previously discussed, each set represents a group of trials in which the tasks and feedback parameters were the same. Therefore, each set (HABaseSet) would most likely differ in how it defined trials (which reach states it provided) as well as the analysis algorithms that would need to be conducted on a particular trial. Therefore a group of analysis classes were created. They were all subclasses of HABaseSetAnalysis. Therefore, each instance of HABaseSet needed to have a particular instance of HABaseSetAnalysis (the type of the instance was defined by the type of set loaded from the database). The same was done for the set's trial segmentation algorithms (HABaseTrialSegmentation) and initialization algorithms (HABaseSetInitializers). Therefore, with this design, the set specific algorithms could be defined at runtime and were tied to a specific implementation of a HABaseSet sub-class.

However, while this design is a step in the right direction, there are some code refactoring steps to take place. By separating all of the set specific algorithms into generalized interfaces, there really isn't a need for subclasses of HABaseSet (like HASetLevel2 and HASetLevel3). Currently, the only difference is in how the set saves and loads segmentation files as well as saving evaluation results to file. However, these could also be abstracted to a general interface, such that there is general HASaveSegmentation (which could likely be combined with the HABaseTrialSegmentation class) and HASaveFeatures class, which specific subclasses for set unique needs. This would remove two unnecessary, general classes and instead provide more specific, encapsulated classes, which again would support future re-design and extensibility.

6.1.4 Class Responsibilities Should be Refined to be More Specific

As previously discussed, the macro-scale functional decomposition was a combined effort of the design team in coming up with a design where large-scale functionality could be divided into individual plugins. However, the concepts of modularity and functional decomposition change when looking within a plugin, which is where most of my work existed. Currently, there are many classes that hold too many responsibilities and therefore if code needed to change (which is an inevitability in iterative design), changes to features in one class could be hard to separate from other features in the same class.

As an example from a plugin I was solely responsible for (Adaptation Plugin) is ADMainController. As previously discussed, this class serves as the main class for the Adaptation Plugin. When looking at this class, the following functionalities can be identified: control video recorded program (setup and send commands), trigger start up functions, trigger opening ports, handle incoming NSNotifications, parse and translate incoming table button input, trigger specific demo videos, determine which demos need to be played (and remember which ones have already played), make sure all plugins are notified of the details of an upcoming set, trigger the tangible object socket test, trigger the manual stop of a set, and parse incoming sensing data stream states. What results is a class with over 2000 lines of code with too varied responsibilities. To further highlight why this is a problem, for example, if changes were needed for the sequencing of demos, changes would have to be made in the ADMainController, which if not careful, could impact other functionalities of the main class of the plugin. This is a design that should be corrected by functionally separating out some of the pieces of ADMainController into separate classes (as a start): video recorded handler, a startup command class, an incoming notification class, an outgoing notification class, a demo controller and a stop command class. In this way, the functionality is a little but more separated and it allows for easier extensibility. If someone wants to change how the Adaptation Plugin calls for specific demos, those changes could be encapsulated within the demo controller (and it's related classes). In addition, this also makes it easier for someone to remove functionality. If demos are not required for a specific application of the code, then a null demo controller could be put in place, and none of the other code needs to change.

6.1.5 Template Design Pattern Should be Utilized Within Motion Analysis for Quick Extensions

The Motion Analysis plugin, which is responsible for all data analysis and drives the live interaction, needs to also be updated for better modularization. Primarily this should be done to allow for easy extension of the reach states. The reach states should be encapsulated as they are one of the key areas of difference between each analysis engine (real time, Level 2 and Level 3). (Figure 23) Not only would this further highlight the reach states (across analysis) that are the same or similar, and thus could be refactored, but it would also allow for someone to input their own reach states much easier.

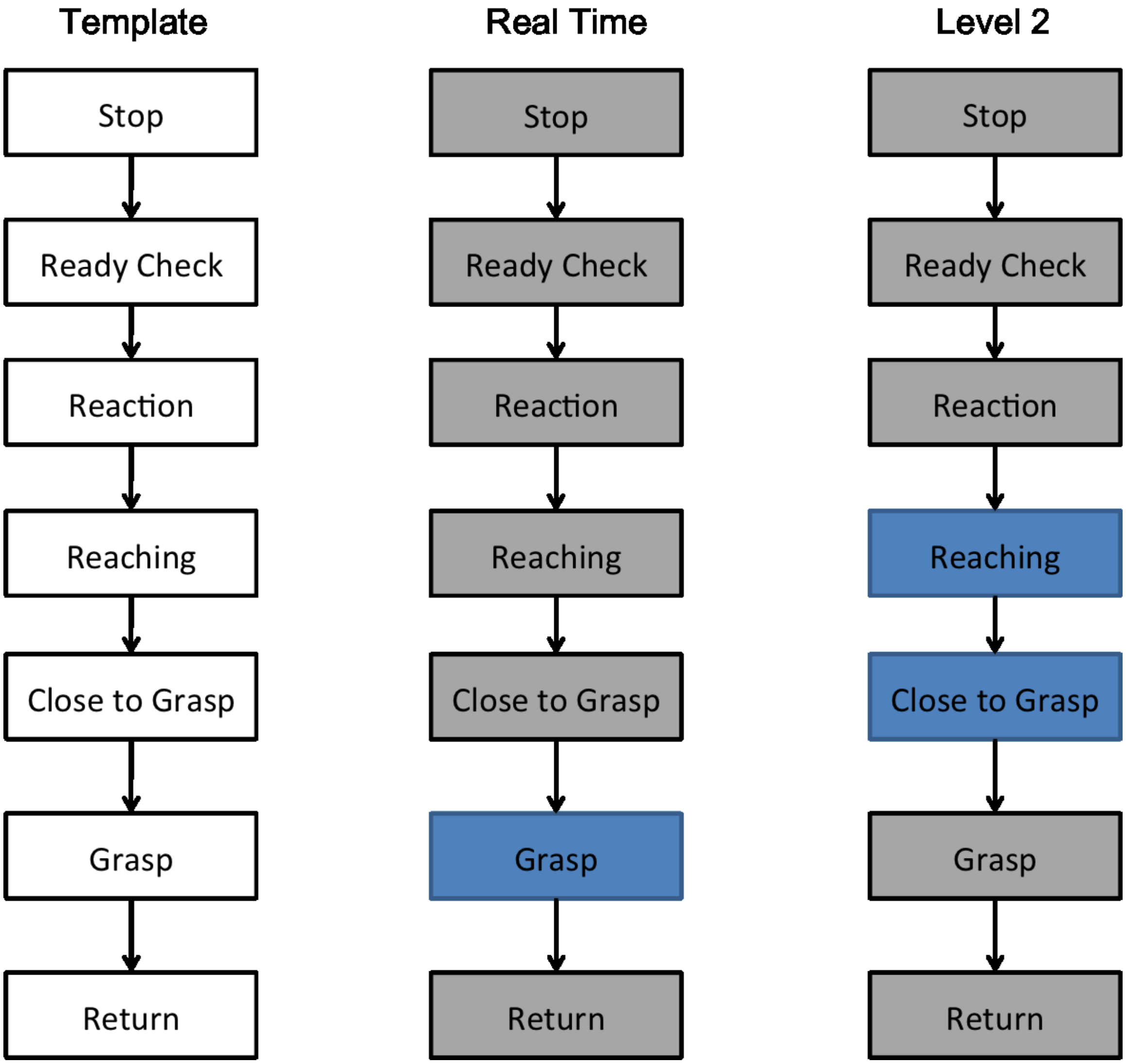


Figure 23 – Encapsulate reach states in templates. With this redesign, new combinations of reach states can use existing template code (grey) and only modify what is new (blue)

To implement this change, a Template Method approach should be taken. This approach would create a general skeleton for an algorithm and allow subclasses to implement specific parts of the algorithm in general. Therefore, a sequence of reach states would be represented as a general template and each subclass (a real time subclass, a Level 2 subclass, a Level 3 subclass and so on) would defer to the general implementation or override with its own implementation as needed. Then, each analysis engine would be represented by a specific implementation of a template. In terms of extensibility, this would allow a future designer to create unique reach state implementations which could be a hybrid of proven effective reach states used by HAMRR and new reach states.

Furthermore it is advised that these reach states should be moved into their own individual state objects. The main parent state class, would have a common function to process an incoming data frame. Then, each state would have their own rules to process that frame and determine if a specific state transition should occur. If a transition needs to happen, the specific state class would update the active state of the analysis engine, so that new data frames would be processed by that state object as well. In this way, each motion analysis engine does not have to be concerned with the specifics of a particular state, and thus, this variability is encapsulated. This would allow a future designer to extend these states or add their own, with minimal changes to the analysis engine.

6.1.6 Data Structures Should Anticipate Future Development Changes

A common software design theme discussed thus far has been to anticipate changes. This includes not only writing code that can be easily extended in the future for new applications (such as in the discussed example of the transportable cone), but also ensuring that the existing code can work with future changes to other components.

One of the areas where this idea is key is with the data objects. Through out the code (Sensing Plugin, Tangible Plugin, Motion Analysis Plugin and Adaptation Plugin) data is stored into data objects. These objects typically just hold properties for each value of the data object, with minimal other helper classes. One of the main set of helper classes is in the implementation of the NSCoding protocol. The NSCoding protocol provides methods that must be defined to save an instance of an object to file. The process of saving and loading an instance of a class to or from file is very straightforward, until the structure of the data class changes. The change in the data class is inevitable through out the design and testing process as new features could be added to or removed from the analysis. Therefore, legacy data objects suddenly don't work with newer code iterations.

As an example, the HAMRR Analysis program saves instances of HABaseTrial (and its child classes) to file in order to save all of the segmentation points and applied filters for a trial so that these features do not need to be re-identified in the future. Saving these features as a serialized object works because there is really not a need to edit any of these features out side of the context analysis program (in other words, for example, there wouldn't be a need to leverage an XML viewer to edit the location of segmentation points of a trial). Therefore, saving the features as a serialized object does not need to change. However, as the analysis program was being developed, it was also being used to start testing the analysis process to identify any problems during the development. As a result, new segmentation and filter features were being added and, as a result, made legacy serialized objects outdated. Therefore, as a result, older data objects could not be loaded, which even in the testing phase of the analysis program, required some segmentation to be completed again. This is not ideal for any iterative design work-flow.

Instead, along with the serialized object properties should be some encoded information representing the version of the serialized object. This feature should be checked first and a resulting decoder should be selected to unpack the data into an appropriate data object, with non-serialized features being initialized with default values. Then, when these data objects are saved, they could either be saved with the most recent encoder, or a legacy encoder so the data object could work with older versions. In the case of the HAMRR analysis program, there was no need to maintain a legacy program, so old data objects should be saved under the latest encoder so that future serialization problems are lessened.

6.1.7 The Adaptation Plugin Should Support the Hierarchy of User Interactions

As the main control component of the system, the Adaptation Plugin needs to support the interactions of the user who is controlling the system. This requires, as a designer, thinking about the high level interactions that a user wants and how lower level code components will perform the commands. The same methodology should be applied to the code. The high-level controller classes should just interact with a high level interface, or façade. This standardizes the details that are part of a large command (such as “Start Set”) and thus organizes the code into more stable components.

For example, currently, the ADMainController has a function called “runStartUpFunctions” that has a series of calls that need to be executed when HAMRR starts up, after all of the plugins have been loaded by DASH. This function:

1. Triggers the loading of a scenario file, if a scenario file is successfully loaded, the function calls continue.
2. Send an NSNotification to the Motion Analysis Plugin with the patient ID to load the calibration files for that subject
3. Query the database (Archiving Plugin) to see if this session was previously left incomplete. If so, return the set ID from which to resume.
4. Notify the Tangible Plugin to open the ports
5. Notify the Visual Feedback Plugin of the current number of sets in the session, to update the session progress bar shown to the patient.

Separating for a second some of the previously reviewed over-reaching functions of the ADMainController class, it will continue to be true that at system start up, a series of function calls to various plugins will be necessary. However, the variability of these steps and their specific implementation (which will be application specific, and thus, a good candidate for encapsulation) should not be part of the main controller class. Rather, a generalized Façade should be created such that macrocommands (such as Start Session) can be called by the ADMainController class, but the specific implementation and calls to other plugins, can be handled by a separate class. This further strengths the modularity. A future designer looking to extend the code knows that a "Start Session" function is crucial to implement, but can update the specific calls within an encapsulated class.

The same methodology could be applied to the function that is called at the start of a set: "startSet". Currently this function

1. Ensures there is a valid Session and Set ID for the next set
2. Checks to see if a specific demo needs to be played, and if so, triggers that demo (Which is a protocol connected with the Visual Feedback plugin)
3. If necessary, it triggers a reminder of the starting location of the transportable object, as well as the directionality of a multi-object task.
4. Calls a series of specific protocols to different plugins (Tangible, Audio Feedback, Visual Feedback, Motion Analysis and Archiving) to setup prior to running a live set
5. Starts the video recording
6. Starts the set (via a protocol connected to Motion Analysis)

Similar to the "runStartUpFunctions" example, here are a series of steps that are application specific, but represent a macro-command that will be important across applications ("What features need to be configured or ran before a live interactive set begins). Again, ADMainController shouldn't have to implement these specific calls. Rather, these specific calls should be abstracted to the Façade class under a more general "Start Set" function. Therefore, the ADMainController stays more modular. It does not need to know what specifically constitutes starting a set, but rather that starting a set is an important step in the interaction.

6.1.8 Code Modularity Within a Plugin Requires Inner Class Communication Optimizations

Similar to the previous discussion on standardizing the style of communication between plugins, a similar review is necessary of the communication within a plugin. Just as the previous mentioned suggestions for between plugin communications allow for large scale modularity such that plugins remain more separate and contained, modularity within plugins is equally important for code extensibility. Thus communication between classes and the flow of function calls should be considered closely.

Using the Adaptation Plugin as an example, there are currently circular function calls, in which a high level class depends on a low level class, which in turn, depends on a high level class. This entangles functionality quickly and makes debugging and extensions difficult.

For example, when the ADMainController executes “runStartUpFunctions,” as previously discussed, it makes a call to ADScenarioController to attempt to load a new scenario. At the end of this function, a call is made back to ADMainController to initialize the archiving file (which notifies the Archiving Plugin of the current Patient Id, Session Id). This leads to a problem called dependency rot. In this example, a high level component (ADMainController) depends on a low level component (ADScenarioController), which in turn depends on a high level component (ADMainController). If any changes were made to the higher level class (ADMainController) it could unknowingly have consequences for higher level classes in more indirect ways. Plus, the ADScenarioController now requires that an instance of ADMainController exists. While this many not be an issue in some applications, in this specific example, a low level class (ADScenarioController) is adding an additional constraint that not only is the high level class required, but the implementation of a specific function is required.

To avoid this problem in the future, the high level classes should dictate when information or algorithms are needed from low-level classes. Therefore, in the previous example, instead of the ADScenarioController calling back to the ADMainController to initialize the archiving, the ADMainController should know to initialize the archiving if a scenario file was successfully loaded. As previously described, this is another step that should be wrapped up within the macro-call façade. Thus, two problems are eliminated simultaneously: more variable steps are encapsulated in an interface and low level classes have less dependency on high level classes.

6.1.9 Data File Types Should be Easily Accessible and Reliably Organized

As previously discussed, two of the main users of INR system are the therapists and system designers. Both of these users will require access to the data after an interactive session is completed. They may also wish to script automated scenario files or make changes to the sensitivity of the training. In all of these cases, data files representing these details are required for the system to load. For the ease of use of the therapist and design team, these files should be easily accessible and organized.

As has been reviewed, HAMRR currently uses five different file types:

1. XML – All scenario files are composed as XML files. The HAMRR Composer writes scenarios to XML files and the Adaptation Plugin loads scenarios from XML files.
2. Serialized Objects – All marker calibration information is saved as a serialized object (from the Motion Analysis Plugin) and trial segmentation and applied filter information is saved as a serialized object (from the HAMRR Analysis program).
3. pLists – pLists are used throughout the Visual Feedback plugin to load and play appropriate content, especially for the on screen demos.

4. SQLite database – The Archiving Plugin saves all data frames and resulting analysis, as well as calibrations and control information (ADScenarioData information) to a table separated database.
5. Text Files - All of the sensitivities (for tasks and feedback parameters) as well as task reference trajectory information is loaded from text files.

For most of these file types, there was a specific reason for its selection:

1. XML allows for easy review of the structure of the data, which made sense for the structured data information from ADScenarioData. In addition, if the scenario paths need to be edited by a therapist in future applications, multiple external programs can edit XML files.
2. Serialized objects don't allow for easy review outside of the program, but are reliable to save from the program. Therefore, calibration data and trial segmentation points, which were not needed outside of the program, could be saved and loaded safely and quickly.
3. pLists, like XML files, provided an easily viewed and editable structure to data.
4. SQLite provides a very organized and reliable way to save data, which is crucial for the data results of the system in use. Similar to XML and pLists, SQLite is easily viewable outside of the HAMRR program by external programs.

As can be seen, these file types offer a fair amount of flexibility, however the one excluded file type (text files) can be problematic. Text files, on their surface, are easy to understand. They can be filled out, separated with a character (such as a comma or tab) and easily loaded and parsed by a program. While this was advantageous for me in quickly being able to load sensitivity files, text files can provide problems if not composed with the correct structure. For example, if specific values are space separated and the program is expecting tab separation, the incoming data parse will be incorrect. As another example, sometimes blank lines at the end of a file can also create problems in parsing the incoming values. Therefore, while they initially seemed like the easiest way to load information (such that it could be edited outside of the program), it is not the most stable solution. Rather, for system control information, pLists would be a better solution. These file types can still readily be editable outside of the HAMRR program, but they inherently have a more standardized, clean structure than text files.

6.1.10 Design for System Assessment to Provide Information for Future Design Iterations

While it is incredibly important that patient data is archived in a manner that is robust and in an easily accessible format, the same approach should be taken to system state information. Because of the strength and ease of extensibility of the database structure, more information should be archived that can allow for not only later analysis of patient performance, but also of system performance. In an adaptive system, many times changes to protocol will be made in response to errors found or sensitivity tweaks that were not previously encountered. This requires making changes to the state of the system for that particular user, but these changes may not be applicable for all users of the system. These variable changes in system state are very important for system designers to assess the performance of the system.

Within the Adaptation Plugin, a component was developed to write simple log messages to file. However, this component was not developed very far as development priorities were pushed in other directions. Related to the previous discussion of having a standardized way for plugins to report errors to the control plugin, the control plugin in turn should also have a way to save these reported errors to a log

6.2 System Architecture Improvements

6.2.1 Code Architecture Could be Further Improved with Standardized Plugin Templates.

From my work in developing many of the plugins that were used by HAMRR, it was clear that the initial functional decomposition design was quite comprehensive. While some additions were needed within plugin, none of the additions required the inclusion of a new plugin or splitting a plugin into two functionally separate plugins.

Therefore, the overall hierarchy architecture suggested should be utilized further. And, in order again to support long term code modularity and extensibility by retaining what was found to be successful of the current architecture, plugin templates should be created for future developers.

Inputs

Each Input plugin will represent taking raw sensor data from a varying number of pieces of hardware, possibly providing some low level data filtering and analysis, and proving access to this processed data for other plugins.

The input plugin will have some dependence on what type of hardware sensor is being used (in terms of the type of data connection and the style and format of data sent from the hardware). In this case, if the developer wants to user serial inputs, it would be beneficial to have an API to create a new serial input within the existing Tangible Plugin structure.

The low level analysis and data parsing will be hardware specific as well. Extensible interfaces should be created for both so the developer has the ability to use an exiting data parsing method, or implement their own.

Finally, the template should provide a standardized way to report (via NSNotification) that data is ready to be processed by other plugins. Frame rates will differ for hardware, therefore it will be up to the designer how often data is provided to the rest of the system. Along with the NSNotification, should be protocols for accessing the data contained within, along the examples provided in section A1. Therefore, this input can be used by any other plugin without significant modification of the code in other plugins. Examples are shown in Figures 24 and 25.

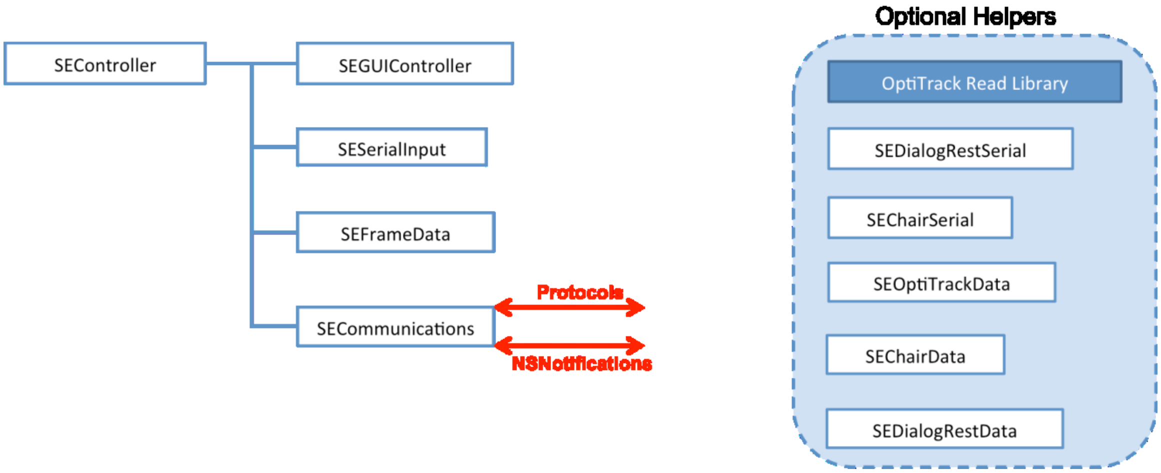


Figure 24 – Example Sensing Plugin template

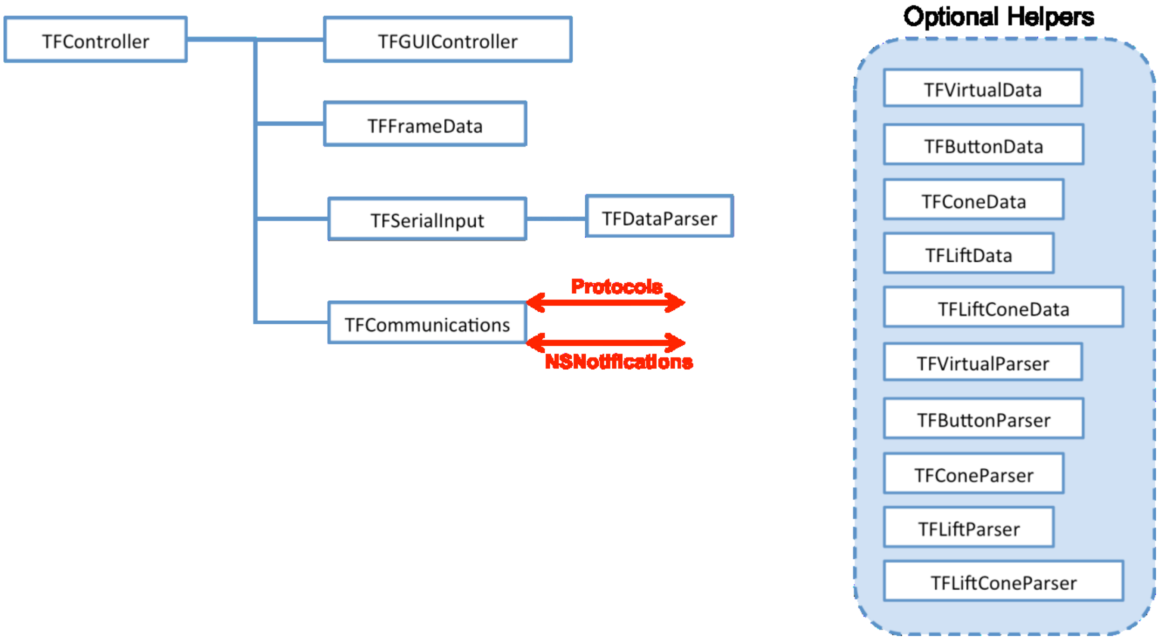


Figure 25 – Example Tangible Plugin template

Analysis

Each analysis plugin will represent taking data objects from the Inputs plugins, conducting detailed analysis on the data within, and providing access to the resulting analysis data to other plugins.

The Analysis template should have a class for receiving the standardized data objects. It will be up to the designer to determine what to do with the received objects.

A template for HAMRR development should provide an API for utilizing the existing analysis. This API would provide access to an interactive reach state as well as the results of various analysis types. It would also allow for a designer to create their own analysis type by extending reach state templates (as previously described).

Finally, the analysis plugin template should provide a standardized data output object, similar to the input data object. A standard NSNotification would be provided, with necessary protocols to access data values. Again, it would be up to the developer to determine how often these data object notifications would be sent out from the analysis plugin. An example of this template can be seen in Figure 26.

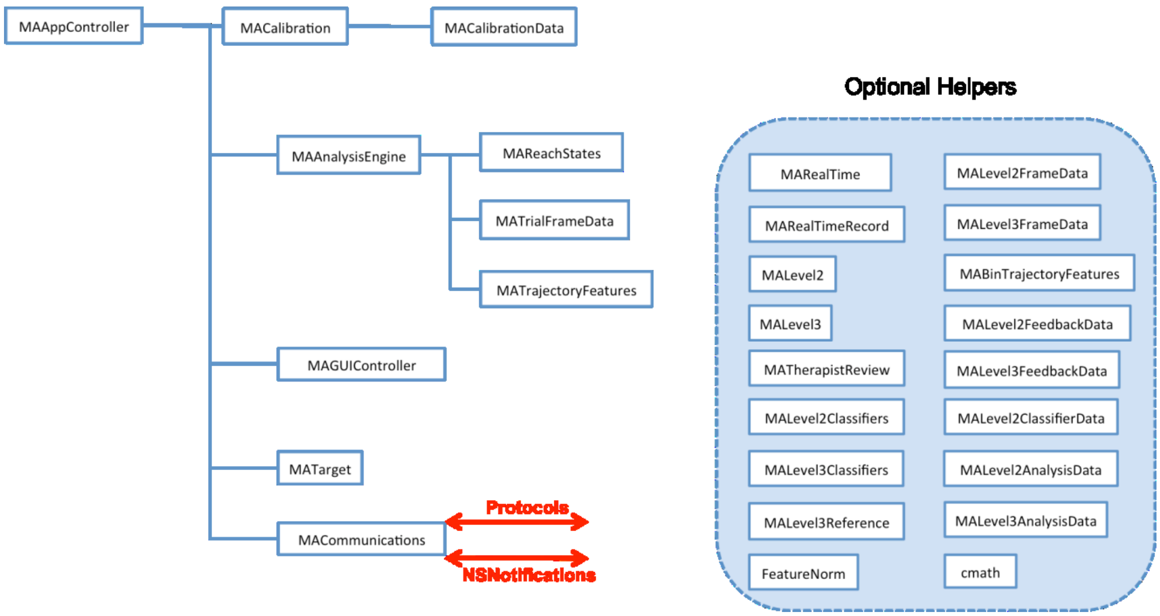


Figure 26 – Example Motion Analysis Plugin template

Outputs

Each output plugin will represent taking resulting data analysis and providing some feedback based on those results. While I will not discuss in detail what a template might look like for a feedback engine (since I was not involved in the feedback plugin development), I will leave my comments to interfacing an output plugin with the rest of the program.

Similar to how an Analysis Plugin handles the notification from and Input Plugin, an Output Plugin would receive a notification from the Analysis plugin. Therefore, to an Output plugin template, a general class should be provided to receive the Analysis notifications. Once again, in a similar fashion to the Analysis plugin, the

Output plugin would have generalized protocols to get access to the results of the analysis without having to know the structure of the data.

All of these suggestions would also be true for implementing an archiving output plugin. It would received generalized notifications (a template should once again provide a class for receiving these notifications), access the updated data via protocols, and save the data to an SQLite database. An API here would be very helpful to allow for easy creation of and access to a database file. An example of the template can be seen in Figure 27.

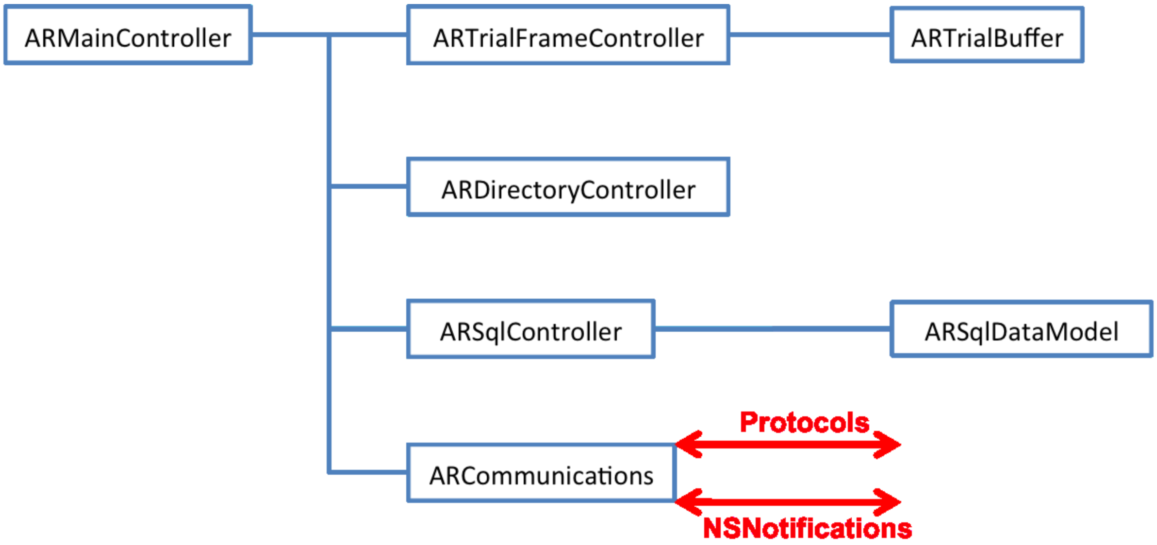


Figure 27 – Example of Archiving Plugin template

Control

The control plugin represents a master interface for the whole program. It should have standardized protocols for each style of plugin (an input protocol, to be implemented by input plugins, an analysis protocol for analysis plugins and an output protocol for output plugins). Depending on the application, there many need to be some more differentiation of the protocols, but this would provide a basic start. In addition, related to the control plugin, each other type of plugin should provide a standardized method to report errors to the control plugin (as previously discussed). Accordingly, the template should provide a class to implement protocols from other plugins as well as a class to call protocols to be implemented by other plugins. The template should also provide general sensitivity data object, similar to the data and analysis object, so there is a standardized way to access sensitivity and task data. Finally, a generalized GUI controller class should also be provided, with access to key sensitivity, task and error information. The developed could then use this provided information in any way desired. An example of this template can be seen in Figure 28.

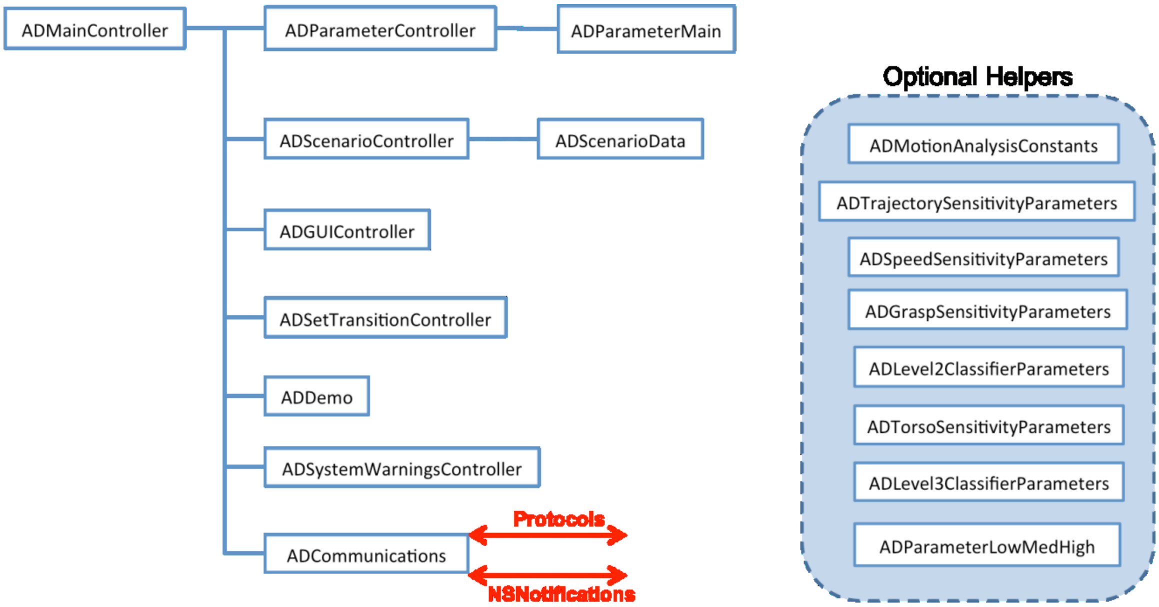


Figure 28 – Example of the Adaptation Plugin template

By implementing templates, or APIs of existing plugin implementations, quick extensions to code are possible. The template provides a suggestion for overall system development, plus it provides a standardized (and thus stable) communication method between plugins. Plugin communication was something that needed constant testing during the development of the system. With the formation of templates and the standardization of the communication between plugins, this testing would be minimized. Developers could add content with the assurance of the form of the incoming and outgoing data. Within each plugin, by providing an API, we are able to leverage tested and usable code. When building the HAMRR Analysis program, I wanted to use a fair amount of the analysis functions from the hospital system, since these methods have been substantially tested and validated. However, these analysis algorithms expected a specific data type implementation, which was not used in HAMRR. Therefore, substantial time and testing was required to make sure the analysis algorithms were implemented correctly. If an API existed for AMRR analysis, this inclusion of analysis methods would have been smoother with less uncertainty if prove analysis would work in a new context. Therefore, future developers should be able to leverage the existing input, analysis, output and control code that has been tested and validated within the HAMRR pilot study.

Related to the idea of further modulation and the creation of templates, the modularity of code does not necessarily mandate the utilization of DASH. The selection of DASH, as referenced earlier, was done due to its ability to stably and reliably handle visual feedback development. It allowed for clean ways to interface with OpenGL drawing. In addition, beyond this capability, it kept all the same functionality of Objective-C code. The fact that DASH is structured to run multiple plugins of code certainly reinforced the design idea to functionally decompose the HAMRR system into modular units. In addition, by having macro-level functional decomposition based plugins, it ensured less overlap in people's work. This working within a plugin usually had limited chances to negatively impact other people's code. Of course plugin communication issues could arise, but very rarely were team members editing the same classes simultaneously.

6.3 System Hardware Improvements

6.3.1 Move Away from Reflective Marker Tracking, Especially When Camera Numbers are Reduced

Now that a substantial amount of data has been collected with eight subjects, it was shown that the end point is still a very crucial input as many levels of data could be extracted from this one input. However, an OptiTrack system is not a practical idea for moving forward with home based sensing solutions. It was a design selection made to keep the system development moving forward with the compromise that for the first iteration of HAMRR, it would provide the most reliable data for the team.

However, for the reasons identified in the Results chapter, this sensing setup should be quickly replaced. Not only can it lead to some of the noise previously identified, but also it is a heavily proprietary solution that requires consistent upgrades (requiring continual financial commitment) to use the latest software capabilities of the tracking system.

Therefore, computer vision solutions that will not be interrupted by, or can compensate for, occlusion will be key. The hybrid solution of introducing sensing in the chair really only provides a basic sense of how a patient is sitting in a chair. As was shown, it could be used as a very basic indicator to see if a patient began a reach with compensation (if compensation is to be defined as torso movement away from the back of the chair). However, significant compensation will take a patient off of the chair (as was seen consistently in more impaired subjects), and thus eliminate the utility of these sensors. Plus it requires instrumenting another aspect of someone's home, which is not ideal. Thus solutions for torso tracking that can utilize a single camera setup would be optimal.

6.4 User Experience Improvements

6.4.1 The Design Team Should Have Remote Access to the System

In order to support the iterative design workflow framework, the design team will need to access the software of the system frequently to make adjustments and fixes in response to needs that arise. For the majority of the HAMRR testing protocol one of the main system designers was on site to oversee the study. Each person had a deep knowledge of how the system worked, including its code, and could make updates in a very straightforward manner with the system. However, when the system was controlled by a therapist for two subjects, the previously described workflow was invalid. Further more, expecting the therapist to make code updates is incorrect and does not benefit the needs of the therapist as a user of the system (monitoring the performance of the patient).

During the therapist led portion of the study, when issues or suggestions arose, making changes to the system required e-mailing project files that then the therapist needed to build or put in the correct directory while one of the design team members talked through the process. This led to confusion and consumed a fair amount of time to complete. One solution to this would be to leverage a remote desktop access such that a system designer could log in and make the necessary changes in-between patient sessions. Another possibility would be to create a network access point within the system itself along with a remote client program to interface with the system. This solution, however, would require more time to develop the client and may only work as a debugging tool to remotely view the state of different program components.

Providing remote access could also be a way for the design team to get data files from patient sessions. In the HAMRR study, the anonymous data files were added to a password protected Dropbox account. However, this required the therapist to manually add the files to the Dropbox account. While this procedure overall was not very difficult for the therapist to complete, it's still an added step that should be removed. It would be better, for example, for the program to automatically add saved database files to the Dropbox directory. Or, depending on the style of remote access developed, the design team could log into the system and pull the data manually.

Due to network restrictions at the site, we were not able to smoothly implement a remote desktop solution as it was felt the therapist would still need to make changes to system settings to establish the remote desktop connection. However, in retrospect, these changes required (cycling on and off the wireless connection) would be much simpler than any other procedure. Therapist would have to turn off the wireless (one step), answer an incoming call from a remote desktop client (one step) and then turn off the wireless once the updates were complete (inverse of first step).

6.4.2 The System Should Incorporate Standardized System Error Reporting for the Therapist and System Designers

Designing an experimental complex system that leverages iterative design cycles implies multiple short-term tests of integrated components, each with their own respective development timeline. Therefore, the design team always strives to minimize system bugs, they can be inevitable. As a result, designs should be put in place to maximize the information available to characterize a problem. In a system like HAMRR, this can be further complicated because an error can come from any one of the plugins. Also, just because an error occurred, does not mean that all other parts of the system will be impacted. Thus, portions of the live interactive use of the system may appear fine, while there is a larger issue building underneath.

Therefore, all problems and system errors need to be easily and clearly brought to the attention of the testing design team, as well as the user (including the system designer, therapist and patient). Currently, as previously discussed, there is a framework in place to report the patient when noise enters the OptiTrack data stream and how to correct the problem. However, for more internal system problems, a similar approach should be taken.

Similar to standardizing communication between plugins, in future designs, there should be a standard way to report errors. The Control plugin (in this case the Adaptation Plugin) should be configured to be the default handler of all error messages and all the other plugins should have standardized ways to report errors. For example, if noise is found in the incoming data signal, the Sensing Plugin should have a set way to inform the Adaptation Plugin that a problem is occurring. Then, the Control Plugin should have a manner in which to respond to the events, which should result in an instructional message on the screen for the user or therapist to respond to. If the Archiving Plugin had an issue in saving the data to file, the

Adaptation Plugin should be notified such that the system controller could assess if the set needs to be completed again. In the redesign of future plugins, this should be a mandatory piece. Along with the functional decomposition, should be a parallel design of what problems may arise from each function, and which of these problems need to be reported up to the main control plugin. In the case of HAMRR, the Adaptation Plugin could handle these incoming errors and functionality could be created to handle each differently from stopping a set and creating an onscreen prompt to silently logging the issue to file in the background. In either case, each plugin should be aware of possible problems and should notify any control plugin of this issue.

6.4.3 Don't Compromise on Core Principles That Can Affect the Patient Therapy Experience

In the previously discussed example of quickly developing and integrating a new transportable cone object for smaller hand apertures, an iterative design approach was used: a perfect solution could not be achieved in the development time allowed, so priorities had to be identified such that the system could keep working and collecting informative data. In the case of the transportable cone, this meant removing the data streaming from the object as it was interfering with the stability of turning on and off and IR light at the top of the cone. It was decided that tracking the location of the object computationally (and therefore, not requiring the therapist to focus on this) was more important than training a grasp. However, this created a discontinuity in experience.

The cone and transportable cone are nearly the same in appearance, and thus it would be expected by the user that their behaviors are the same within the context of the training task. However, this became clearly identified as not true by the patients who used this object. In the case of the stationary cone, task completion is determined by a combination of factors including the pressure exerted on the object. However, in the case of the transportable cone, this data was not available, and thus task completion was defined as proximity of the hand to the object. This introduced some confusion, both to the patient and therapist, about why a success cue was given so quickly for one object and not the other. There was a discontinuity in the training experience. However, it was known from the past AMRR study that tangible data is crucial for assessing task completion. This is especially true for more impaired subjects, who were the users that were in mind for the design of this new object. Therefore, an incorrect decision was made in this iterative process, and more emphasis should have been placed in ensuring the object could provide some reliable tangible data for assessing basic task completion.

Therefore, in the future, some basic indication of physical interaction should always be a requirement for task success. It doesn't necessarily need to be a complex, grasp-classifying model, but it should provide, at minimum, a basic, binary indication of grasp. Therefore, as the transportable cone was initially designed, two simple sensors to detect opposable grasp would be an adequate starting point. Due to a late change in the electronics design as well as a lack of a full testing environment, this wireless protocol could not be tested fully, and thus needed to be removed. Looking forward, this idea should be revisited and refined for integration with the transportable cone.

6.4.4 Design a GUI to Support the User Experience of the Therapist, a Primary User of the System

While the system was overall successful in running multiple patients through the protocol, it became a challenge when trying to train other therapists to use the system. The main control interface, implemented in the Adaptation Plugin, while allowing access to all of the necessary sensitivity features, did not do this in an efficient way. The design of the interface did not consider the usage needs or time constraints of a therapist. Instead the interface designed for a system designer to use who had full knowledge of all of the system details. However, as previously discussed, the therapist needs to be considered a primary user of the system.

A key user experience observation is that a physical therapist's attention needs to be focused on the patient and the subtleties of their movement; not on how the system is performing and what buttons need to be pressed next to continue the therapy protocol. In the future, the GUI should be completely redesigned to maximize limited allowable focus and minimal required knowledge to run the system.

6.4.4.1 Create a Functional Decomposition Hierarchy

Similar to the approach that was taken to designing the overall system architecture and identifying where individual plugins could be separated, a similar approach to the GUI should be utilized. When designing the next interface, significant time should be given to collaborate with therapists (especially those who have already used the system) to understand what information is absolutely essential, and which controls are not. Some of this information has been gathered from the second round of patient sessions at Emory. My suggestion is to break down the controls into tiers of necessity: Real-time controls, Debug Controls and Set Controls.

First of all, the Real Time controls should be reduced to three categories: “What am I running?”, “What is the state of the data?”, and master start and stop controls. These three themes seemed to be the most important during therapist ran sessions. “What am I running” refers to understanding where in the protocol the patient currently is as well as which feedback streams are currently on and their sensitivity. Secondly, “What is the state of the data” refers to clear indications of the noise level of the incoming data and clear instructions of what to do if there is a problem with the incoming data. Finally, there needs to be quick access to a start and stop control. If the patient needs to break during a session or a hardware or software bug comes up and needs to be solved, these controls are necessary.

The second tier of controls should be Debug Controls. This represents a set of controls that only needs to be accessed if something goes wrong with the system. Within this category should be the demo controls (skip demos, turn off demos, only use demo prompts). On and off controls for using the wristband sensor and using torso feedback should also be included. The selection of these components comes from observations of therapist use of the system. If DASH needed to be restarted, therapists need easy access to skipping unnecessary demos and getting quickly to the correct set in the protocol to quickly recover from any problems. Secondly, when there were problems with the interactive sets, they usually were due to noise in the rest calibration and wrist pad sensor or noise in the torso calibration, using the markers or chair data. Therapist should have quick access to turning these off if they are incorrectly interfering with the interactive sessions.

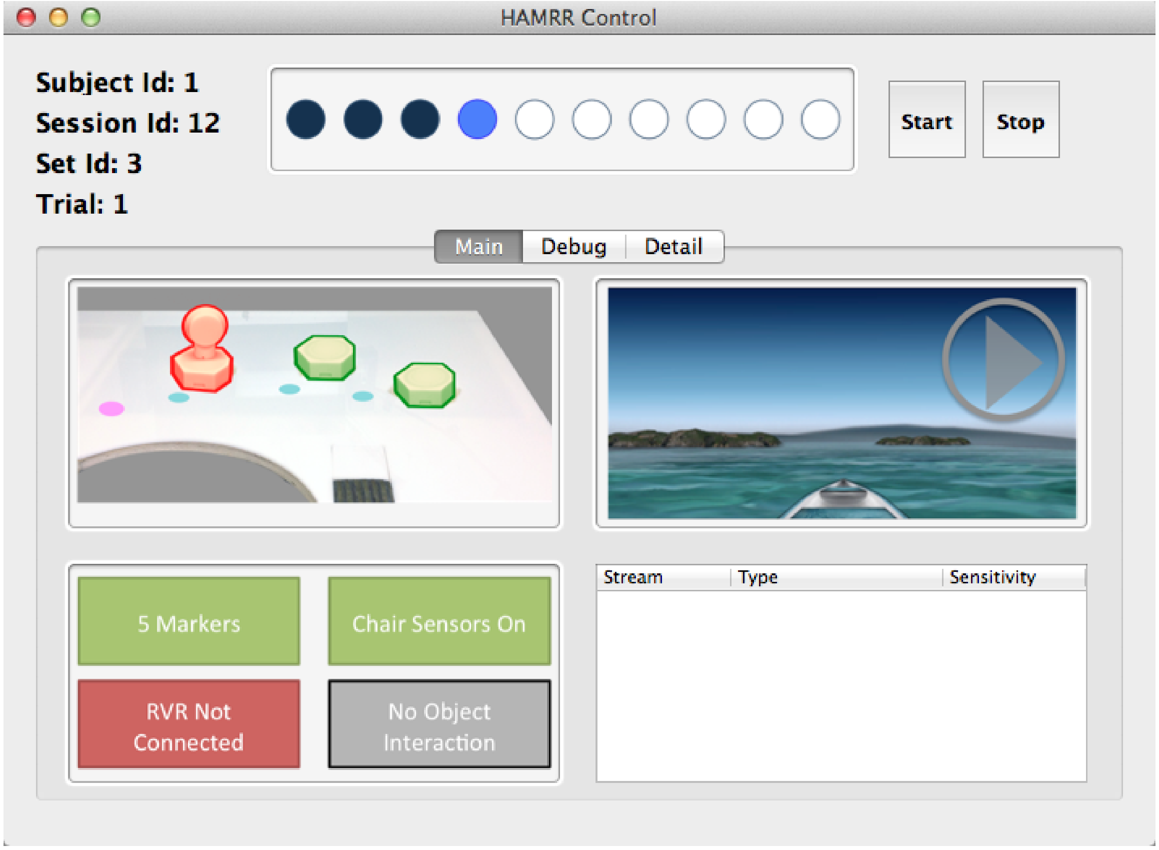
Finally, in the third tier are the detailed Set controls. These are all the parameters that the Motion Analysis, Visual Feedback and Audio Feedback plugin use during an interactive set. These need to be grouped into intuitive categories, with more detailed and clear labels than what the current HAMRR GUI window provides. The labels need to be presented using language and units that are quickly intelligible by a controlling therapist.

6.4.4.2 Address the Needs of the User in a Form That is Efficient for That Specific User

Significant time and design went in to building feedback environments that could efficiently convey complex yet structured information to the patient on their performance of the reach. Similar consideration needs to be given the therapist (as a key user of the system) and how information can be efficiently presented to them.

When designing the proposed new tiers of controls, there should also be an emphasis to use a "Show, Don't Tell" approach to the design. Instead of an interface of just numbers and pull down selectors, parameters and their resulting effects should be shown in a visual or auditory way to directly demonstrate what each parameter does. There should be an interactive set preview window that will quickly play back all of the feedback (tangible, audio and visual) so that the therapist can directly see what the upcoming set will look like. Currently, the therapist has to make another mapping transformation in their mind so the sequence progresses from: physical space consideration to abstract parameter in the system to resulting effect on the interactive set. The middleman here should be removed so the therapist can intuitively change sets. This would also help empower the therapist to diagnose problems should they occur.

In conjunction with this "Show, Don't Tell" approach to the task and feedback environment settings, this should also be applied to error reporting. When the system was tested under therapist control, it still required a system designer to diagnose problems remotely over a Skype call. While this may be unavoidable for some detailed code bugs, this should not be the case for higher-level sensing or functionality issues. The GUI interface needs to clearly communicate what the error in the interaction is, as well was, suggestions for how to diagnose the problem further or directly fix the problem. Over time, the therapist who ran HAMRR was able to identify some problems automatically, but a GUI that could bootstrap this process and have therapists feeling further empowered to use the system, by reducing the amount of troubleshooting required on their part, would be beneficial. An example of a possible new mock up can be seen in Figure 29.



Subject Id: 1
Session Id: 12
Set Id: 3
Trial: 1

Figure 29. Example updated control GUI

6.4.4.3 Design Interfaces with Continuity in Mind by Creating Design Guidelines

As previously mentioned, it is important to maintain a uniformity in experience across systems. This is especially true for INR, where the likelihood of one system being able to address all training needs across systems is very low. Therefore, both the physical interface and virtual (GUI and patient experience) interface should provide a continuous experience across systems. There are many examples of this work used in other applications, including Apple where a set of human interface design guidelines have been proposed for the apps installed on their devices.[86]

While it will not be described in detail here (as it is beyond the scope of my work), many of the feedback design principles that were validated in AMRR were continued in HAMRR. Therefore, if AMRR and HAMRR were to be tested in a continuum, it is likely that the user would be able to transfer some experience of the feedback and tasks from one system to the other. However, the testing of AMRR and HAMRR were separated by a significant amount of time. They were also tested with different sites. Therefore, continuous experience in terms of the user interface for the system controls was not considered at all. However, moving forward, it will be important to consider the interface that the therapist uses to control the system and how similar design choices can be made for other systems. As has been previously described, the resources available to therapists are limited, and therefore a continuous experience would limit the amount of time needed for training. Similarly, the interface with system controls by the patient (table buttons and demo buttons) should be considered moving forward as well to see if these interactions are valuable to re-implement in future systems. Therefore, overall, it may be beneficial for the design team to identify design guidelines before future development to ensure that this continuity of experience is not lost.

6.4.5 Enhance the Patient Experience of Controlling the System and its Setup

In order for HAMRR to be closer to a system that can be reliably implemented in the home, the start up and shut down procedure needs to be streamlined. This process was initially developed such that by turning on the computer, the necessary programs would start-up automatically and begin the appropriate session. However, this code was ultimately not used as it was determined better for the system to be tested in a semi-supervised clinical environment first before going to the home. However, the user experience of a simple system startup and shutdown should explored more in the future, as it would not only benefit the patient autonomy using the system, but would also improve the user experience of the therapist using the system.

6.5 Support the Human Experience and Allow the User to Provide Input to the Experience Narrative (Reach for the Higher Levels of the Hierarchy of Needs)

In my observations of the patients using the system, there was noticed to be a distinct lack of human presence, which seemed to be a detriment. As a system controller and tester, my role required me to remain mostly invisible during the sessions. The protocol was structured to see how easily the patient could setup and run the system on their own. I would intervene only if I saw something going wrong with the system that was not possible to be fixed by the patient. However, it was clear with some of the patients I saw that the interaction with the computer wasn't enough. As experience design would suggest, the patients come in as anyone would to the study: with their own concerns, stories, questions and experiences. Currently, the system was setup such that the interactions were very controlled, and therefore limited. While variables needed to be controlled in the first implementation of a semi-supervised system, this user experience is not ideal in moving the system forward long term. This is not to suggest that teleconferencing is the solution. However, the abstract therapy environment and narrative needs to be grounded.

6.5.1 Provide the Patient with Interfaces for Reflection

The patient should be encouraged to reflect on long-term progression with the system and where challenges occurred, and how they were overcome. I think there is an opportunity, with the iPad app developed by Nicole Lehrer, to make programs not only for therapist review, but also for patients to log their own progression in interesting ways. The disparity of the little one-hour notch in someone's 24-hour day that they use the system needs to be acknowledged. There are so many contexts that could be helpful for the patient to reflect on during their use of the system and improvement in ability. A reflection interface could also be a collaborative tool to use with friends and family as well to appreciate their sense of progression of the patient and provide another means of motivation and support.

6.5.2 Support the Ability of the Patients to Ultimately Create Their Own Training Experiences

An additional part to the solution of supporting the human experience more is to review how the system fulfills (or doesn't) the Hierarchy of Needs [22]. The Hierarchy of Needs was originally a theory of factors for human motivation created by Abraham Maslow. The model provides a proposed ranking of needs that humans strive for, and they a structured in such a way that the lower level needs must be reached first before striving for the higher level needs. This model provided a similar way to think about system design. Analogues for the human psychology model can be found, starting at the lowest level (Functionality) and culminating in the highest level (Creativity).

One of the limitations of HAMRR, was it did not consider the whole hierarchy. As the model suggests, if the system is not functional, then there is no need to consider any higher level aspects. HAMRR, as previously discussed, was design with functionality and reliability at the forefront and as a result impacted many decisions on the hardware, software and experience. As a result of an emphasis of functionality and reliability, less importance was given to usability. Some core usability aspects were considered (such as system setup), but they were created under the assumption that a system controller would be available to provided assistance if needed. However, the highest levels of the hierarchy (Proficiency and Creativity) were considered to an even lesser extent. Some proficiency was considered through the design of the protocols to assist patients in becoming more adept at completing complex tasks. However, limited considerations were given to help connect this gained proficiency with ADLs, which is where I propose the "Creativity" level lies.

A user of the system should be provided with opportunities for creativity. With proficiency in the system, the patient should be allowed to use the system provided tools to create their own training experiences. Due to the proposed long term use of the system, proficiency will be likely in many users (and signs of proficiency in HAMRR were already noticed with the more mild impaired subjects). Therefore, with this experience gained in the system, the patient should be empowered to set their own goals and thus impact their daily life in the manner they desire.

6.6 Considerations for Adjustments to Existing Therapy Protocols

What follows is a discussion of suggestions for changes to the path protocols and where automated adaptation might fit in this landscape. The following also provides many examples of the process of iterative design. Decisions with underlying constraints were made based on available information and the desire to progress the system forward to collect data with stroke patients, and therefore, learn more about INR design. As will be shown, in each case a decision was made and, as a result of making that decision and collecting data, new design concepts can be presented now for future work that may not have been possible during the initial design of HAMRR.

6.6.1 Introduce Further Variation to Increase Challenge

The current protocol paths were composed under the constraint that challenge could be introduced through target type, target location and sensitivity of feedback. This was a design decision to limit the variables to introduce challenge and progress the development forward so more data could be collected. It has been found that these paths did not challenge the highest performing patients enough. However, preliminarily, the paths seem to successfully connect all the levels of feedback, as subjects were able to understand their meaning based on responses to a questionnaire provided to the subjects. Therefore, additional dimensions in which challenge can be introduced should be identified within the existing paths, as a starting point. One of these dimensions should be randomization. Currently, the patient experiences sets of repetitive reaches, meaning that within a set, the five reaches are exactly the same. This is crucial early on in the training to build motor learning skills and build a relationship with the feedback environments. However, for the more advanced users, randomization within a set could offer a bit more challenge. An example of this would be to have every trial randomly select one of the objects in the table for the next reach. This could be limited to a certain type of object or range across multiple types. Later on in the training it would be more of a challenge to have a patient form a new motor plan on demand based upon a target object the system randomly suggests. This randomization may need some structure so that it conformed to a pre-determined overall number of reaches to each target for a given session.

6.6.2 More Impaired Participants Require a Redesign of Task Sequence and Training Dosage

In addition to thinking about how scenarios could be increased in difficulty, considerations need to be given to how scenarios could be made easier as well for more impaired subjects. It was observed with one of the patients that 100 reaches is really a challenge to complete in an hour of time, and many times fatigue would set in halfway through the scenario. Therefore, shorter scenarios may need to be considered in the future. Similarly, the most challenging tasks of a scenario should not be held exclusively till the end of the scenario. While it is good to start a day of training with easy tasks to build confidence and work towards more complex activity, scenarios need to consider that fatigue is increasing as the scenario progresses. Therefore, some complex tasks should be introduced early in the scenario as well to have the patient attempt the tasks with minimal fatigue. This may help build some confidence in being able to complete the task. Also, by not weighting the end of sessions with complex tasks, the patient may leave a session with more confidence and optimism if they are provided the opportunity to reinforce the progress they have made with easier tasks.

6.6.3 Automated Instructions that Facilitate Training Should Vary Just Supervision Should Vary

As presented earlier, in INR design the dimension of supervision should vary such that a user can transition from a fully supervised clinical space to a non-supervised home environment. In a similar way, the automated demos that are presented to the user should also evolve over time. For the first iteration, the design of the demos sided cautiously in the direction of repeating things multiple times. It was the first time we tested the automated demos with patient populations, and we were unsure what would be clear and what would be more confusing, especially given the vast difference between design team development of the system and patient use of the system. Overall, with more advanced users it became clear that the instructions were boring. So much so, that some would not pay attention to the demo, and as a result do the first reach or two incorrectly. For the more advanced users, the number of demos and prompts should reduce and change. The instructions could be mixed up, for example, where one set provides a text prompt, and another set provides a short visual prompt in the form of the media feedback we want to see their reaches result in (like a straight path of rocks). By varying the text instructions with media prompts, this might induce some problem solving and keep advanced users from getting bored.

6.6.4 Maintain an Introductory Set of Sessions Under Full Supervision

The utilization of the first three training sessions was very beneficial. It offered a transition from following a therapist's instructions to interacting with an automated system. Not only did this allow the patient to feel a little more comfortable in interacting with a new system, but it also provided the therapist an opportunity to see how the system was reacting to their movement.

6.6.5 Sensitivity Values Should be Re-Evaluated with the Existing Set of Collected Data

The sensitivity values should be revisited for tasks. In the process of iterative design of the system, the initial values for all of HAMRR's sensitivities came from the tested values of AMRR. From here, adjustments were made as needed based on both non-impaired and impaired testing. As a starting point, only a high and low sensitivity value were created for each system parameter. During the pilot test, it was observed that the low and high sensitivity are still too low for really high functioning patients. Therefore, through experimentation and further review of the collected data, more sensitive parameters should be identified to create a new higher sensitivity setting. Also, the creation of a medium sensitivity may be advisable in conjunction with the protocol adjustments previously suggested for more impaired patients.

6.6.6 Complex Tasks Should Maintain Coherence of Experience of Integrating Components

The complex task of lifting and transporting an object against gravity was the most complex task provided by the training protocols. In an attempt to break this task into training components, I created training sequences that divided the reach, grasp and lift phase of the overall task from the actual transport portion of the task. The idea behind this design was to have the patient become comfortable with grasping and lifting an object with weight before moving the object across the table.

During the pilot study, it was noticed that separating the lift training on its own within the protocol did not seem to provide the desired effect. It was observed that some patients would break up the phases of a transport task by hesitating when grasping the object and lifting it in an artificial way to the height to which they were trained to raise the object up. It would probably make more sense to remove the isolated lift tasks and instead introduce a new challenge dimension of progressively more difficult transportable objects. Patients could begin with a lightweight transportable cone (such as the current transportable cone) and progress to heavier and wider hand aperture grasp objects. In this manner, complex tasks could still be introduced early on in the training to maintain the feedback and task fading, but the improper training of an isolated lift would be removed. If a better solution was determined for training the lift exclusively, that trained a grasp as part of a fluid lift movement, that task could be substituted in as well.

6.7 Automated Adaptation Should be Included

As was shown in the results, kinematic improvement was incredibly variable across subjects and the current protocols were not able to provide the appropriate challenge for each patient. AMRR showed the benefit of providing interactive feedback and integrative training under constant supervision of the therapist. Therefore, moving forward in the design of HAMRR, more finite adaptation should be included, but it needs to be automated to support the system's fading of supervision.

While the therapy protocols had limitations as previously discussed, it would not be possible to identify appropriate manners to integrate or design for automated adaptation without the knowledge gained from the fixed, controlled protocols. Now that these protocols (designed with the primary goal of forming reductionist hierarchies through integrative training) have been tested and successes and limitations have been identified (as previously discussed), avenues for automated adaptation can be discussed.

6.7.1 Overview of Current Approaches to Automated Adaptation

Currently there are a few major approaches to adapting training within the context of stroke therapy.

6.7.1.1 Therapist Control

One category of adaptive technology in stroke rehabilitation systems is therapist-controlled systems, like AMRR. These are systems in which the task type and sensitivity are set by a therapist who continuously or periodically observes the therapy.[87-90] There are also many examples of telerehabilitation systems that, while the therapy is performed in the patient's home, the therapy requires constant therapist telepresence.[91,92]

6.7.1.2 Reactive Sensitivities

Another category of rehabilitation systems feature reactions to a previous activity's performance. The IIT-Robot[93], Braccio di Ferro robot[94] and ADAPT system[95] have methods to evaluate patient performance based on extracted features that are utilized to evaluate patient progression within a task. Changes in this evaluation are then connected to methods to update the task challenges for the next trial such that the challenge is at an appropriate level for the patient at a given moment in time.

6.7.1.3 Decision Networks

One of the more unique approaches to adaptation with rehabilitation tools found comes from a system developed by Kan et al for upper limb reaching tasks.[96] This system utilizes Partially Observable Markov Decision Process (POMDP) to model the patient's progress through therapy. [97] This model requires that there are a finite set of states, actions and observations and a probability distribution of transitions between states. POMDP was utilized because the developers viewed the measurement as uncertain and some observations to be unobservable, such as fatigue. So instead, fatigue was modeled as a probability of belief states. This model was also utilized because the developers believed that a given action will not always have the same result given a starting state, but they could begin to estimate the probabilities and rewards of outcomes. The model also allowed for incorporating the history of which actions were found to be most effective.

Given a state, the decision of which action to take took into account fatigue, the furthest target distance achieved by the patient, the stretch beyond current ability the action would require and the learning rate of the patient. The observations were set as time to reach the target, the ability to stay within a path, and the compensatory movement of the patient. On top of this, through the use of reward functions, the system was "motivated" to keep certain challenge parameters high while maximizing patient control and minimizing compensation. The resulting space was very complex, having an estimated 3,000 states. But when using the model, it had a 90% therapist agreement when changing task parameters and a 43% therapist agreement when deciding to stop the task due to fatigue. This methodology could be applicable to music instruction as well, as there could be a set of decisional and observational criteria for progressing through lessons as well as motivating the algorithm to keep the challenge level appropriate.

6.7.2 Application of Decision Networks to HAMRR

Decision networks seem to be the best next logical step for automated adaptation. Each state would represent a configuration of the system (including task and feedback environment settings). Starting states could be pre-composed based on an initial patient profile and therapist ranking of priorities. Then the rules for the transition between states could be set based on a combination of current and long term patient assessment as well as rules for task variability and feedback environment progression. For example, if a therapist wanted to prioritize trajectory, the system would shift the probability towards states with an emphasis on trajectory training.

Based on the overall progression of the patient, the selected object would be based on current difficulty and as well as the desire to introduce some variability in the protocol, which was already tested within the HAMRR pilot study. For example, if the training were beginning, a higher probability would be given to utilizing a flat object. If the patient had been using the flat object for quite a few sets in a row, a higher probability could be given to a different location or a different type of object.

The transition between states could also be dictated by the need to fade in-between feedback levels, and thus higher probabilities might be given to the lower feedback levels early in training, with some focus on fading between them. Again, this fading rule was established in the HAMRR pilot study, thus giving a good basis for exploring automated approaches to application of this rule.

Finally, much like the example research project suggested, fatigue could be assessed. By looking at current performance, if a sustained decline was detected, either a different task could be introduced for variability or the session could be ended to avoid exhaustion and frustration. As previously described, this was a key limitation of the standardized, inflexible sessions that were tested. There was no way to react to problems, such as fatigue, except by making changes manually to the following session's protocol. Therefore, beginning to determine a metric to assess frustration or fatigue would be a really important feature to add, which would not only have direct benefit to within-session adaptations, but also longer-term changes across sessions. With all of these examples, the therapist could still have some oversight in adjusting sensitivity parameters that could affect the state transition probabilities, and in all cases should have oversight.

At this point some confirmation has been received about components of the protocols that worked, and others that need refinement. Therefore, from an iterative design perspective, automation should be introduced to these protocols with the previously identified rules that were confirmed in the pilot study. However, there are still remaining questions of dosage and the appropriate timing for an adaptation to a protocol. Therefore, therapist input will still be valuable and necessary to help answer some of these unknowns simultaneously with collecting data to create better automated protocol adaptation decisions.

6.8 Considerations for Implementation of INR Within an Interdisciplinary Team

The design and implementation of HAMRR has not only provided suggestions for system and user experience improvements, but also opportunities to reflect on the nature of interdisciplinary collaboration design within INR.

6.8.1 INR Requires an Aggressive and Pro-Active Testing Schedule

Looking back on the project development cycle, there was a lack of organized regular user testing of the system, and this is crucial in any aggressive iterative design flow. As previously discussed, small studies were run to test the feedback clarity and get a basic sense of the usability of the system. However, more should have been done.

The challenge with INR is that, ultimately, the best test of the system is with actual stroke patients. However, this does not mean that iterative testing should be avoided and held until the end of development. Rather, small tests with non-impaired subjects should be held frequently. It should be part of the design plan to identify moments in which test of iterations of a prototype can be conducted.

When developing HAMRR, the design and development mainly fell within structure that most closely resembled a Waterfall Model[22]. In this structure, stages of development primarily move forward in a linear fashion: one stage leads to the next. Overall, this is how HAMRR developed: Requirements led to Design, which led to Implementation, which lead to Verification and Maintenance. One of the reasons that the Waterfall Model fell into place naturally was most likely due to the main form of verification for INR: patient user testing.

The reason to bring outside people into the system frequently is to provide an outside perspective. Designers of a system know, at a very high resolution, how their systems work. This comprehensive knowledge, however, can bias their testing of their own systems. Even if non-impaired subjects cannot replicate characteristic movements of a stroke patient, their input and responses will still help to identify problems in the design or implementation before a significantly more expensive patient testing period.

6.8.1.1 Code Design Approaches Can Provide Testing Methodologies

Code development paradigms can be looked to for examples of establishing regular testing protocols. Test Driven Development work-flows are setup to identify tests for code functionality, before the code is even written. Similarly, test for components should be identified early on and throughout the development, as suggested by engineering design as well, so that small user tests can be setup effective and efficiently, and validation of components can be drastically improved.

6.8.1.2 Move System-Wide Iterative Development to Spiral Method

Similarly, the use of a Spiral Method would also be advisable.[22] In this model, on the way from the start of development to release, multiple cycles are completed of:

1. Determining objectives
2. Identifying and resolving risks
3. Development and Testing
4. Plan next iteration

While each stage is completed in each cycle, the implications of the stage different depending on where in the overall development timeline the stage is attempted.

6.8.1.3 The Design Team Needs Constant Access to a Full Toolset for Testing

In order to support a productively aggressive user testing schedule, the design team needs consistent access to a full mock up of the system for debugging purposes. In the development of HAMRR, two full systems were built and sent to the testing sites. Within the lab at ASU, was a prototype system that had many legacy hardware components that did not accurately represent the full system. This made it very difficult to test new components before the later portion of the study with more impaired subjects. Many components of the transportable cone had to be tested in isolation, and even then, this did not consider the majority of connections that needed to be tested. Part of the solution is better code modularity, as previously described, which would likely limit the time to extend code for new components as well as lessen the opportunity for code conflicts. However, if the full system was available, it is likely that the grasp training discontinuity problem could have been identified sooner and a possible solution could have been approached.

6.8.2 Apply a Modular Architecture Approach to Understanding the Research Goals of INR Design and Development

Just as it has been suggested that code development should be modularized within a predesigned, more stable architecture, research questions of INR should be modularized the same way. Any INR system will have overall research questions, such as in regards to the usability of the system and if it has any significant clinical benefit. However, within INR system development research, there will also be other sub-questions in regards to the stability and utility of a new sensing hardware or computational algorithm.

These research questions need to be properly categorized to identify what needs to be tested with a complete system or with an isolated component as well as what can be tested with normal or impaired subjects. By doing so, the development team can begin to form an experimental architecture to see where the individual, more specific, research questions originate and how they combine within the larger questions. Many times, a team leader may have an innate sense of this space, but a more public representation or model would be beneficial for the cohesiveness and productivity of a team.

This organization can standardize how individual components need to be tested as well as optimize how they will be tested. In addition, it will clarify what is still unknown after component level testing, and therefore, what will need to be discovered with comprehensive patient-system testing. This can also be helpful to spur more system component level iterations as questions that can be identified with isolated testing are made explicit, and thus will strengthen aspects of these components prior to their integration. Finally, by seeing how these research questions combine, project creep can be more easily avoided. By looking at the landscape of questions to answer and how they reflect in various stages of testing, components that appear to be more problematic or require a different set of resources than what are available can be efficiently shelved or removed.

6.8.2.1 Example Framework

This framework would begin with identifying what are the Large Research Questions. As INR is inherently research driven, its design and implementation will have many large goals for questions to answer throughout the design and implementation process. Within each Large Research Question, will be at least one (but possibly many) Sub Research Questions. These questions begin to break down the larger question into specifics that can be implemented and tested. Thus, from these Sub Research Questions will come specific Implementations. (Figure 30)

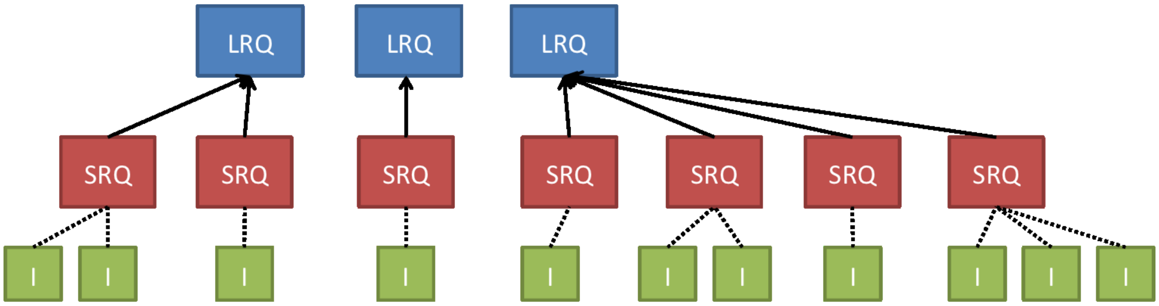


Figure 30 – Example generalized knowledge framework. Large Research Questions (LRQs) are broken down into Sub Research Questions (SRQs) with appropriate Implementations (Is) attached.

It is very important for the entire design team to see this space. Each team member will have their own research questions, however when co-developing on a complex integrated system, it is very important for each person to see how their research fits within the overall system design context. The modeling of research questions and implementations will also allow the team to make informed decisions during iterative design cycles in regards to which questions have a higher priority, and as a result, which implementations will have priority. It will also show which components are absolutely necessary and which can be cut. (Figure 31)

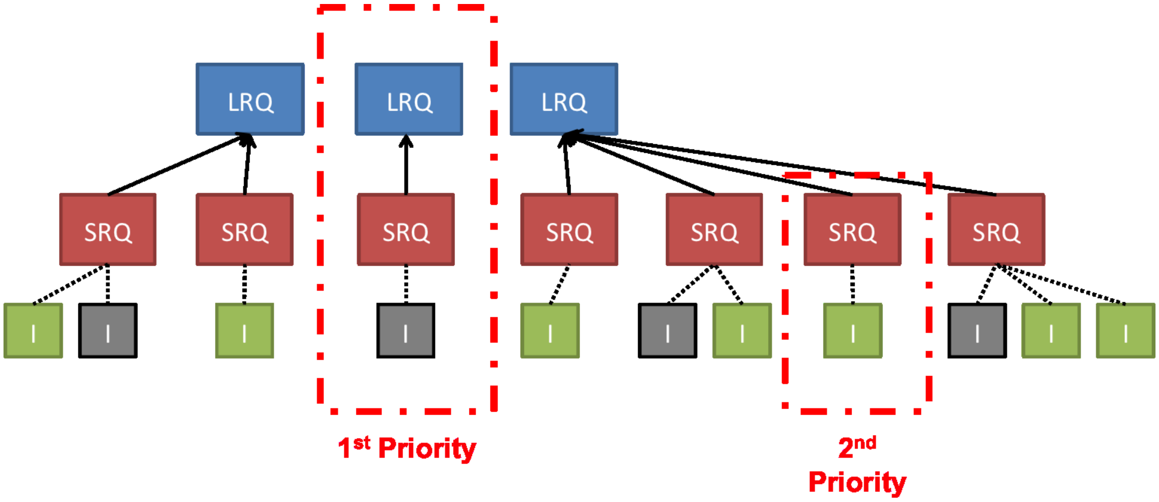


Figure 31 – Example prioritized generalized knowledge framework. Research, and thus development priorities, can be identified. It is also helpful to identify which implementations have already been completed or can leverage significant existing code (grey squares).

From this map, it is important to consider each implementation within the overall system architecture that the team develops. It is possible that some parts of the implementation are already complete, or that implementation will require modifications to multiple plugins or components. This information should be readily available to the team so that designers can see the interdependency of specific components as well as remaining work to be completed.

Most importantly, with each added implementation, consideration should be given for the addition/connection criteria. Similar to the Test Driven Development philosophy, each implementation (before design and building begins) should identify the metrics and tests needed to test its validity. This reinforces the idea of the designer understanding that they are not developing in isolation and may need to address specific implementation criteria based on underlying research questions.

As a crucial part of identifying these validation metrics, each implementation should further define what components can be tested in isolation and what components need to be tested after being integrated into the whole system. Further definition should also be provided as to which of these tests (isolated or integrated) can be completed with non-impaired subjects or with impaired subjects. In this way, not only is a given designer required to think about their work in context of the larger system, but also each member of the team has a big-picture overview of where each questions and implementation fits within the system and where respective priorities lie.

6.8.3 Documentation and Dissemination Should Support Cohesive Team Iterative Design

Iterative design in an academic environment can be a challenge. By nature of academia, team members are only involved with projects for finite periods of time (with very specific focuses) before moving on to other projects or institutions. Therefore, the design team will be in flux, as well as individual component development. This emphasizes again the need for a stable architecture. As previously shown, this is not to say that the architecture can never change, but it should be the most stable aspect of the system development so that it can keep development moving forward with components in flux.

Similar to the suggestion of maintaining an architecture of research questions, there needs to be an architecture of development history. There should be a standardized way in which team members can document their progress and demonstrate how their components work. As with any group, the team member flux can be completely unannounced in academia and therefore a team cannot wait for a culminating paper or thesis to summarize and pass on to future team members all of the knowledge gained. Instead, to minimize risks of losing information or having significant development delays, research and development work should be documented along the way. In addition, this documentation should be written in the language of the team. In other words, papers that are released to the general public may present ideas in a certain way to bring a wide audience into the domain of the team. However, this step is not needed internally. For example, code should be documented along the way (which could also offer each team member a further chance to debug and refactor code). In addition, simple documentation should be created for how to run a GUI interface (again, by merely writing this process down, another opportunity is afforded to think critically about the GUI interactions). There will certainly be pressure during tight design iteration cycles where this may seem like a lower priority, and in development processes such as agile code development it is not given, accordingly, significant importance. However, the documentation of progress should be just as integral as writing new lines of code. Instead of writing one summary document at the end of the process, the document should grow and live with the project and a core technical, utility language should be agreed upon by the team to efficiently share this knowledge. In addition, this documentation does not have to follow the literal connotation of the word and provide text documents of explanation. When possible, the “documentation” should be functional. An example of this, connection back to a previous suggestion, is an API. By creating an API for a plugin, a top level summary is provided of the key inputs and outputs. This would already provide a good starting point to understanding the code as well as immediate opportunities to develop on top of the code, which could bootstrap a developer’s understanding of how the plugin functions.

By coming up with a method for documentation and dissemination, team members can also stay on the same page. Ideally, team meetings would hold this responsibility, but even so, complex systems require a wide and varied team. Evolving documentation would ensure that team members could get a detailed, comprehensive glance at a component to address immediate questions. Each team should come to a consensus in the early stages of the design of what this documentation should look like and how often it should be presented. Team members should also understand, especially in an academic context, that their work will be developed beyond themselves. Someone else should be able to (and undoubtedly will need to) pick up this work and continue. Comprehensive development of documentation will help this along.

6.8.4 Build Systems for Future Designers' Iterations

Similar to the previous point, systems need to be designed for iterations beyond the initial team. Given that it can be difficult to identify proper requirements for one system iteration, let alone speculative future iterations, this point may be relatively dismissed. However, it should be given serious consideration, and categorized in multiple time scales.

First, systems should be designed to give immediate feedback (both to the direct user and outside design teams) when problems occur. Users should have the ability and tools to document problems or errors in a smooth streamlined way. As previously discussed, INR system should have some automated way of documenting computational problems. In addition, GUI interface for live or later annotations to the user are extremely helpful. INR systems are experimental, and due to their experimental nature, problems are bound to occur that were never accounted for in the initial design. This is not to say designers of INR should not look to eliminate problems or bugs, but all members of the INR development and implementation should realize that they are part of an experimental process. And instead of casting this as a limiting pall over the experiment, it should be utilized. Systems should have integrated ways to document when things went wrong or when suggestions for future improvements are made. That way they can be fairly time synced with the data and use of the system and designers, in the short term, can make adjustments for immediate iterations.

However, longer-term interactions with the system should also be considered. The life of the project should extend beyond the initial team. This can be done through the previously discussed documentation and error reporting. In addition, there should be a push to make the code open source. There is certainly a competitive nature to scientific research, similar to corporate developments, in which there is a push to be the first one with a solution. However, there is more value for people to learn from one another and not waste time reinventing the wheel, in order to advance stroke rehabilitation research faster. For example, motion analysis algorithms, and its architecture (after incorporating the previously discussed changes), should be available to others with documentation about what it can do. This grows the vitality of the original project in unique directions and affords more opportunities for the code and hardware to be tested and refined. HAMRR code has already been extended to look at specific effects of audio feedback on different kinematic attributes of a reaching movement.[98] This mandates, however, that code be developed in modular, scalable ways. It will be difficult for outside groups to interface with or utilize code that is highly entangled for one specific application. Widely disseminated, efficiently modularized code, with documentation and presentation of results, could be a huge push forward for INR design as well as other complex system design.

CHAPTER 7

A GENERAL MODEL FOR INTERACTIVE SYSTEM DESIGN

The work previously shown has demonstrated the complex space that INR design must consider, and through the specific details of HAMRR design and evaluation process, how interdisciplinary research and integrated design (utilized within iterative design) is crucial to form comprehensive solutions. The Toyota model for Set Based Design[71] exemplifies the strength of iterative design within a complex application space. In order to standardize comprehensive approaches to INR design, implementation and evaluation, a similar model should be developed for INR research. As a conclusion to this work, and a starting point for more generalizable applications, I present a synthesized model for INR that highlights and synthesizes approaches and considerations previously presented for reflections on improved complex system design.

7.1 The Inherent Complexity and Necessity of the Complex Space

As was demonstrated, INR exists within a complex space of dichotomies (Figure 2). INR systems must balance human and digital components as well as physical and computation. In addition, the varying nature of these relationships will most likely need to be addressed across an ecosystem of INR systems that are not only individually flexible and extendible, but are designed with overarching ideas of continuity.

In addition, there are three crucial users of INR systems that each need their own complete user experience needs met. Patients require an enriching training experience that can help form individual models for self-assessment and generalization of skills towards ADLs regardless of current ability. Therapists require assistive tools to maximize their therapy time with patients and assist in forming detailed, consistent assessments of patient impairment. System designers, as members of the iterative process, need constant information about the system performance as well as reflections on other users' experiences in order to maintain and improve the system.

In order to address this space of human and technological constraints, interdisciplinary fields (beyond neurorehabilitation exclusively) should form robust design constraints for how interactive learning could maximally be implemented through task experience, assessment, and feedback design. As discussed, in order to determine the optimal combinations of designs addressing these factors for a given implementation requires iterative design approaches as well as critical considerations for modularity in hardware and software components.

7.1.1 Embrace the Complex Design Space and Flow Between Space of Possibilities and Specific Implementation

While the previous considerations of the complex INR design space are required to be addressed, this complexity can also provide a benefit. As the Set Based Design

(SBD) method highlights, problems in iterative design can result from honing in on a solution too finitely and too quickly and spending any remaining design time on making the solution work for a given problem.[99] Rather, the SBD method suggests to gradually hone ideas from a full consideration of the possible solutions.

This approach is especially crucial for identifying solutions to improve or replace implementations that have been found to reach their limit of returns. When this limit is identified, designers can take a step up from a finite solution, to the more complex possibility space to identify where a new hybrid solution might exist.

7.1.1.1 Applicability to Improving HAMRR's Sensing Infrastructure

As suggested, one of the key hardware limitations in the current design of HAMRR is the ability of the OptiTrack cameras to properly track complex movements within the more variable, confined tabletop space. Therefore it is suggested to replace this solution as properly tracking and assessing complex movements will be crucial for transitioning training from the system to activities in the home.

Therefore, in this example, a limit to the return that OptiTrack can provide has been reached and a more appropriate solution needs to be found. Instead of honing in further on the OptiTrack sensing solution and working to see if this hardware setup can be modified to improve the sensing capability, the design should take a step towards the more complex space of possibilities and see where new integrative ideas might be found. Work at Carnegie Mellon University has begun to see how computer vision solutions might provide more robust sensing in a home environment. However, this form of tracking may not be enough, and instead, an integrative approach that couples computer vision sensing with the embedding sensing in the system's physical setup and training objects may be the most robust.

7.1.1.2 Applicability to Improving Protocol Adaptation

It has been demonstrated that the fixed protocols were able to test and provide promising results for how feedback environments and varying complexity tasks should be utilized to structure training from simple reach to touch interactions to complex task training. However, because the paths were fixed and could not react to evolving patient task performance, many patients found the protocols to either be not challenging enough or, for the more impaired participants, prohibitively difficult to complete. In this example, the limit of returns of fixed path protocols has been reached. Previously identified interactive training rules have been verified through the pilot studies, but more responsive adaptation is required.

Again, instead of honing in on the fixed path solution and modifying it try and improve its efficacy, the design should take a setup up in complexity toward the more varied solution space.

As previously presented, automated adaptation has already begun to be demonstrated in INR systems through reactive sensitivities, and more interestingly, decision networks. The applicability of these approaches and their integration with the identified interactive training rules should be explored.

However, it may also be appropriate to look further in the complex possibility space to see from where else adaptive training decisions might be made. Since INR systems have multiple users, what can each of these users provide to improving adaptive protocols? What information can the therapist provide (at different time scales depending on supervision levels) to strengthen adaptation decisions? Or, more interestingly, what can the patient himself or herself provide to make adaptation decisions?

As previously discussed, one of the limits of the user experience of HAMRR was an inability to address higher levels of the user experience to encourage patient proficiency and creativity. What if HAMRR was able to demonstrate and empower the patient to not only self-assess their own individual movements, but also think critically and understand the form of creating their own training protocols? Future system designs should look to informing protocol adaptations by allowing a proficient user to identify their own goals and create their own training protocols utilizing all of the tools available during interactive training.

Thus, benefits of taking a step back in the design to the more varied space of possibilities can be demonstrated. Not only can this methodology provide solutions where previously a limit had been reached, but it can also provide more comprehensive solutions that improve multiple aspects of the system design and user experience.

7.1.2 Embrace the Complex Space of the Design Team

It should not be overlooked that comprehensive INR design not only requires considering integrative design and interdisciplinary research, but requires that these be achieved through a highly interdisciplinary team. Therefore, not only is the complex contextual research and design space important for INR, but the integration of a design team will directly impact the success of the overall design.

Therefore it is crucial for the INR design process to consider this complex design team space. As previously discussed, an understanding of how research ideas connect across members of a team and integrate implementations within an overall system architecture is critical to understand. I propose that this is especially true for academic research projects. The implementation of SBD within Toyota leverages the presence of increased resources that likely does not exist in the same magnitude in academia. Within Toyota, this mainly manifests itself in the form of increased personnel to specialize in a particular part of the iterative process or suppliers who can focus on a specific implementation, and therefore, provide multiple solutions to a particular system component.[99]

This industry flexibility does not necessarily exist within academia. Rather, teams are inherently smaller and specializations are many times defined within research goals or contained graduate degree applications. The result is a dynamic team that has to balance larger, long-term research questions with highly specific and motivated implementations.

However, just as was presented in regards to the complex research and design space of INR, the complexity of a design team is not something that should purely be addressed, but should also be leveraged in order to come to comprehensive, truly integrative solutions. When a research question or implementation reaches its limit and its returns are diminishing, the design teams needs to take a step up and look at the space of possibility within the team and the research questions that the interdisciplinary team are able to define. This approach and utilization of the complex design team expertise and focus is fully integral to fading back and forth between a complex design space and specific implementations.

7.2 Address Complexity Through Modularization

In order to simultaneously address and leverage a complex design space, I propose that modularity is key. To implement an integrative, iterative design approach (which has been previously proposed as integral to INR) and flow back and forth between the complex space of possibilities and specific implementations requires modularity, or the ability to define solutions within components and have the flexibility to combine and redefine these components as necessary.

7.2.1 Utilize Hierarchical, Functional Decomposition Approaches to Identify Key Components

As has been previously discussed, functional decomposition approaches should be used to identify sufficient, yet integrative, development components. These components should be considered within a relative hierarchy to understand how these components should be developed and, importantly, integrated. This requires that modularity be used within all of the complex aspects of INR design.

7.2.1.1 Modularity in Software

As has been emphasized throughout this dissertation (and my work), modularity in code design is crucial. Modular plugins within a flexible overall architecture supports faster iteration and integration as well as provide malleable structures that can be extended for future applications. Thinking of the task experience, assessment, feedback, and user experience created design constraints for software modules to be developed.

The benefit of HAMRR's plugin structure has already been demonstrated. Components could be developed in parallel by multiple developers and maintained as such. As iterative design identified one solution over another, code could be extended to accommodate new solutions (such as the transportable cone) with limited need to think about how this fits within the overall system architecture. Rather, focus was spent on how the new solution fits within an individual plugin (which was already connected within a stable architecture).

7.2.1.2 Modularity in Hardware

One aspect that has not been discussed in great detail, but should be pursued further, is the benefit of modularity in INR hardware. As has been discussed, task design needs to address varying patient ability while simultaneously maintaining a consistent end goal of training a complex task. HAMRR's design demonstrated the strength of utilizing rapid prototyping methods and cheap electronics to create modular task objects that can cover a range of patient ability. This development work also allowed for the quick design and implementation of a hybrid object that would be usable by more impaired patients. This approach should be pushed even further.

Coupled with adaptations and sensitivity changes within the system, hardware should be utilized as an additional adaptation component. Objects could have different aperture requirements or physical weights that would assist in making tasks more challenging in the physical space. The location of an object on a table could be modularized to explore training in multiple body spaces.

Similarly, as hardware will largely be space and context specific, sensing solutions should also be designed as modular components. The transition between marker-based sensing and computer vision solutions should be flexible such that it does not impact the overall training experience and can be selected based on the environment. As previously discussed, continual technologic improvement and evolution will continually provide new components, and modular hardware design will be able to react to these new solutions quickly.

7.2.1.3 Modularity in Assessments

As previously discussed, INR systems (and experimental hardware and software solutions in general) will likely provide more detailed, yet experimental, measures of user performance or progression. These new measurements represent the benefit of introducing technologies in the complex space of INR, however they should be validated with the breadth of existing knowledge from traditional measures. Therefore, assessments should also be designed in a modular fashion, to span the range of validated measures as well as integrating multiple sources of assessment (across all users). These modular, yet integrated assessments provide the most complete picture of user progression, and therefore will be crucial in making comprehensive adaptation decisions that leverage multiple sources of assessment. In addition integrated assessments may also provide benefit in standardizing traditional qualitative measurements.[100]

7.2.1.4 Modularity in User Experience

Just as has been discussed with software and hardware, user experience can also be modeled with modularity. INR systems need to address a spectrum of therapist supervision (both in terms of type and amount), adaptation, and user control. The therapist interface should be designed within modules as well, such that development within a system can be streamlined, but also to help maintain continuity across systems.

For example, the usage of the system by the therapist should be broken down into a functional decomposition (much like the larger code design) in order to best understand how an interface could be created. The notion of designing an interface with a user in mind can only be tested so much before its implementation, which necessitates being able to reach to design changes to the interface as observations are made. Therefore, identifying how an interface could be modularized is crucial.

Similarly, modularity in user experience also connects to the idea of the patient creating their own therapy protocols. If therapy protocols (representing the combination of interactive training constraints, tasks and feedback environments) were designed in a modular fashion, then these same modules could be presented to the patient as tools to meet their own self-identified training goals. Therefore, as a start, HAMRR identified some crucial components to structuring interactive training in an unsupervised environment. The next large step is to see how automation and patient input might improve the automation decision.

7.2.1.5 Modularity in Design Team Goals

As previously discussed, an academic environment does not have the specialization resources that an industry development team may have available. Therefore it is crucial that each member be able to innately see the connection between implementation, their own respective research question, and the larger research questions of the group. Therefore, seeing how larger research questions break down into components that can be developed and prioritized is important.

The SBD method also demonstrates the importance of defining and refining from sets of solutions, not just a singular concept.[99] In a highly integrated, complex design as INR (where individual team members may be focusing on specific implementations) a designer needs to therefore know how their implementation integrates with the overall system design, as the solution for a particular component, may need to make certain considerations for its integration within the system. As the SBD method suggests, the solutions for complex system design may not be contained within individual modules, but rather, across the integration of modules. Therefore, identifying the larger research project as a collection of modules can not only simplify the development space, but also help ensure that solutions maximize both individual performance metrics and larger research goals.

A development strength can also directly result from this approach, as an individual with the larger research framework in mind, now thinks about how their component integrates with the larger system. I propose that this would not only help the design team be more integrative (in that members have a sense of large scale system goals and how components integrate) but would also strengthen individual development as a module cannot be exclusively developed in isolation. This approach helps reduce the integration errors that might result if the process of integration and development are maintained as separate processes completed in series.

7.2.1.6 Maintain an Architecture that can Accommodate Parallel Module Development

As a result of the functional decomposition process, functional modules should be identified, but connections and larger concepts of architecture should be identified as well. In order to iterate and develop in parallel, there needs to be some baseline understanding of how these modules fit together. HAMRR demonstrated plugins that fit within specific categories to maintain a flow of data through the system. This architecture could be extended if needed, but each module was developed with a basic understanding of the larger context with which it fit.

7.3 Strengthen Future Design Iterations from Exploration of Possible Solutions

Another strength that the SBD method presents is the idea that time invested refining from a set of multiple solutions is not wasted, but rather supports the development of future iterations by creating a knowledge-base of the solution possibility space.

As has been discussed, INR requires the use of iterative design, and due to its design in academia being inherently research-based, forming a knowledge base for future improvement and extension is important. In addition to creating the previously presented modules for understanding the connection of research questions, iterative design needs to be supported by previously gained information.

Therefore, dissemination of knowledge gained is crucial. Members of a team need an understanding of the advances that have been made at the system level. Depending on the integration of components, knowledge of advances at components may be crucial as well. As discussed, this requires an internal language and documentation method for team members to utilize so information can be shared efficiently and effectively.

Furthermore, this also presents a design consideration to build systems for future designers. Systems should be developed with the mindset that iterative design will move beyond their work (especially in academia where a specific program goal may be met before the work of the larger project is done). This further reinforces the idea of design with modularity: support the refinement and extension of the system not only within the cycle of preparing for a pilot study, but for other extensions that may not yet be identified.

7.4 Maintain an Aggressive, Prioritized Iterative Testing Schedule

Testing and evaluating within iterative design is crucial to form an aggregating knowledge base. INR development should prioritize and understand the evolution of testing, such that non-impaired and impaired participants can help to the maximal benefit.

As SBD suggests, iterative testing through gradual solution refinement should iterate on larger ideas before iterating on specifics.[99] This concept is equally true in INR. As presented, just because a pilot study begins, does not mean the iterative testing process ends. Therefore, iterative testing will and should occur at multiple time scales throughout the process of design, implementation and evaluation. However, more detailed component adjustments should be made through iteration late in the development cycle as opposed to changing large-scale ideas during the pilot study. If large concepts need to be readdressed, this may be symptomatic of improper iterative testing. Therefore, not only should iterative testing be frequent, but also it needs to be prioritized both in terms of required participant population and granularity of testing.

7.5 Extending the Model Beyond INR

The proposed methodology and synthesized model is not limited to INR but rather is applicable to other complex interactive systems. The need for approaches to complex system design will become ever more pressing. As the Internet of Things begins to rapidly introduce more sensing hardware possibilities, a deluge of available data will continue to flow. Therefore the challenge will be to create systems that can provide unified experiences with truly empowering information at the integration of all of the hardware and software.

One immediate next step is to look at how the interactive training paradigms of HAMRR might be used in other learning applications. I have already begun to consider what a mobile guitar tutor might look like and how this experience might allow for individualized guitar instruction through an iPad app. This exploration was begun as part of my comprehensive exam for the Media Arts and Sciences program.

7.5.1 Example Application within Guitar Instruction App

One of the specific areas of the guitar instruction concept that I was interested in was how to show long-term progression to a user. Much like physical therapy, it can many times be very easy to focus on the short term gains made in learning an instrument and lose the overall sense of progression. This can especially be true during days of frustrations and seemingly slow progress. I began to think about how a user might have a sense of overall progression of their long term learning gains by purposefully recording their journey. The same exploration would be interesting for INR applications. How can a stroke survivor come to an appreciation of the results of their day-to-day therapy in the context of the long rehabilitation process? Here again, the idea of identifying modules for user information and making a first iteration attempt at how this information might integrate within the overall form of the experience was key.

Another area that the guitar instruction app exploration had me thinking about is how users can feel empowered to create their own practice sessions. I proposed that initially, the app would suggest a modular structure of dividing 30-60 minutes of practice time into structured, focused pieces. Then once the user had gained competency, the user could connect these pieces however they wished to create their own practice sessions based on their desired goals. As previously discussed, a similar idea would be interesting to explore for INR.

7.5.2 Creation of an Academic Project Management Tool

I would also be interested in seeing how the proposed ideas for modularizing team research goals could drive the development of a novel productivity tool for academic technology research projects. As was discussed, academic research environments provide very unique collaborative spaces and designing a productivity tool that would help manage the many research, designing and testing requirements and interests of each team member could be very beneficial for other collaborative workspaces.

7.5.3 Overall Emphasis on Inclusion of Prospective Users

One of the key areas where I would like to push the methodology furthest is in its inclusion of the users in the system to help define the scope and functionality of the system. In academic research, it feels as if many times there is a top down approach to development where the grander ideas come from larger research and we as system designers provide something that we think will be beneficial to a prospective user. It could be very easily argued that my methodology resembles this top down approach. However, I would readily like to enhance this approach and ensure it is always a hybrid, integrating the contextual multi-domain research with collaborative design with users (which for INR would include patients, therapists and system designers).

REFERENCES

1. Go AS, Mozaffarian D, Roger VL, et al. Heart disease and stoke statistics – 2014 update: a report from the american heart association. Circulation. 2014;128:00-00.

2. Centers for Disease Control and Prevention. Outpatient rehabilitation among stroke survivors: 21 states and the district of Columbia. MMWR Morb Wkly Rep. 2001;56:504-507.

3. Gutiérrez RO, Galán Del Río F, Cano de la Cuerda R, Alguacil Diego IM, González RA, Page JC. A telerehabilitation program by virtual reality-video games improves balance and postural control in multiplesclerosis patients. NeuroRehabilitation. 2013 Sep 12. [Epub ahead of print]

4. Rostami HR, Arastoo AA, Nejad SJ, Mahany MK, Malamiri RA, Goharpey S. Effects of modified constraint-induced movement therapy in virtual environment on upper-limb function in children with spastic hemiparetic cerebral palsy: a randomized controlled trial. NeuroRehabilitation. 2012;31(4):357-65.

5. Subramanian SK, Lourenço CB, Chilingaryan G, Sveistrup H, Levin MF. Arm motor recovery using a virtual reality intervention in chronic stroke: randomized control trial. Neurorehabil Neural Repair. 2013 Jan;27(1):13-23.

6. Saposnik G, Levin M. Virtual reality in stroke rehabilitation: a meta-analysis and implications for clinicians. Stroke. 2011;42:1380-1386.

7. Laver K, George S, Thomas S, Deutsch JE, Crotty M. Cochrane review: virtual reality for stroke rehabilitation. Eur J Phys Rehabil Med. 2012 Sep;48(3):523-30.

8. Loureiro RC, Harwin WS, Nagai K, Johnson M. Advances in upper limb stroke rehabilitation: a technology push. Med Biol Eng Comput. 2011 July;49:1103-1118.

9. Lo AC, Guarino PD, Richards LG, et al. Robot-assisted therapy for long-term upper-limb impairment after stroke. N Engl J Med. 2013 May;362:1772-1783.

10. Ruparel R, Johnson MJ, Strachota E, McGuire J, Tchekanov G. Evaluation of the TheraDrive system for robot/computer assisted motivation rehabilitation after stroke. 31st Annual International Conference of the IEEE EMBS. 2009 September 2-6. 811-814.

11. Holden M. Virtual environments for motor rehabilitation: review. CyberPsychology & Behavior. 2005 June;8(3):187-211.

12. Pridmore T, Green J, Hilton D, Eastgate R, Cobb S. Mixed reality environments in stroke rehabilitation: interfaces across the real/virtual divide. Proceedings of the 5th International Conference on Disability, Virtual Reality & Associated Technology. 2004 September 20-22: 11-18.

13. Krebs, I. et.al., Robot-aided neurorehabilitation. IEEE Transactions on Rehabilitation Engineering. March 1998:6(1):75-87.

14. Piron, L. et.al. Virtual environment training therapy for arm motor rehabilitation. December 2005. 14(6).732-740.

15. Sanchez-Vives MV, Slater M: From presence to consciousness through virtual reality. Nat Rev Neurosci 2005, 6:332-339.

16. Vinayagamoorthy V, Brogni A, Gillies M, Slater M, Steed A: An investigation of presence response across variations in visual realism. Proceedings of the 7th International Conference on Presence: 13-15 October 2004; Valencia 2004, :148-155.

17. Gombrich EH. The story of art. 16th ed. London: Phaidon Press; 1995.

18. Pappert H, Harel, I. Situating constructivism. In: Constructionism. Ablex Publishing Corporation; 1991.

19. Wilson B. Constructivist learning environments: case studies in instructional design. Education Technology Publications; 1996.

20. Krakauer JW, Mazzoni P, Ghazizadeh A, Ravindran R, Shadmehr R Generalization of motor learning depends on the history of prior action. PLoS Biol. 2006;4(10): e316.

21. Furht, B. Encyclopedia of multimedia. 2nd ed. New York, NY: Springer; 2008.

22. Lidwell W, Holden K, Butler J. Universal principles of design: 125 ways to enhance usability, influence perception, increase appeal, make better design decisions, and teach through design. Beverly, MA: Rockport Publishers; 2010.

23. Reprinted from Phys Ther. 2014 November 25; doi:10.2522/ptj.20130581; [Epub ahead of print], with permission of the American Physical Therapy Association. Copyright © 2014 American Physical Therapy Association. This is the unedited author version.

24. Davis B, Sumara D, Complexity and education: inquiries into learning, teaching, and research. 1st ed. New York: Routledge; March 28, 2006.

25. Saffron JR. Statistical language learning: mechanisms and constraints. Current Directions in Psychological Science. 2003 August;12(4):110-114.

26. Kleim JA, Jones TA. Principles of experience-dependent neural plasticity: implications for rehabilitation after brain damage. J Speech Lang Hear Res. 2008 Feb;51(1):S225-239.

27. McEwen SE, Huijbregts MP, Ryan JD, Polatajko HJ. Cognitive strategy use to enhance motor skill acquisition post-stroke: a critical review. Brain Inj. 2009 April;23(4):263-277.

28. Dewey J. Art as experience. New York: Minton, Balch & Company; 1934.

29. Arnheim R. Art and visual perception. Berkeley: University of California Press; 1974.

30. Wong D, Pugh K. Learning science: a Deweyan perspective. Journal of Research in Science Teaching. 2001 March; 38(3):317-336.

31. Pappert H, Harel, I. Situating constructivism. In: Constructionism. Ablex Publishing Corporation; 1991.

32. Rogers Y, Price S. Lets get physical: the learning benefits of interacting in digitally augmented physical spaces. Computers & Education. 2004 August-September;43(1-2):137–151.

33. Mussa-Ivaldi, F. Modular features of motor control and learning. Current Opinion in Neurobiology. 1999 December;9(6):713–717.

34. Braun DA, Aertsen A, Wolpert DM, Mehring C. Motor task variation induces structural learning. Curr Biol 2009 February 24;19(4):352-357.

35. Morasso P. Spatial control of arm movements. Exp Brain Res. 1981;42:223-227.

36. Abend W, Bizzi E, Morasso P. Human arm trajectory formation. Brain 1982 Jun; 105(2):331-348.

37. Krakauer JW. Motor learning: its relevance to stroke recovery and neurorehabilitation. Curr Opin Neurol. 2006 February; 19(1):84-90.

38. Juntunen ML, Hyvonen L. Embodiment is musical knowing: how body movement facilitates learning within dalcroze eurhythmics. British Journal of Music Education. 2004; 21(2):199-214.

39. Choksy L. The kodaly method. 2nd Ed. New Jersey: Prentice Hall; 1988.

40. Warner B. Orff-schulwerk: application for the classroom. New Jersey: Prentice Hall; 1991.

41. Griffin R. The suzuki approach applied to guitar pedagogy. ProQuest Dissertations and Theses; 1989.

42. Timmermans AA, Seelen HA, Willmann RD, Kingma H. Technology-assisted training of arm-hand skills in stroke: concepts on reacquisition of motor control and therapist guidelines for rehabilitation technology design. J Neuroeng Rehabil. 2009 January;6(1).

43. Rizzolatti G, Luppino G. The cortical motor system. Neuron. 2001 September 27;31(6):889-901.

44. Magill R, Hall K. A review of the contextual interference effect in motor skill acquisition. Human Movement Science. 1990 September: 9(3-5):241-289.

45. Avila H, Cox MT. Case-based plan adaptation: an analysis and review. IEEE Intelligent Systems. 2008 July-August;23(4):75-81.

46. Bengio Y, Courville A, Vincent P. Representation learning: a review and new perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2013 August;35(8):1798-1828.

47. Levin MF, Kleim JA, Wolf SL. What do motor "recovery" and "compensation" mean in patients following stroke? Neurorehabil Neural Repair. 2009 May;23(4):313-9.

48. Stergiou N, Decker LM. Human movement variability, nonlinear dynamics, and pathology: is there a connection? Hum Mov Sci. 2011 October;30(5):869–888.

49. Wolf SL, Lecraw DE, Barton, LA, Jann, BB. Forced use of hemiplegic upper extremities to reverse the effect of learned nonuse among chronic stroke and head-injured patients. Experimental Neurology. 1989;104:125-132.

50. Venkataraman P, Turaga N, Lehrer N, Baran M, Rikiakis T, Wolf SL. Decision support for stroke rehabilitation therapy via describable attribute-based decision trees. International Conference of the IEEE Engineering in Medicine and Biology Society, August 2014.

51. Uswatte G, Taub E, Morris D, Light K, Thompson PA. The motor activity log-28: assessing daily use of the hemiparetic arm after stroke. Neurology. 2006 Oct 10;67(7):1189-1194.

52. Mancinelli C, Patel S, Deming LC, Nimec D, Chu JJ, Beckwith J, et al. A novel sensorized shoe system to classify gait severity in children with cerebral palsy. Conf Proc IEEE Eng Med Biol Soc. 2012;2012:5010-5013.

53. Huang K, Sparto JP, Kiesler S, Siewiorek D, Smailagic A. iPod for home balance rehabilitation exercise monitoring. 2012 16th ISWC. 2012 June 18-22;116-117.

54. Ding D, Ayubi S, Hiremath S, Parmanto B. Physical activity monitoring and sharing platform for manual wheelchair users. Conf Proc IEEE Eng Med Biol Soc. 2012;2012:5833-6.

55. Eckert CM, Clarkson PJ. Planning development processes for complex products. Res Eng Design. 2010; 21:153-171.

56. Smith RP, Morrow JA. Product development process modeling. Design Studies. May 1999; 20(3): 237-261.

57. Ko YT. Optimizing product architecture for complex design. Concurrent Engineering. June 2013; 21(2): 87-102.

58. Eggert RJ. Engineering design. New Jersey: Pearson Prentice Hall; 2005.

59. Ulrich KT, Eppinger SD. Product design and development. Second Edition. Boston: Irwin McGraw-Hill; 2000.

60. Forlizzi J, Battarbee K. Understanding experience in interactive systems. Proceedings of DIS 2004 ACM, New York, NY: 261-268.

61. McCarthy J, Wright P. The critical potential of experience in experience-centered design. CHI 2010, April 10-15, 2010, Atlanta GA.

62. Blythe M, Wright P, McCarthy J, Bertelsen O. Theory and method for experience centered design. CHI EA 2006; 1691-1694.

63. Davis, M. Theoretical foundations for experiential systems design. 2003 ACM SIGMM ETP Proceedings. ACM, New York, NY; 45-52.

64. Sengers P, Boehner K, David S, Kaye J. Reflective design. 4th Decennial Conference on Critical Computing Proceedings 2005; New York, NY; 49-58.

65. Shalloway S, Trott JR. Design patterns explained: a new perspective on object-oriented design. Addison-Wesley; 2002.

66. Zhang X, Hu T, Dai H, Li X, Software development methodologies, trends and implications: a testing centric view. Information Technology Journal. 2010; 9(8): 1747-1753.

67. Dyba T, Dingsoyr T. Empirical studies of agile software development: a systematic review. Information and Software Technology. February 2008; 50: 833-859.

68. Bainbridge WS. Berkshire encyclopedia of human-computer interaction. Berkshire Publishing Group; October 30, 2004.

69. Baskerville R, Pries-Heje J. Short cycle time systems development. Info Systems Journal. 2004; 14: 237-264.

70. Dieli M. The usability process: working with iterative design principles. IEEE Transactions on Professional Communication. December 1989; 32(4): 272-278.

71. Ballard G. Positive and negative iteration in design. 8th Annual Conference of the International Group for Lean Construction. 2007; 17-19.

72. IDEO. Human-centered design toolkit. 2009. http://www.ideo.com/work/human-centered-design-toolkit/

73. Matthews, J 3D printing breaks out of its mold. Physics Today.2001 October; 64(10):25-28.

74. Samuelson, P. Is open-source software the answer? Issues in Science & Technology. 2012;28(3):92-95.

75. Lehrer N, Attygalle S, Wolf SL, Rikakis T. Exploring the bases for a mixed reality stroke rehabilitation system, part I: a unified approach for representing action, quantitative evaluation, and interactive feedback. Journal of NeuroEngineering and Rehabilitation. 2011 August 30;8(51).

76. Lehrer N, Chen Y, Duff M, Wolf SL, Rikakis T. Exploring the bases for a mixed reality stroke rehabilitation system, part II: design of interactive feedback for upper limb rehabilitation. Journal of NeuroEngineering and Rehabilitation. 2011 September 8;8(54).

77. Duff M, Chen Y, Cheng L, Liu SM, Blake P, Wolf SL, et al. Adaptive mixed reality rehabilitation improves quality of reaching movements more than traditional reaching therapy following stroke. Journal of Neurorehabilitation and Neural Repair. 2013 May;27(4):306-315.

78. Wolf SL, Catlin PA, Ellis M, Archer AL, Morgan B, Piacentino A. Assessing wolf motor function test as outcome measure for research in patients after stroke. Stroke 2001 July;32(7): 1635-1639.

79. Chen Y, Duff M, Lehrer N, Sundaram H, He J, Wolf SL, et al. A computational framework for quantitative evaluation of movement during rehabilitation. Conf Proc IEEE Eng Med Biol Soc. 2011;2011:1399-402.

80. Duncan PW, Wallace D, Lai SM, Johnson D, Embretson S, Laster LJ. The stroke impact scale version 2.0. Evaluation of reliability, validity, and sensitivity. Stroke. 1999 Oct;30(10):2131-40.

81. Baran M. Portable torso tracking system: a preliminary study [thesis]. Tempe (AZ): Arizona State University; 2010.

82. Huang CC, Kusiak, A. Modularity in design of products and systems. Systems, Man and Cybernteics, Part A: Systems and Humans, IEEE Transactions. Jan 1998. 28(1): 66-77.

83. Nasreddine ZS, Phillips NA, Bedirian V, et al. The montreal cognitive assessment, moca: a brief screening tool for mild cognitive impairment. J Am Geriatr Soc. 2005; 53(4):695-99.

84. Yesavage JA, Sheikh JI. Geriatric depression scale (gds). Clinical Gerontologist. 2008; 5(1-2):165-173.

85. Fugl-Meyer AR, Jääskö L, Leyman I, Olsson S, Steglind S. The poststroke hemiplegic patient. 1. A method for evaluation of physical performance. Scand J Rehabil Med. 1975;7(1):13-31.

86. iOS Human Interface Guidelines. https://developer.apple.com/library/ios/documentation/userexperience/conceptual/mobilehig/. Updated September 17, 2014.

87. Bradley D, Acosta-Marquez C, Hawley M, Brownsell S, Enderby P, Mawson S. NeXOS – the design, development and evaluation of a rehabilitation system for the lower limbs. Mechatronics. 2009;19:247-257.

88. Colombo R, Pisano F, Mazzone A, et al. Design strategies to improve patient motivation during robot-aided rehabilitation. Journal of NeuroEngineering and Rehabilitation. 2007;4:3.

89. Taub E, Lum PS, Hardin P, Mark VW, Uswatte G. AutoCITE: automated delivery of CI therapy with reduced effort by therapists. Stroke. 2005 June;36(6):1301-1304.

90. Lum PS, Taub E, Schwandt D, Postman M, Hardin P, Uswatte G. Automated constraint-induced therapy extension (AutoCITE) for movement deficits after stroke. Journal of Rehabilitation Research and Development. 2004 May;41(3A):249-258.

91. Holden MK, Dyar TA, Dayan-Cimadoro L. Telerehabilitation using a virtual environment improves upper extremity function in patients with stroke. IEEE Trans. Nueral Systems and Rehabilitation Engineering. 2007;15:36-42.

92. Durfee W, Carey J, Nuckley D, Deng J. Design and implementation of a home stroke telerehabilitation system, IEEE EMBC 2009. Sep. 3-6.

93. Masia L, Casadio M, Giannoni P, Sandini G, Morasso P. Performance adaptive training control strategy for recovering wrist movements in stroke patients: a preliminary, feasibility study. Journal of NeuroEngineering and Rehabilitation. 2009;6(44).

94. Vergaro W, Casadio M, Squeri V, Giannoni P, Morasso P, Sanguineti V. Self-adaptive robot training of stroke survivors for continuous tracking movements. Journal of NeuroEngineering and Rehabilitation. 2010;7(13).

95. Choi Y, Gordon J, Park H, Schweighofer N, Feasibility of the adaptive and automatic presentation of tasks (ADAPT) system for rehabilitation of upper extremity function post-stroke. Journal of NeruoEngineering and Rehabilitation. 2011; 8(42).

96. Kan P, Huq R, Hoey J, Goetschalckx R, Mihailidis A. The development of an adaptive upper-limb stroke rehabilitation robotic system. Journal of NeuroEngineering and Rehabilitation. 2011; 8(33).

97. Littman M. A tutorial on partially observable markov decision processes. Journal of Mathematical Psychology. 2009;53:119-125.

98. Faith A, Ghez C. Accuracy and adaptation to visuomotor gain changes in reaching and reversal movements using auditory feedback of distance and hand speed. Society of Neuroscience, 2013.

99. Sobek II DK, Ward AC, Liker JK. Toyota’s principles of set-based concurrent engineering. MIT Sloan Management Review Magazine. Winter 1999.

100. See J, Dodakian L, Chou C, Chan V, McKenzie A, Reinkensmeyer DJ, et al. A standardized approach to the fugl-meyer assessment and its implications for clinical trials. Neurorehabil. Neural Repair. 2013 Oct;27(8):732-41.

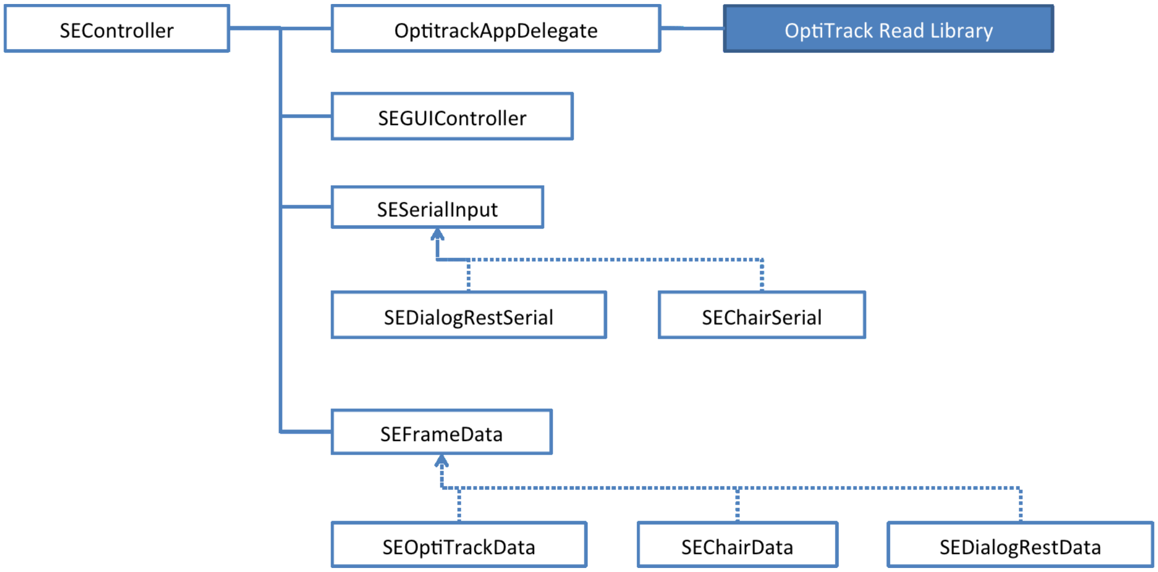
APPENDIX A

DETAILED CODE CLASS DESCRIPTIONS

The following appendix includes detailed class descriptions for each of the plugins.

A1. Sensing Plugin

Design and Implementation by Michael Baran, using an OptiTrack parsing library from Joseph Junker



Sensing Plugin class diagram

This plugin was designed to organize incoming sensor data into objects and send it out such that other modules could use the data. Therefore, the flow of data within the Sensing Plugin is mainly in one direction, with the majority of the data flowing out to other plugins. However, since it is the main gateway for the chair sensor system and system dialog buttons, it also contains interfaces to send messages (primarily on and off commands) out to the hardware.

SEController

SEController is the main controller class of the plugin. It has a couple of core functional responsibilities. This class monitors the status of each input data stream (OptiTrack, chair sensors and Kinect). This entails not only checking if any data frames are being received, but also ensures that the data streams are valid, in the case of the OptiTrack. Since the Tracking Tools does not support single marker labeling, and the wrist marker worn by the patient is a single marker, each data frame needs to be check to see if there are the proper amount of markers seen in the space at any given moment. More details will be discussed about this feature later.

This class is also the main interface for other plugins in some key manners. First it updates local copies of data objects (containing key OptiTrack, chair sensor and Kinect data information) and broadcasts the objects to subscribers via NSNotification. It also handles incoming commands (sent via NSNotification and implemented protocols) from other plugins and routes those commands to hardware (the chair sensors, object sockets and table demo buttons).

OptiTrackAppDelegate

The OptiTrackAppDelegate is the main handler for the incoming data from Tracking Tools. This class interfaces with a separate data parser library written in C by

Joseph Junker, which cleanly unpacks the data sent by Tracking Tools into C/Objective-C objects. This simplifies the process of extracting desired features from the data packet, which extends the usability of this code to other applications. In fact, this module was developed for a separate project and was later integrated into the HAMMR code due to it's demonstrated utility. OptiTrackAppDelegate takes the parsed data objects from the library and checks the information contained within. It verifies that the count of the total number of markers seen in the space is valid. In other words, it checks to see if there are noise or ghost markers as well as if valid markers are missing or occluded from camera view. If the total marker count is correct, it takes the parsed data and stores it into an instance of SEOptiTrackData. If the total marker count is incorrect, the class then attempts to identify from where the marker or multiple markers are missing (the wrist, the torso or the transportable cylinder object). If it can identify where the marker is missing from and make a guess as to where the wrist, torso or object is, it will still update and instance of SEOptiTrackData. If the class cannot safely guess where the missing markers are, it will notify SEController that improper data packets are being received from TrackingTools, and SEController will broadcast this warning out via NSNotification.

SEGUIController

The SEGUIController class was a component added in for debugging purposes as well as manual control of the table object sockets and dialog buttons. The class controls a small GUI with buttons to send on and off related commands to the hardware inside the table. These were the same commands that were sent in response to NSNotification and protocol input to SEController previously described, except the GUI provided manual control.

SESerialInput

SESerialInput is the base class for interfacing with a data stream USB serial input. It abstracts some of the specific properties and general functions needed to initialize a serial port, read from the port and write to the port.

SEChairSerial : SESerialInput

SEChairSerial overrides SESerialInput's read data from port function to have its own unique way to parse incoming chair data into individual sensor readings. It stores these individual readings into an instance of SEChairData.

SEDialogRestSerial : SESerialInput

SEDialogRestSerial overrides SESerialInput's read data from port function to have its own unique way to parse the incoming data from the Arduino controlling the table buttons and object sockets. It stores the parsed data into an instance of SEDialogRestData.

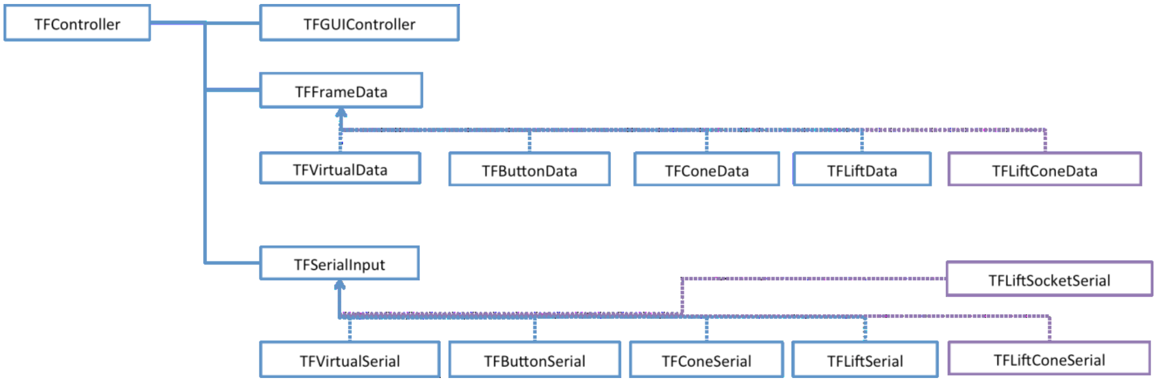
SEFrameData

SEFrameData is the main data object class that packages together all of the parsed data objects from each incoming data stream. It contains an instance of:

- SEOptiTrackData – Contains all parsed marker location information
- SEChairData – Contains all parsed chair sensor data
- SEDialogRestData – Contains all parsed data from table buttons and table object sockets

A2. Tangible Plugin

Design and Implementation by Michael Baran, using data parsing functions by Margaret Duff; All original tangible object code by Margaret Duff; Modifications by Assegid Kidane and Michael Baran



Tangible Plugin class diagram

TFController

TFController is the main class of the tangible plugin. It contains instances of all of the data objects, which it updates and broadcasts out via NSNotification to subscribers. The controller also opens and configures specific serial ports based on protocol input. When a specific object port is needed to be active (with incoming data from the object), TFController sets an opened port as it's primary port and sends all communication messages to that port. TFController also checks to see if the correct objects are connected in specific ports. The class receives a list of expected objects and a particular port via protocol and checks each port for the corresponding object. This check utilizes a handshake method in which TFController sends a specific message to the object port with the expectation of an object specific message back in response.

The other main function of TFController is to handle all commands for data streaming and object light feedback during live system interactions. TFController receives analysis objects via notification, and depending upon the state of particular variables within that data object, it sends specific commands to the active object port. These commands are specific to a particular feedback environment. During real time feedback, white, yellow and red colors are shown in the object in response to live trajectory error. However, during level 2 and level 3 feedback, only a white light go prompt is provided at the start of the reach. TFController knows which feedback environment is chosen via protocol, and then sends specific messages to the objects based on the feedback environment and transition in specific state variables.

TFGUIController

TFGUIController was created as a debugging interface to be able to manually send messages to all of the objects at all locations on the table.

TFFrameData

TFFrameData is the main data class for all of the parsed tangible object input data. It contains an instance of:

- TFVirtualData: Parsed data from the flat, planar objects
- TFButtonData: Parsed data from the slightly elevated button object
- TFConeData: Parsed data from the cone object
- TFLiftData: Parsed data from the cylindrical lift object

Each data object contains the reading from each sensor in the object, as well as an interaction success variable that indicates if the objects has been successfully touched or grasped.

TFSerialInput

TFSerialInput is the base class that abstracts parameters and functions needed for initializing a serial port as well as reading from and writing to the port. This base class is extended into object specific classes:

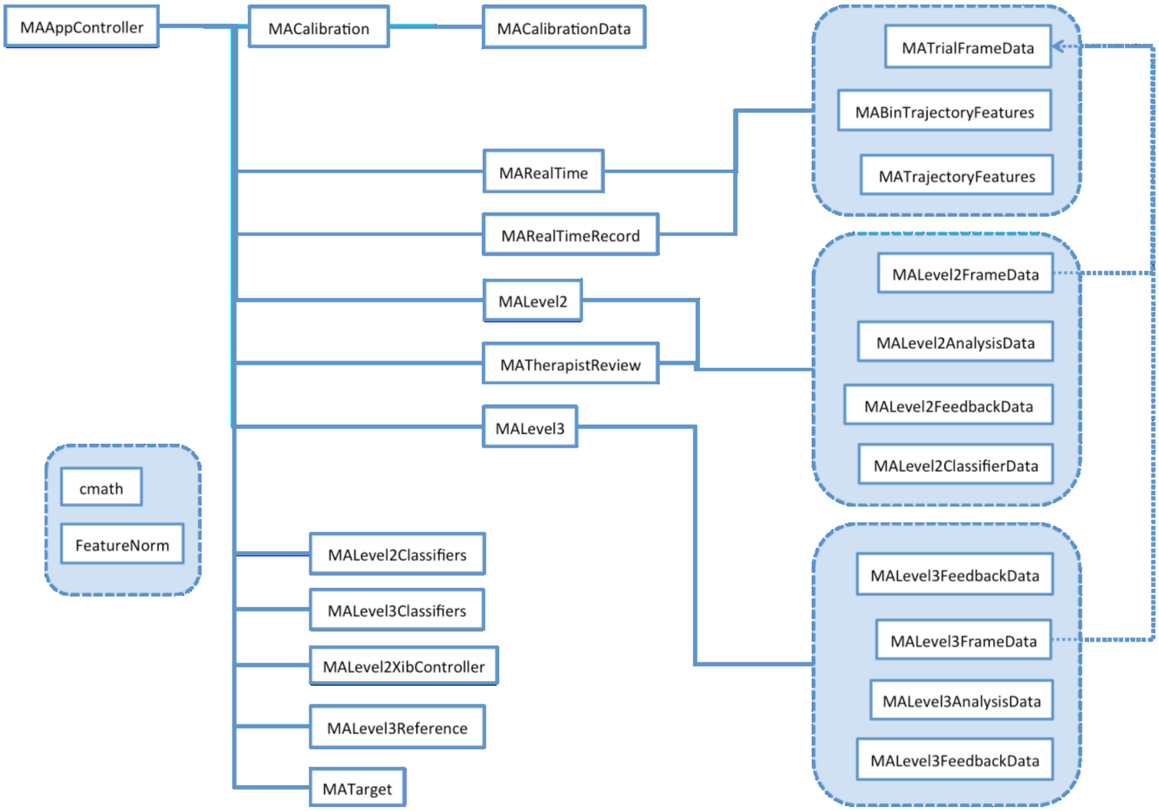
- TFVirtualSerial parses data into TFVirtualData
- TFButtonSerial parses data into TFButtonData
- TFConeSerial parses data into TFConeSerial
- TFLiftSerial parses data into TFLiftSerial
- TFLiftSocketSerial parses data from lift object socket

Each subclass has a unique manner to parse incoming data into its corresponding object specific data class. It also has a unique manner in which to check if the object has seen a successful interaction based on the readings taken from the object sensors. TFLiftSocketSerial is different from the other classes, in that it only parses data from the object in response to the hand shaking method to check if the object is successfully connected in the correct socket. Otherwise, there is no significant data stream to parse as the main object interaction data comes from the TFLiftSerial.

A3. Motion Analysis Plugin

Design by Michael Baran

Implemented by: Rajaram Singaravelu and Michael Baran, using analysis functions from Yinpeng Chen and Long Cheng



Motion Analysis class diagram

MAAppController

MAAppController is the main class of the Motion Analysis Plugin. It loads object based references from file as well their corresponding sensitivity parameters. Based on input from the Adaptation Plugin, it will update specific local task sensitivities from the loaded reference parameters.

During live use of the system, it routes incoming data (from both the Sensing Plugin and Tangible Plugin) to the selected analysis engine. Similarly, it will also route data appropriately when a calibration is being conducted.

MACalibration

MACalibration is an engine that handles marker-based calibrations. It takes routed incoming marker data and generates key calibration features, which it stores into MACalibrationData objects. It also writes calibration data to file for later use. Similarly, it loads calibration data that was previously saved. MACalibration also controlled a small interface for performing a calibration by selecting the target to calibrate.

MARealTime

MARealTime is the real time task analysis engine. It analyzes each frame of camera data to determine the current reach state of the interaction. It also routes data to real time analysis classes for the calculation of frame specific features. MARealTime compiles the resulting frame analysis and features and stores them into data objects. These data objects are then broadcasted via NSNotification. MARealTime also monitors the live interactions to see if the all the reaches for a given set have been completed. Once these tasks have been completed, MARealTime sends out a corresponding NSNotification.

MALevel2

MALevel2 is the analysis engine for Level 2 tasks. It is very similar to MARealTime in terms of its functionality. The main differences are in terms of reach state specific analysis and the types of post reach analysis conducted.

MALevel3

MALevel3 is the analysis engine for Level 3 tasks. It is very similar to MALevel2, with similar corresponding differences previously described.

MARealTimeRecord

MARealTimeRecord is the analysis engine used exclusively for recording data. Incoming data is passed though frame based analysis, the results are stored in data objects, and the data objects are broadcasted out for archiving purposes. In the case of this analysis engine, no reach states are utilized.

MATherapistReview

MATherapistReview is an extension of MALevel2. This analysis engine is used exclusively during training monitoring sessions. The real-time interactivity needed during a training monitoring session is very similar to that required during a level 2 task, except no post task feedback is required. Therefore a specific analysis engine was created for the purposes of training monitoring tasks.

MATarget

MATarget is a data object that holds the entire target specific information, included reference based sensitivities and target calibrations.

MALevel3Reference

MALevel3Reference is a data object which, similar to MATarget, holds level 3 reference based sensitivities which are unique to a particular combination of two objects, as well as the sequence of the objects (aka, which object is interacted with first)

MALevel2AnalysisData

MALevel2AnalysisData is a data object that holds the results of post level 2 set analysis.

MALevel2ClassifierData

MALevel2ClassifierData is a data object that holds the results of the application of level 2 classifiers to a set of data from a level 2 task.

MALevel2FeedbackData

MALevel2FeedbackData is a data object that holds feedback specific parameters for which feedback streams to turn on, as well as their respective sensitivity, based on the results of level 2 classifier analysis.

MALevel3AnalysisData

MALevel3AnalysisData is a data object that holds the results of post level 3 set analysis.

MALevel3FeedbackData

MALevel3FeedbackData is a data object that holds feedback specific parameters for which feedback streams to turn on, as well as their respective sensitivity, based on the results of level 3 classifier analysis.

MATrialFrameData

MATrialFrameData is the base data object that holds all the frame-based features for real time task analysis.

MALevel2FrameData : MATrialFrameData

MALevel2FrameData is an extension of MATrialFrameData that adds on additional level 2 task specific frame based features.

MALevel3FrameData : MATrialFrameData

MALevel3FrameData is an extension of MATrialFrameData that adds on additional level 3 task specific frame based features.

MACalibrationData

MACalibrationData is a data object that contains all of the features generated during a marker-based calibration.

MABinTrajectoryFeatures

MABinTrajectoryFeatures is a data object that holds key post-reach features used in the generation of post reach summary visual feedback.

cmath

cmath is a collection of C functions that perform core mathematical computations used in the higher-level frame based analysis. This class was taken from the AMRR code.

FeatureNorm

FeatureNorm is responsible for taking input marker location data and normalizes the distance of the marker from a trajectory reference based on the value of set zero zone and hull space dimensions.

MATrajectoryFeatures

MATrajectoryFeatures calculates frame based features used primarily in real time interactions, but are also used as a part of higher level analysis seen in Level 2 and 3. These features include the marker’s position along a target rotation axis, real-time speed, and torso compensation.

MALevel2Classifiers

MALevel2Classifiers contains a collection of functions that are applied at the completion of a level 2 task to categorize the movement quality of a set of reaches.

MALevel3Classifiers

MALevel3Classifiers contains a collection of functions that are applied at the completion of a level 3 task to categorize the movement quality of a set of reaches.

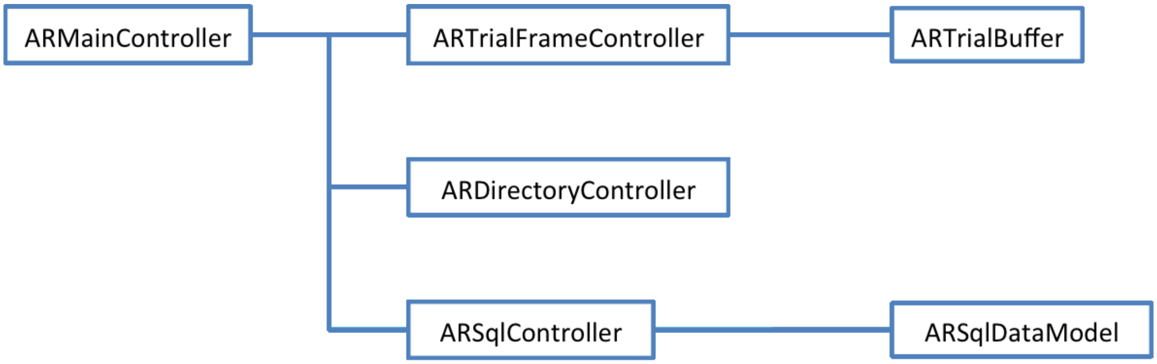
MALevel2XibController

MALevel2XibController was a GUI controller created for debugging purposes to manually trigger level 2 visual and audio feedback to test communication between the Motion Analysis and feedback plugins.

A4. Archiving Plugin

Design by Michael Baran

Implementation by Rajaram Singaravelu and Loren Olson



Archiving Plugin class diagram

ARMainController

ARMainController represents the main class of the Archiving Plugin. It serves as the main interface for all incoming communications (both NSNotifications and protocols). Depending on the type of analysis being completed by the Motion Analysis plugin (this type is set by and incoming protocol), the incoming data frames are routed to a corresponding buffer with it's own respective save method. ARMainController also triggers queries to the database to determine if a session about to begin is incomplete and if so, where the session left off.

ARTrialFrameController

ARTrialFrameController routes the incoming data into specific buffers. There is one buffer for each data frame type. This class also executes the save of each buffer to a database table.

ARDirectoryController

ARDirectoryController handles the directory location for saving a database file.

ARSqlController

ARSqlController creates each database table. It also sets the specific details of each SQL query as well as controlling the specifics of how data is saved to the table.

ARSqlDataModel

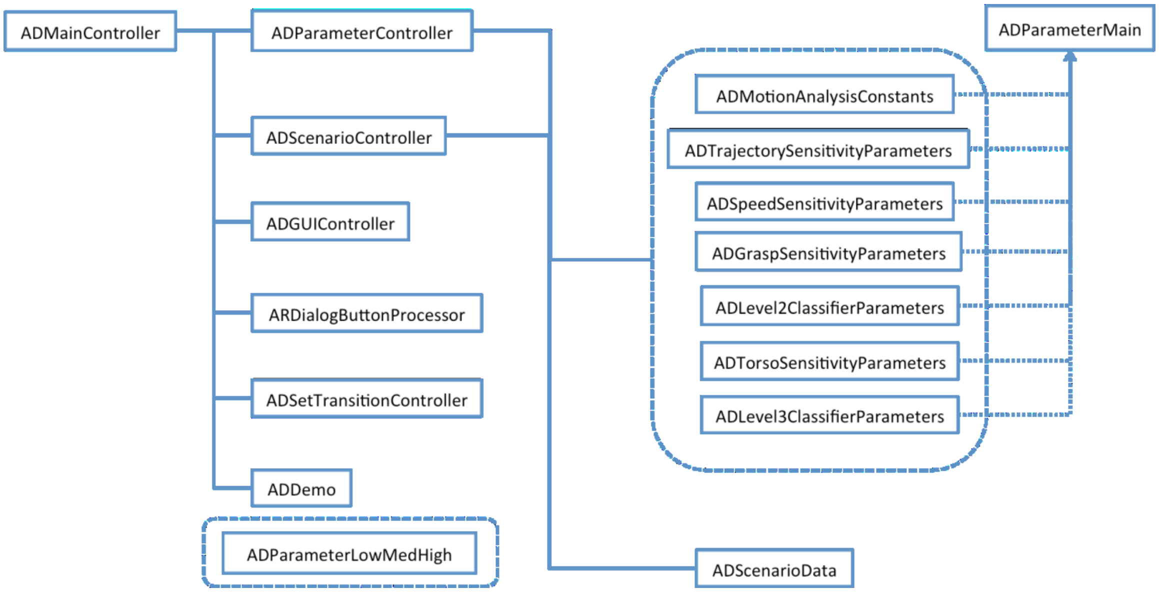
ARSqlDataModel creates the data table structure for the database. The structure is determined by the class properties of the object that is to be saved.

ARTrialBuffer

ARTrialBuffer is the class used for each data frame buffer. It provides additional helper functions for establishing a buffer beyond the core features of NSMutableArray.

A5. Adaptation Plugin

Design and Implementation by Michael Baran



Adaptation Plugin class diagram

ADMainController

ADMainController is the main class of the adaptation plugin. It controls multiple protocols for communication with other plugins (primarily used to setup the properties and sensitivities held within other plugins). It is also the primary handler of both outgoing and incoming NSNotifications for other communications with plugins. The class interfaces with RVR (a video recording program) so that video recording can start and stop with the corresponding start and stop of interactive training sets. The class also triggers some one-off session start features such as the playing of a tangible socket demo, checking of the objects connected at each socket and instruction the Tangible Plugin in regards to which objects are at each port. The class also processes incoming data frames from the sensing plugin to check their state and to then update the GUI with each data streams state.

ADMainController houses the main set used at startup of HAMRR. This function triggers multiple functions, including the loading of a scenario file, loading marker calibrations and querying the database for which set of the session to start first. Similarly, the plugin also resets key parameters when a set is complete in preparation for the next set to begin. In order to determine if the interaction of a current set is complete, and thus move on to the next set, ADMainController listens via NSNotification for "all complete" messages from other plugins in order to determine if all of the plugins are ready for the next set to begin.

ADMainController also receives, and parses, incoming data from the table buttons and interprets the results in terms of the progression or replay of demos as well as the progression of training sets. Similarly, the class also serves as the main control behind the playback and sequencing of demo videos. It determines which, if any, of the full demos (which are a full detailed video walkthrough of the task or feedback) are needed. It checks to see if a full demo for a particular task or feedback element of the upcoming set has been played. If not, ADMainController notifies the Visual Feedback Plugin (via NSNotification) that a particular full demo needs to be played. If the full demo has been played previously in the current session, it notifies the Visual Feedback Plugin to play a specific demo prompt (a text only reminder of an aspect of the task or feedback). Relatedly, ADMainController also keeps track if feedback streams have been turned off in between sets. If it detects this feedback transition, it notifies the Visual Feedback plugin to notify the patient via a text prompt that a particular feedback stream will not be presented.

ADDialogButtonProcessor

ADDialogButtonProcessor provides a method for parsing incoming data frames from the table buttons to determine the input selection from the patient.

ADSetTransitionController

ADSetTransitionController contains methods to determine if specific plugins have all reported that they have completed their respective functions. This is needed to determine if the next set can be started.

ADDemo

ADDemo holds information on the total demos available and if the specific demos have been previously played during the session.

ADGUIController

ADGUIController controls the main GUI interface of HAMRR. This interface allows a system controller to manually update sensitivity parameters as well as key set task parameters.

ADScenarioController

ADScenarioController loads an XML file and parses the contained information into ADScenarioData objects. It updates ADParameterController with sensitivities contained within a specific set's corresponding ADScenarioData instance. ADScenarioController also monitors the sequence of sets in terms of which one should be selected at the start of a session, which one is next, and when a session is done (all sessions are complete). At one time, the class had additional log capabilities, but this component's development was abandoned after it was determined that the information contained in the log would not be of high priority.

ADScenarioData

ADScenarioData is the main sensitivity data object. It holds all of the highest-level task and sensitivity parameters for a given interactive set.

ADParameterController

ADParameterController loads all task sensitivities (excluding reference trajectories) from file and sorts then into specific dictionaries for later access. It updates the sensitivities of specific parameter objects based on the values contained within the corresponding ADScenarioData object for a given set.

ADParameterMain

ADParameterMain contains abstract functions used in subclass parameter objects, which is primarily the manner in which sensitivity parameter values are loaded from file.

ADMotionAnalysisConstants : ADParameterMain

ADMotionAnalysisConstants holds parameters that are not sensitivity specific. These parameters are usually held constant across patients and sessions.

ADSpeedSensitivityParameters : ADParameterMain

ADSpeedSensitivityParameters holds all speed related sensitivity parameters.

ADTrajectorySensitivityParameters : ADParameterMain

ADTrajectorySensitivityParameters holds all trajectory related sensitivity parameters including rest and grasp zones and trajectory zero zones and hulls.

ADGraspSensitivityParameters : ADParameterMain

ADGraspSensitivityParameters holds all object interaction related sensitivity parameters.

ADTorsoSensitivityParameters : ADParameterMain

ADTorsoSensitivityParameters holds all torso compensation related sensitivity parameters.

ADLevel2SensitivityParameters : ADParameterMain

ADLevel2SensitivityParameters holds all level 2 classifier related sensitivity parameters.

ADLevel3SensitivityParameters : ADParameterMain

ADLevel3SensitivityParameters holds all level 3 classifier related sensitivity parameters.

ADLevel2LocationParameters

ADLevel2LocationParameters represents the object that actually stores the values found with ADLevel2SensitivityParameters. It was created so that object location specific sensitivity values could be specified for the level 2 classifiers.

ADLevel3LocationParameters

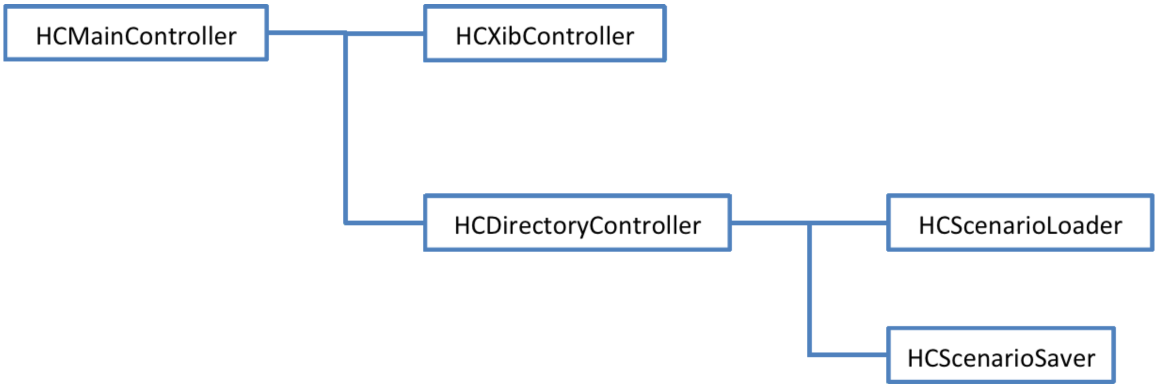
ADLevel3LocationParameters represents the object that actually stores the values found in the sensitivity parameter classes. It represents how each parameter is defined for low and high sensitivity. It originally had the capability to hold a medium sensitivity value as well, but this feature was removed when binary sensitivity was decided for the final implementation of HAMRR.

Adaptation Plugin GUI

The Adaptation Plugin GUI serves as the main GUI for the use of HAMRR. It is divided into three main panels and a header.

A6. HAMRR Composer

Design and Implementation by Michael Baran



HAMRR Composer Class Diagram

HCMainController

HCMainController instantiates and connects the main components of the composer program.

HCXibController

HCXibController controls the GUI interface to create and edit scenario XML files. The GUI has controls for each of the parameters within ADScenarioData. Scenarios can be crafted by adding, removing or editing sets.

HCDirectoryController

HCDirectoryController creates and maintains the directory for where the scenario XML files are loaded from and saved to.

HCScenarioLoader

HCScenarioLoader contains the methods to load XML files and parse the information into ADScenarioData objects.

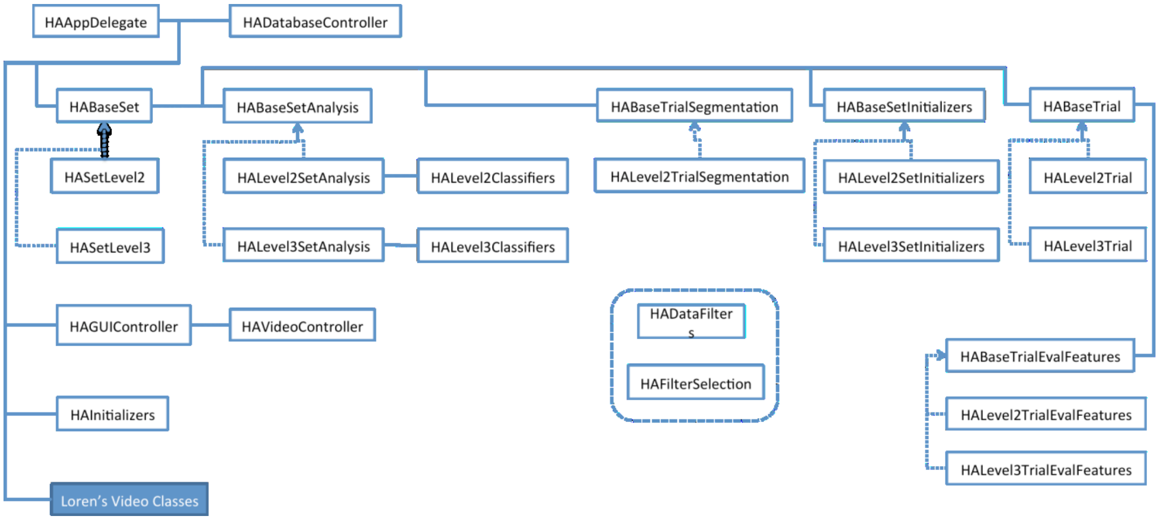
HCScenarioSaver

HCScenarioSaver is the converse of HCScenatioLoader: it contains the methods to save ADScenarioData objects into XML files.

A7. HAMRR Analysis Program

Design by Michael Baran

Implementation by Michael Baran, using some functions by Rajaram Singaravelu and Yinpeng Chen



HAMRR Analysis Class Diagram

HAAppDelegate

HAAppDelegate is the main class of the analysis program. It loads all of the sensitivities from file (similar to the functionality seen in the Motion Analysis Plugin). It selects which database table to load based upon the selected analysis type. It also queries the database to load data. HAAppDelegate also create an object that represents a set of data, and update the object with appropriate sensitivity values based on corresponding sensitivity values loaded from an ADScenarioData object.

HADatabaseController

HADatabaseController handles all queries and functions related to pulling data from the database.

HAInitializers

HAInitializers contains all of the methods to load reference trajectory specific information used in later analysis.

HAVideoController

HAVideoController controls a small video GUI window for playback of patient recorded videos. It automatically loads the corresponding video for the current data se that is being analyzed in the main GUI.

HAGUIController

HAGUIController controls the main GUI interface for the analysis program. It has specific controls to load database files and select a particular set, as well as a property within the set, to visualize. It has a table view that controls the main segmentation points for the trials within a set. Another table view is used to show the results of the trial and set-level evaluations. The HAGUIController also has the main controls to segment and analyze a set. It also features a separate table view to add different data filters to the set of data. Finally, there are controls to load separate calibrations that what is contained in the main loaded database file. This feature is required for when a calibration wasn't completed during a session or if the calibration was capture incorrectly.

Visualization Classes: Implemented by Loren Olson

Utilized by HAGUIController are a series of classes implemented by Loren Olson to assist in the visualization of data in a graph. These classes translate data magnitudes into the range of pixels within a window and handle all of the plot drawing. In addition to these base classes, I added a small functionality component to visually zoom into a range of data based on mouse input.

HABaseSet

HABaseSet is the main data object that contains all of the necessary data and parameters for a given set. It contains all of the data frames for a set. It holds the rest and target calibrations for a set. It also holds the corresponding ADScenarioData object, which is loaded from the database. It holds instances of a trial segmentation object, a set initialization object and a set analysis object. This class also contains the main methods to write data, including analysis results, to file as well as methods to load previous applied segmentations and filters.

HABaseSet : HASetLevel2

HASetLevel2 is a subclass of HABaseSet. It has a level 2 specific initialization object, set analysis object, and trial segmentation object. It also has unique methods to save evaluation features and save/load segmentation points.

HABaseSet : HASetLevel3

HASetLevel3 is a subclass of HABaseSet. It has a level 3 specific initialization object, set analysis object, and trial segmentation object. It also has unique methods to save evaluation features and save/load segmentation points.

HADataFilters

HADataFilters provide class methods for applying the following filters: median, data shift (shifting a selection of data by a uniform amount), average smoothing, and peak removal (removing data between two points and replacing the data with interpolation between two points).

HAFilterSelection

HAFilterSelection is the specific object that represents an applied filter. It contains the filter selection as well as general methods to apply and revert applied filters. It also contains the methods to save the details of applied filters to file.

HALevel2Classifiers

HALevel2Classifiers contains all of the level 2 classifier analysis methods.

HALevel3Classifiers

HALevel3Classifiers contains all of the level 3 classifier analysis methods.

HABaseSetAnalysis

HABaseSetAnalysis contains all of the core analysis methods for a set of data. It is primarily used to analyze level 1 task as well as sets of data that were collected using the MARealTimeRecord analysis engine.

HALevel2SetAnalysis : HABaseSet

HALevel2SetAnalysis is a subclass of HABaseSet. It overrides the base analysis methods with specific implementations for level 2 analysis.

HALevel3SetAnalysis : HABaseSet

HALevel3SetAnalysis is a subclass of HABaseSet. It overrides the base analysis methods with specific implementations for level 3 analysis.

HABaseTrialSegmentation

HABaseTrialSegmentation is the base class for the core segmentation methods for a trial.

HALevel2TrialSegmentation

HALevel2TrialSegmentation is a subclass of HABaseTrialSegmentation. It overrides some of the segmentation methods with specific implementations for segmenting level 2 trials.

HABaseSetInitializers

HABaseSetInitializers contains all of the core methods for initializing a base set of data. Primarily it is used to separate the data into trials and run some normalization of the data.

HALevel2SetInitializers : HABaseSetInitializers

HALevel2SetInitializers is a subclass of HABaseSetInitializers and overrides some methods for level 2 specific initialization.

HALevel3SetInitializers : HABaseSetInitializers

HALevel3SetInitializers is a subclass of HABaseSetInitializers and overrides some methods for level 3 specific initialization.

HABaseTrial

HABaseTrial is the main data object that holds all of the data and pertinent parameters for a trial of data. It contains the location for all of the segmentation points for the trial. It also contains all of the raw data for the whole trial. It also holds all of the results for trial level and set level evaluations. HABaseTrial also holds an array of applied filters for the trial. It also contains the methods to save trial properties to file.

HALevel2Trial : HABaseTrial

HALevel2Trial extends HABaseTrial to hold level 2 specific trial evaluations.

HALevel3Trial : HABaseTrial

HALevel3Trial extends HABaseTrial to hold level 3 specific trial evaluations.

HABaseTrialEvalFeatures

HABaseTrialEvalFeatures contains the results of all of the base trial evaluations.

HALevel2TrialEvalFeatures : HABaseTrialEvalFeatures

HALevel2TrialEvalFeatures contains the results of all of the level 2 trial evaluations.

HALevel3TrialEvalFeatures : HABaseTrialEvalFeatures

HALevel3TrialEvalFeatures contains the results of all of the level 3 trial evaluations.

APPENDIX B

SYSTEM DESIGN AND IMPLEMENTATION DETAILS

B1. Design and Implementation of a Transportable Cone

Because all of the objects were 3D printed, a new, basic object could be created in minimal time. The stationary cone design was modified only slightly so that it could now interface with the transportable object table sockets at the base as well as contain a battery, XBee, two FSRs and a single IR led at the top. Since this was a prototype, the functionality was stripped down to the core essentials: the system needs to know where the object is in the table space at various moments (IR light) and it would be helpful to detect a grasp of the object (two FSRs). These decisions were made in order to keep the study moving forward and collect data from more impaired subjects. The quick creation of a new object demonstrates the validity utilizing prototyping tools, as previously discussed, in the iterative development of an INR system.

From the perspective of the HAMMR code, the integration of this new object only required a few straightforward changes due to the modular design of the software architecture. Within the Tangible Plugin, a new tangible object TFSerialInput was extended so that it could use all of the standardized input, output and data processing functions. TFFrameData was extended to include the new FSR sensor data from the transportable cone. Within the Adaptation Plugin, the new object was added to the list of possible objects (through ADScenarioData) to use in training. This also allowed HAMRR Composer to include it in composed sets, and therefore easily replace all of the instances of transportable cylinder in previously composed protocol paths.

Within the Motion Analysis Plugin, some adjustments were made to the analysis reach states and calibration procedures. Much like the cylindrical lift object, the system needs to know where the transportable cone is in space to determine if the patient had configured the space properly as well as lifted the object high enough in certain training tasks. With the cylindrical lift object, this is done through the four markers attached to the top surface. However, this same design could not be completed with the transportable cone. TrackingTools cannot differentiate and label individual markers. They have to be part of a rigid body, requiring at least three markers to be grouped together in a physically stable orientation. If this were created for the top of the cone, it would be cumbersome and prevent the patient from sliding their hand over the top of the cone to grasp. Therefore, the transportable cone was designed with a single IR light on top of the cone that could be turned on and off by HAMRR during certain reach states. During this time, two loose markers would be seen by HAMRR, but it would check to see that one was within a target calibration, and the other was within a rest zone calibration (assuming that before a reach starts the locations of these two points would be very distinct). If both of these conditions were true, the IR light atop the cone would be shut off, and the reach states would progress as normal since the system was now seeing the correct number of markers in the space. If after a fixed amount of time, the conditions were not met, an instructional prompt would be presented on the screen instructing the patient to double check the transportable cone location and move it to where it needed to be. As can be shown, because the software of HAMRR was modular and extensible along the previously mentioned guidelines of software design, the new object was integrated into the code with minimal new code written. However, improvements to the motion analysis component would be an important consideration for future work.

Before the object and updated code was released for patient testing, it went through a series of internal tests with the design team. In the testing of the object, one problem was encountered. When data was streaming out of the object from the FSR sensors, it was difficult to timely and reliably communicate messages to the object. In other words, if the object was actively streaming out data, it would sometimes miss an "IR Off" command from the Tangible Plugin, and thus stall the progression of the live interaction during a set of activity as the Sensing Plugin then saw two loose markers (one for the wrist and one for the cone) in the space, where there should only be one (just the wrist marker), and therefore stopped the set thinking there was errant noise in the camera data. After significant testing to try and improve the speed of sending and receiving messages, it was decided to remove the data streaming from the object completely. This provides an example of utilizing engineering design approaches within iterative design. Engineering design conducts validation of systems with clear metrics. In this case, the metric was known for what delays in the interaction of the system are acceptable and which are not. The transportable cone data streaming created too much of a lag to be useable.

We suspected that the lag was due to a wireless communication protocol that was different from the other wireless communication used in HAMRR. This new method was chosen due to the simplicity of the hardware electronics required to build the new object, however the team had no previous experience with this form of communication within the context of HAMRR. Therefore, in order to keep the study moving forward and because two sensors will not provide the same detailed grasp information that could come from a larger array of sensors, the data streaming was removed. The ability to automatically track the object in the space was deemed much more important than tracking the grasp of the object for the completion of this phase of the study. As a result the grasp conditions for interactive training for the object were modified to be based on the total speed and location of the wrist marker with respect to the object.

B2. Adaptation Control GUI Redesign Details

The HAMRR interface, which is part of the Adaptation Plugin, was built to offer a GUI interface to the underlying parameters of the therapy experience. However, it was never built with any user in mind, other than one of the system designers. As can be seen in the previous discussion of how the user experience was connected to software functional decomposition, there was minimal discussion of the therapist as a primary user. This was a result of the many design iterations that needed to be directed elsewhere and the momentum that was building towards having a system designer run the system for the six-patient study in order to get the cleanest data in the shortest amount of time. However, it became clear that an additional two subjects, who were more impaired, would be helpful to add to the study but one of the system designers would not be present to run the system. As experience design approaches with the therapist in mind were not used in the initial design of the GUI, the experience of the therapist controlling them system had to be reviewed in detail and resulting changes to the system needed to be made.

First, an increase in documentation was required, accompanied with in-person training to use the system. This required walking through the system and creating a detailed textual summary describing, step-by-step, how to run HAMRR. This level of detail was required as the therapists had low technological experience and needed the HAMRR system instructions presented in context of Mac OS interface instructions. As a result, it became readily clear that the interface was not ideal and in fact could provide some unnecessary confusion. However, a full GUI redesign was not practical in the time between the two study sections. Therefore, a priority was given to clarifying the setup of the system, and mainly the need to calibrate different objects, which were identified as the two primary interactions the therapist would need to conduct.

Currently, HAMRR requires a calibration of a patient touching or grasping an object to run any analysis for reaches to that particular object. Therefore, the therapist needs to calibrate each target that is used during the session. While there is a text print out guide to tell the therapist which objects will be used in each session, there was difficulty encountered translating this information to the calibration interface within the Sensing Plugin. Therefore, since the Adaptation Plugin loads pre-composed scenarios, with the target selection pre-set, this information could be used to assist the therapist.

An iteration on the GUI design was made in which within the calibration interface, the targets that required calibration were highlighted in a blue color, and the objects that had been successfully calibrated were green. Therefore, the therapist could quickly see which objects had been calibrated and which ones still needed to be completed.

In addition, tangible data was added as a requirement for calibrating an object. Previously, the system controller had to visually confirm that the object was properly engaged when conducting a calibration. Since the therapist would already have their focus in multiple directions, it was decided to make tangible data part of the calibration success. Therefore, the patient needed to be successfully touching or grasping the object when calibrating. In terms of the new calibration interface, if the calibration process was underway but the object was not successful engaged, the target was highlighted yellow in the GUI. Once the patient successfully touched or grasped the object, the calibration would complete and the target in the GUI would update to green.

B3. Problems Encountered with Torso Tracking

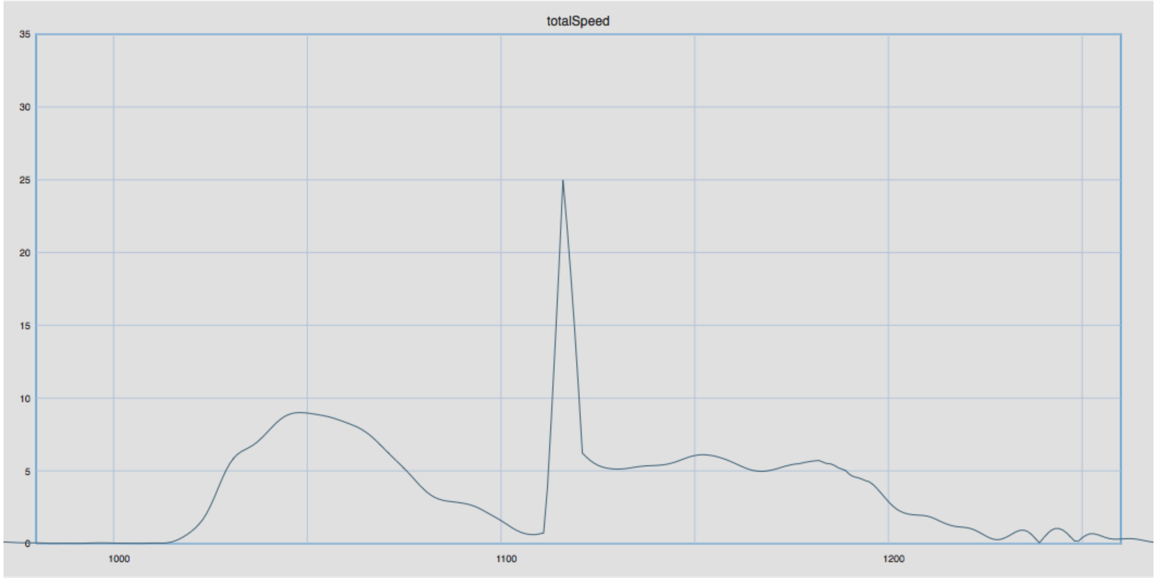
In the previous AMRR study, torso compensation was measured by monitoring the angle orientations of the marker rigid body plate worn on the back of the patient. Through these measurements, the system could determine the maximum lean or twist of the rigid body plate from a calibrated starting orientation.

The idea of torso tracking with HAMRR was the same. Once the Kinect torso tracking component was eliminated from the system, a rigid body plate was introduced for subjects of HAMRR to wear slightly below their left shoulder. This location was chosen as a place on the front of the body (so it could be seen by the array of cameras) that was least susceptible to movement (twisting or leaning that could occur as a natural byproduct of the movement and therefore was not body compensation that should be measured). A simple compensation algorithm was created that would determine the difference in angle orientation for a given frame of data in comparison to the orientation of the plate at the calibrated rest position. While this was conceptually the same as the AMRR system, there were some crucial differences and oversights that lead to problems.

Location of the Rigid Body and Occlusion

In initial testing of the rigid body, while the stability of the rigid body during non-compensating reaching movements was deemed acceptable, the rigid body markers were found to become occluded with severe compensation, which was especially noted when testing the system with the more impaired subjects. Even in more mild compensating cases, some minor occlusion was found as well.

The main problem in both cases was the inconsistency in Tracking Tools ability to find the rigid body again once the occlusion was removed. In most cases, the rigid body could be found, and normal data recording proceeded. However, in some cases, when the rigid body was once again identified by the cameras, Tracking Tools would incorrectly find a double or “ghost marker” in place of one of the markers of the rigid body. This caused the system to pause the data recording as it thought there was a different rigid body in the space than what it was expecting to see.



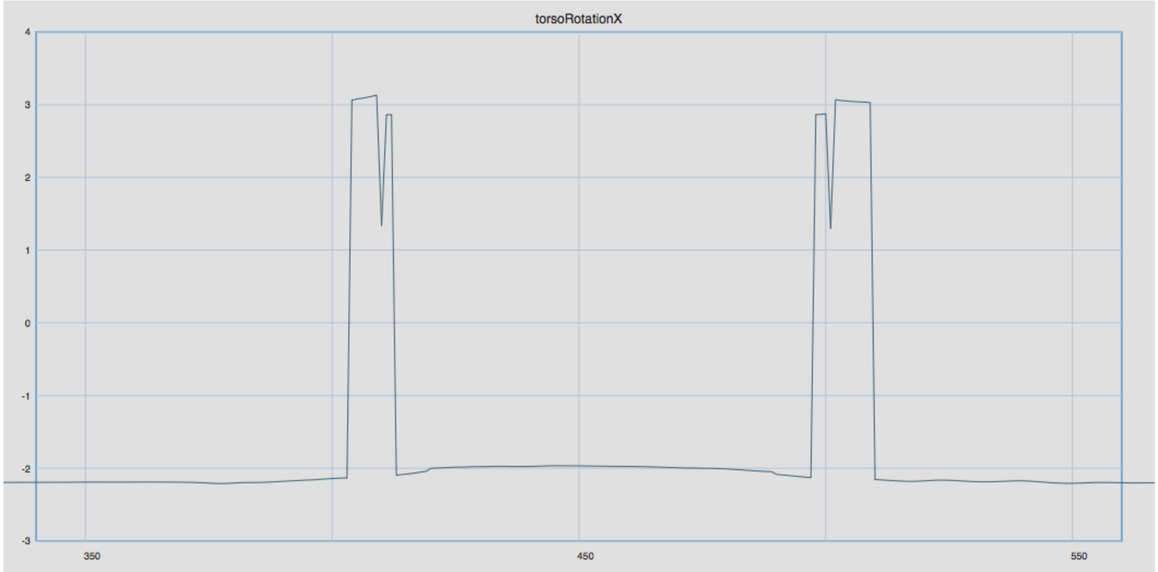
Truncated reach due to marker occlusion. A noise peak marks where data was dropped from a key turn task.

Determining the Starting Orientation of the Rigid Body

One issue that was not found until late data analysis was that determining the starting orientation of the torso rigid body plate was entirely reliant upon the orientation in which the rigid body was in upon creating the rigid body in Tracking Tools. When creating a rigid body in Tracking Tools, a group of markers are selected and the program automatically assigns the current orientation as the zero orientation (each axis of rotation reports a rotation angle of 0 degrees). However, this was not known during testing of the system, and therefore a protocol for maintaining consistent orientation of the rigid body during calibration was not enforced. This means that orientation (and therefore reported angle rotations) would have different magnitudes and directions based on the different orientations the object might have had during calibration of the object.

Determining the Current Orientation of the Rigid Body

Upon detailed review of the data, the angle rotation data had frequent large shifts. The angle rotation data would show significant jumps in the data that did not follow possible logical magnitudes (such as factors of pi). Upon further analysis, it was found that the version of Tracking Tools used reports angle orientation quaternions different than traditional orientation estimations. Therefore, the standard quaternion to angle estimations functions were incorrect and would report false and inconsistent angle orientations that could not simply be fixed by removing the shifts (as the shifts were very inconsistent in magnitude and occurrence).



Example of angle rotation shift. This example demonstrates how inconsistent the shift was and hides the original signal of torso rotation about the x axis.

Momentary Occlusion Leading to Marker Swap

During most training procedures there were five markers in the space: one on the wrist and four on the torso plate worn below the left shoulder. If the task called for transporting the cylindrical object, then this number increased to nine, with four new markers located on the cylindrical object.

The problems seen with this setup, combined with the Tracking Tools software, were two fold. First, Tracking Tools is not able to track unique individual markers. The software is built to track unique rigid bodies. In the design of the system, we looked to avoid putting a rigid body on the wrist in order to not encumber the patient during reaching activities (and as seen in the noisy nature of the key turn data, which used a wrist-worn rigid body, the use of a rigid body does not exclusively eliminate problems). However, the Tracking Tools software will report 3D position of all markers in the camera space, regardless if it is part of a rigid body or not. Therefore, an algorithm was designed to check each marker to see if it was part of a defined rigid body (a given marker had the same location as one in a defined rigid body). If a marker did not satisfy the check, then it must be the wrist marker. The concept of this solution will hold true if no noise is introduced into the space (such as from an errant reflective material). However, more frequent was marker confusion when a rigid body left and re-entered the camera space.

Many times when a subject would significantly compensate (or the transportable cylinder object was lifted above the camera space) markers were not detectable by the cameras. When the rigid body would come back in view, one of the markers was not drawn in the correct position with respect to the rest of the markers. Rigid bodies are defined with fairly rigid geometries, and therefore when this mis-position occurred, the system did not think it was the correct rigid body. This resulted in five “loose” markers in the space. This could be remedied by a staff member by temporarily covering and uncovering the markers, but obviously was not a procedure that could be completed by the subject. With more impaired subjects, who will provide movements that can lead to frequent occlusions, a better tracking solution will be needed.

B4. Implementation Details for Standardizing Communication Between Plugins

Overall, the communication between plugins was structured in the following way. Protocols were used for setup of the plugins (for example, when the Adaptation Plugin needed to communicate updated sensitivities to Motion Analysis or either of the feedback plugins) and notifications were used for real-time, low-level communication (such as processed frames of data from the Sensing or Tangible Plugin). This structure seemed to be very clear and stable and should be a standard fixture of HAMRR plugins. However, further abstraction should be introduced.

Currently, there are NSNotifications used which also include data objects (SEFrameData, TFFrameData, MATrialFrameData, MACalibrationData). In the case of each, as soon as an update is made to these data objects, they get sent out to a group of subscribers. This is a slightly modified version of the Observer Pattern, which in this case the Subject is not aware of the specific Observers, since the Subscribers just connect to a particular NSNotification via a unique name. This model was a reliable way to send real-time data to a variety of plugins without the Subject needing to know who needs the data, which helps with extensibility and should be maintained going forward.

However, as previously discussed within each of these NSNotifications is a data object. Typically, the Observer would receive the object and then unpack the object based upon specific needs. For example, the MARealTime analysis engine would receive a SEFrameData object and get the value of particular properties (such as the X, Y and Z location of the wrist marker) for its own analysis purposes. However, this required that the MARealTime class had to know how the SEFrameData was composed, and as a direct result, had to import the header file for the SEFrameData class. This begins to tangle up the plugins together and may limit future extensibility if a future developer wanted to develop their own analysis module.

What I propose is that an NSNotification should be sent out to notify subscribers that a new frame of data is available. Those subscribers can then use a predetermined protocol to get the information from the updated data frame from the plugin responsible for controlling the data.

As an example of this implementation, let's return to the previous example of the Sensing Plugin sending updated data to the Motion Analysis Plugin. In the updated implementation, the Sensing Plugin would broadcast a notification that a new frame of data is ready (once it parsed the incoming multicast data from OptiTrack). The Motion Analysis Plugin, being a subscriber of this notification, would call a series of functions within a Camera Data Frame protocol, implemented by the Sensing Plugin, in response. The functions would act as "getters" and return specific values for the X, Y and Z location of the wrist marker (among other data from the camera data frame as well).

The benefit of this setup is that now the Motion Analysis Plugin does not need to know how the data is structured within the Sensing Plugin. It is still beneficial for the Sensing Plugin to separate data into objects so that it can cleanly be parsed and accessed. However, in the new implementation, all the Motion Analysis Plugin needs to know is that by using the functions in the Camera Data Frame protocol, the values for specific properties will be returned. It no longer needs to know how the data is stored. Rather it knows, by calling the functions within the protocol, the values will be returned. Now, instead of importing multiple header files that are data object implementation specific, it can import just one header file from the Sensing Plugin that defines the protocol.

It is important to note that the overall NSNotification flow of information should be maintained. First it sets up a model such that every time a new frame is updated it is broadcasted, and therefore Observers can respond to new information automatically on a frame by frame basis. This is crucial for HAMRR as the marker camera frame is the main clock of the HAMRR interaction experience. Secondly, the Sensing Plugin and Tangible Plugin should not need to be concerned with where specifically the data is going. These plugins are designed to parse raw data streams into organized structures and notify other plugins when new data is ready. Therefore, maintaining NSNotifications, coupled with protocols, would maintain the benefit of NSNotification timing with the option for easier extensibility.

A similar review of the protocols currently in use is also required. Currently, protocol are used as a way for the Adaptation Plugin to update other plugins with new sensitivities or one-off commands. These protocols are typically called between sets or at the start of a session, and therefore are called with significantly less frequency than the NSNotifications. This difference in frequency was one of the primary reasons protocols were used. In addition, by implementing a protocol, a class is required to have implementations for all of the contained methods, which supports code stability. However, there were some limitations in the way protocols were used within HAMRR.

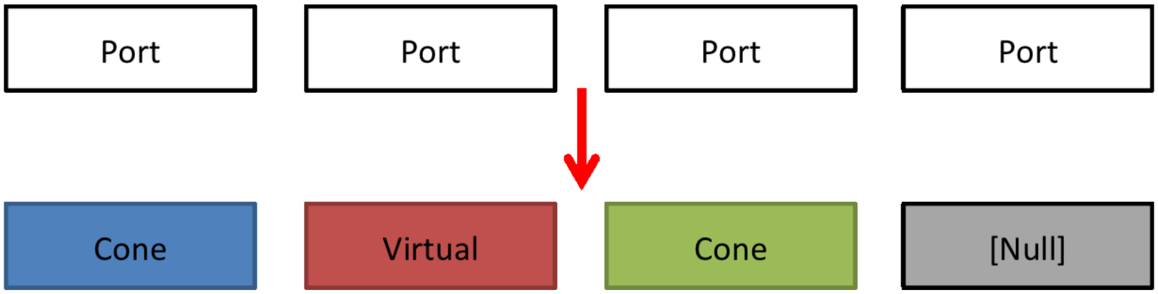
Similar to the NSNotification implementations, with many of the protocols there are included data objects, of which the implementing class needs to know the structure. For example, before a set begins, the Adaptation Plugin uses the ADParametersProtocol to have the Motion Analysis Plugin update its current sensitivity values based on those required by a particular set (represented by an ADScenarioData object). Currently, each method within this protocol passes along one of the ADParameterMain sub-classes. Similar to the limitation of the current NSNotification implementation, this requires the Motion Analysis plugin to understand the structure of an Adaptation Plugin data object.

What I propose is very similar to what was proposed to clean-up the use of NSNotification: the Adaptation Plugin should notify specific plugins that a set is about to begin and in response, each plugin should ask the Adaptation Plugin for needed sensitivity values. Protocols should still be used, as this will help ensure that each plugin is asking for all of the values it needs, plus it requires the Adaptation Plugin to specifically communicate with each specific plug in, thus helping to ensure that each plugin is receiving the important comment. However, slightly different protocols should be implemented.

The Adaptation Plugin should have a standard protocol to notify specific plugins that the set is about to begin. Then, in implementing this protocol, each plugin should ask the Adaptation Plugin for specific sensitivity values, which are going to be plugin specific. The Adaptation Plugin shouldn't dictate which sensitivity parameters need to implemented, it should instead give plugin a prompt to update any sensitivity values prior to starting a set. Therefore, the Adaptation Plugin would implement a series of protocols to return sensitivity values back to the calling plugins. In this way, there is further plugin code separation (just as with the suggestion for the new NSNotification implementations), which helps future extensibility. Plus this will also make the protocols more contained and specific. In contrast many of the current protocols are not completely implemented by a particular plugin as not all of the contained methods are relevant, which is indicative of a bad design. Clearer use of protocols would help remove this ambiguity.

B5. Implementation Details for Increasing Flexibility in Tangible Plugin Data Parsing Interface

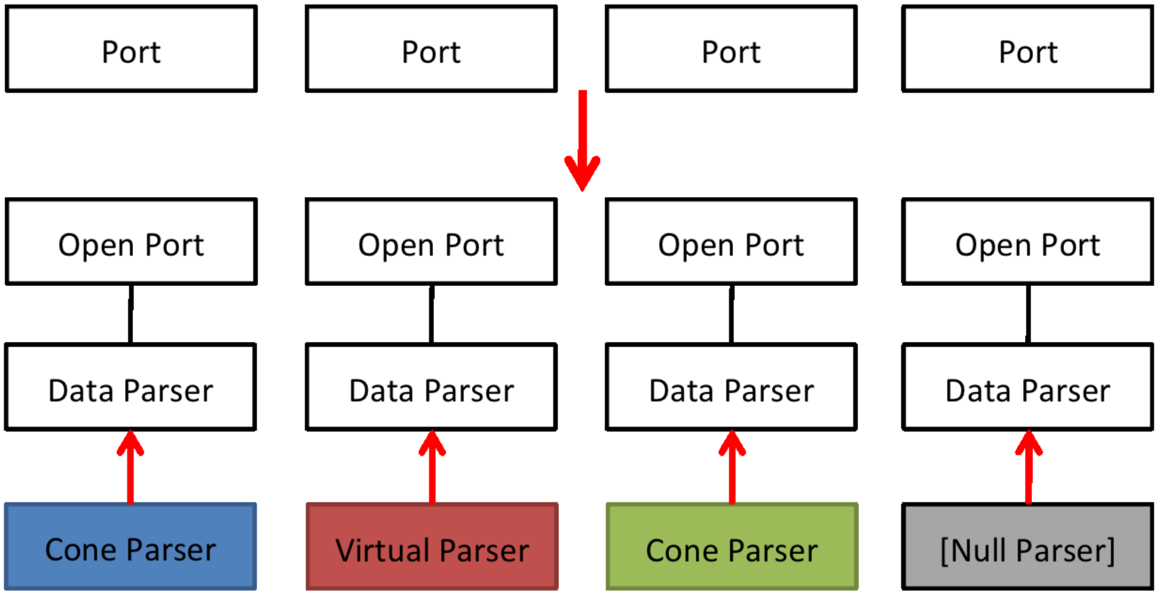
In the development of the system, many problems were encountered when developing the Tangible Plugin interface for receiving data from the tangible objects. If there was ever a crash from any of the plugins, the serial ports were not closed properly, and as a result, the computer would need to be forcefully shutdown and restarted. To avoid this problem at the time, it became necessary to run DASH as a compiled app and incorporate functions for a clean exit that could respond to thrown exceptions or commands to quit the program to safely close the ports. As a result of the problems seen in the stability of the serial port communication, the Tangible Plugin was configured such that only a fixed number of ports could be opened every time DASH was started. This ensured that a specific port could only be opened once and it would have to be safely closed via the port closing resulting from quitting HAMRR. While this created stability that reduced the number of crashes and also vastly improved the recovery time from a crash, it was not an ideal solution for the long term.



Current Tangible Plugin setup with fixed ports

First of all, fixed serial port openings should be removed. While this increased software stability, it does not make complete sense from a user's perspective. If there is an error in the port opening or if an object needs to be swapped from the table, DASH needs to be restarted so fresh port openings can be created. This is not ideal to have to restart a program to use a new functionality, especially for an aspect of the system that should remain very flexible (therapists may want to change object types on demand).

In order to fix this, all of the ports should be opened at the start of DASH, and then when a new object is selected, a new data-parsing component (which is the main difference between object types) should be assigned to that port. Currently, the two are linked. By opening a port, the manner in which the incoming data is parsed is also fixed. This does not allow for the swapping of a button with a cone, for example. Again, when completing the first iteration of the system, it was valued more important to find the most stable solution and keeping the object selection at each port fixed for a given session was held as a design constraint.



Tangible Plugin with flexible ports

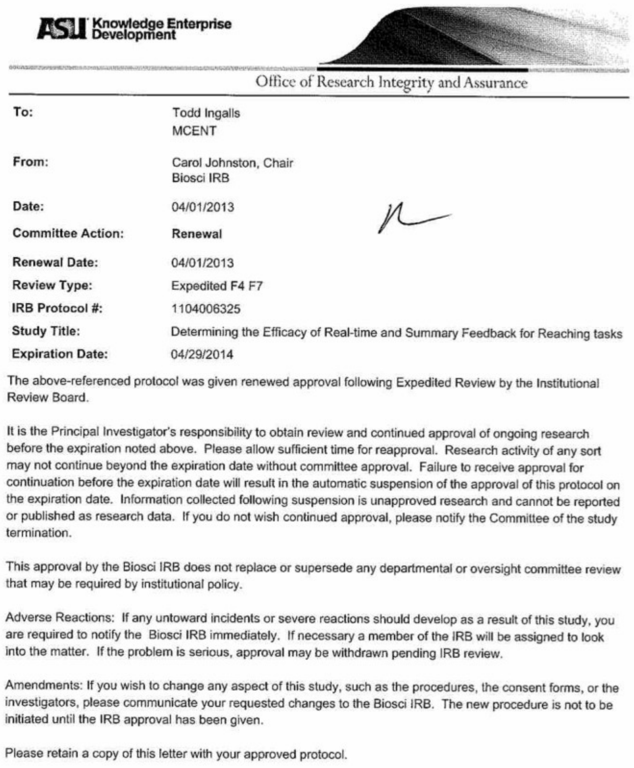
However, moving forward, it may be more desirable to transitions between objects at a particular port during the session. As previously discussed, the fundamental difference between serial ports is how the incoming data is parsed. For the most part, all other port properties are the same, the only difference being that almost all of the tangible objects communicate at a baud rate of 115200, except for the transportable cone (57600) and, though it is handled by the Sensing Plugin, the table buttons (9600). Therefore, given that a port had been opened at a particular address with a configured baud rate, the data parsing is the only remaining difference. Therefore, instead of creating multiple sub-classes of TFSerialInput, a general "TFDataParser" class should be created with specific implementations for each data parse type. Therefore, for example, there would be a TFVirtualDataParser, TFButtonDataParser, TFConeDataParser, TFLiftObjectParser, and TFLiftConeDataParser. Each of these classes would provide a unique method that, given a supplied opened port, could unpack the data accordingly, and update an appropriate data object. This properly encapsulates the opening and closing of the port (which can be a risky operation at times, as previously described), from the parsing of incoming data.

With this procedure, all ports could still be opened once at the beginning of the program, but throughout a session a new data parser could be attached to each port to handle objects differently. The only time a port would need to be closed, is if the lift cone object were to be used in place of the lift cylinder object during a session. However, this situation is unlikely, as the baud rate of this object could always be updated since it was a temporary design.

However, if more objects were being designed with variable baud rates, a method to close and re-open a port with new properties would be required. In this case, there should be a separate class for returning an open port with desired properties. This way the risky operation of opening and closing a port could be separated from other functionalities. This code could even be wrapped in try/catch blocks for further warning and stability.

APPENDIX C

IRB APPROVAL FORMS



ASU Knowledge Enterprise Development

Office of Research Integrity and Assurance

To:	Todd Ingalls MCENT
From:	Carol Johnston, Chair Biosci IRB
Date:	04/01/2013
Committee Action:	**Renewal**
Renewal Date:	04/01/2013
Review Type:	Expedited F4 F7
IRB Protocol #:	1104006325
Study Title:	Determining the Efficacy of Real-time and Summary Feedback for Reaching tasks
Expiration Date:	04/29/2014

The above-referenced protocol was given renewed approval following Expedited Review by the Institutional Review Board.

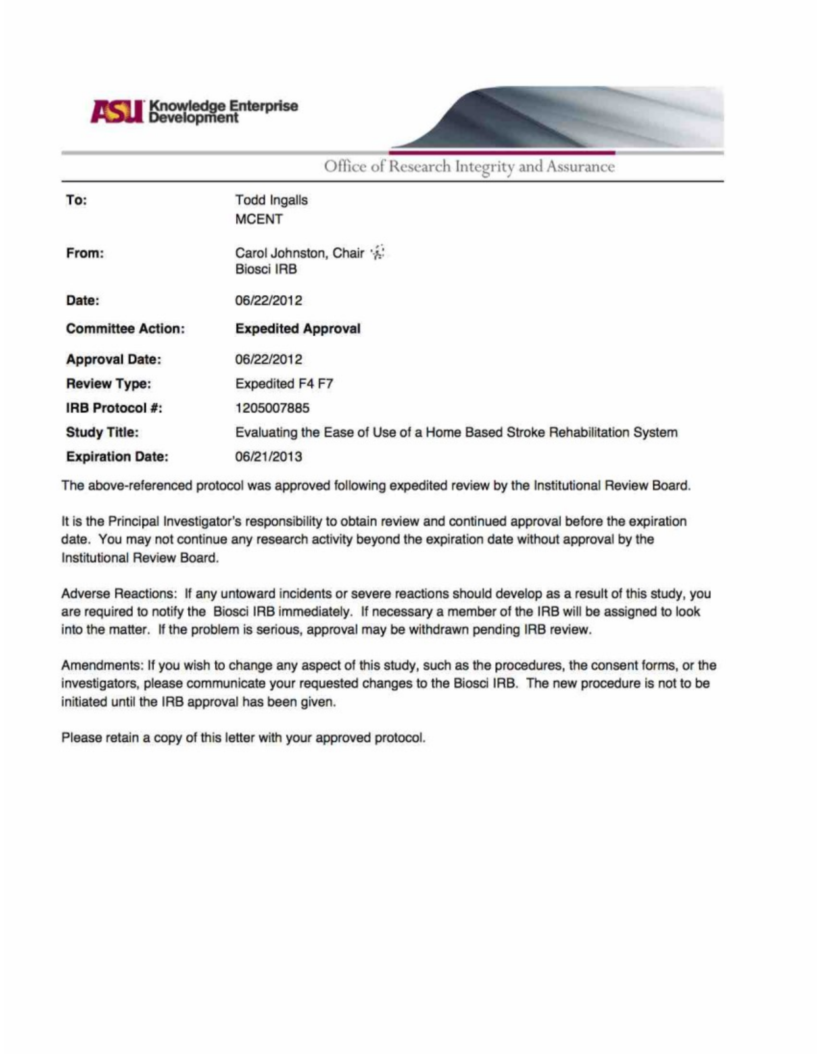
It is the Principal Investigator's responsibility to obtain review and continued approval of ongoing research before the expiration noted above. Please allow sufficient time for reapproval. Research activity of any sort may not continue beyond the expiration date without committee approval. Failure to receive approval for continuation before the expiration date will result in the automatic suspension of the approval of this protocol on the expiration date. Information collected following suspension is unapproved research and cannot be reported or published as research data. If you do not wish continued approval, please notify the Committee of the study termination.

This approval by the Biosci IRB does not replace or supersede any departmental or oversight committee review that may be required by institutional policy.

Adverse Reactions: If any untoward incidents or severe reactions should develop as a result of this study, you are required to notify the Biosci IRB immediately. If necessary a member of the IRB will be assigned to look into the matter. If the problem is serious, approval may be withdrawn pending IRB review.

Amendments: If you wish to change any aspect of this study, such as the procedures, the consent forms, or the investigators, please communicate your requested changes to the Biosci IRB. The new procedure is not to be initiated until the IRB approval has been given.

Please retain a copy of this letter with your approved protocol.



ASU Knowledge Enterprise Development

Office of Research Integrity and Assurance

To:	Todd Ingalls MCENT
From:	Carol Johnston, Chair Biosci IRB
Date:	06/22/2012
Committee Action:	**Expedited Approval**
Approval Date:	06/22/2012
Review Type:	Expedited F4 F7
IRB Protocol #:	1205007885
Study Title:	Evaluating the Ease of Use of a Home Based Stroke Rehabilitation System
Expiration Date:	06/21/2013

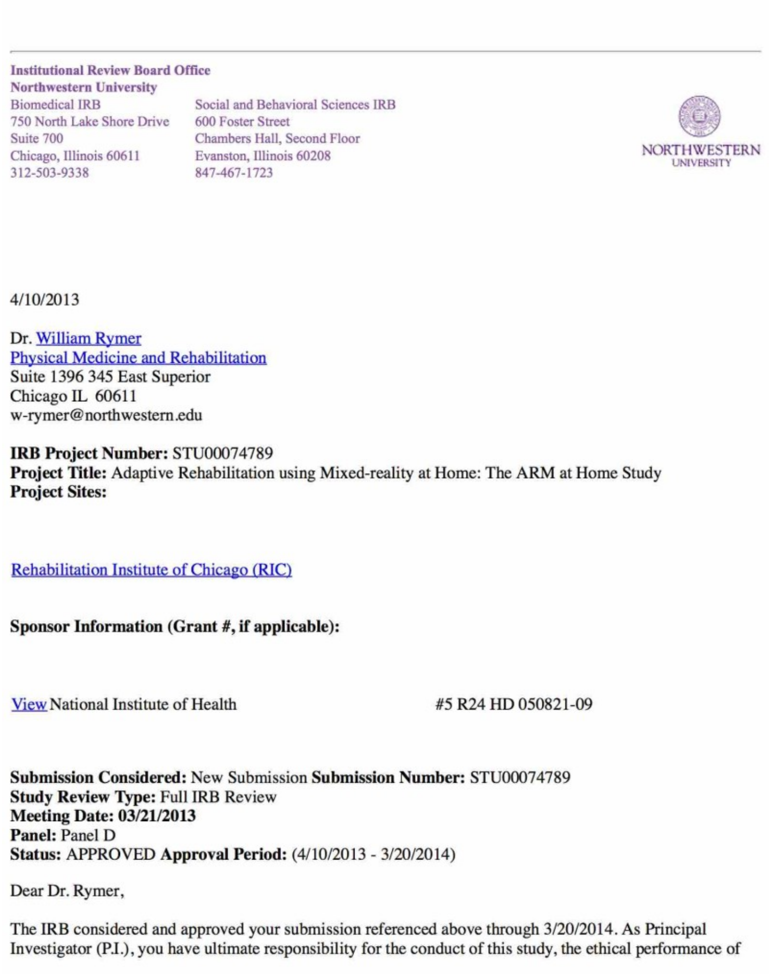
The above-referenced protocol was approved following expedited review by the Institutional Review Board.

It is the Principal Investigator's responsibility to obtain review and continued approval before the expiration date. You may not continue any research activity beyond the expiration date without approval by the Institutional Review Board.

Adverse Reactions: If any untoward incidents or severe reactions should develop as a result of this study, you are required to notify the Biosci IRB immediately. If necessary a member of the IRB will be assigned to look into the matter. If the problem is serious, approval may be withdrawn pending IRB review.

Amendments: If you wish to change any aspect of this study, such as the procedures, the consent forms, or the investigators, please communicate your requested changes to the Biosci IRB. The new procedure is not to be initiated until the IRB approval has been given.

Please retain a copy of this letter with your approved protocol.



Institutional Review Board Office
Northwestern University
Biomedical IRB
750 North Lake Shore Drive
Suite 700
Chicago, Illinois 60611
312-503-9338

Social and Behavioral Sciences IRB
600 Foster Street
Chambers Hall, Second Floor
Evanston, Illinois 60208
847-467-1723

NORTHWESTERN
UNIVERSITY

4/10/2013

Dr. William Rymer
Physical Medicine and Rehabilitation
Suite 1396 345 East Superior
Chicago IL 60611
w-rymer@northwestern.edu

IRB Project Number: STU00074789
Project Title: Adaptive Rehabilitation using Mixed-reality at Home: The ARM at Home Study
Project Sites:

Rehabilitation Institute of Chicago (RIC)

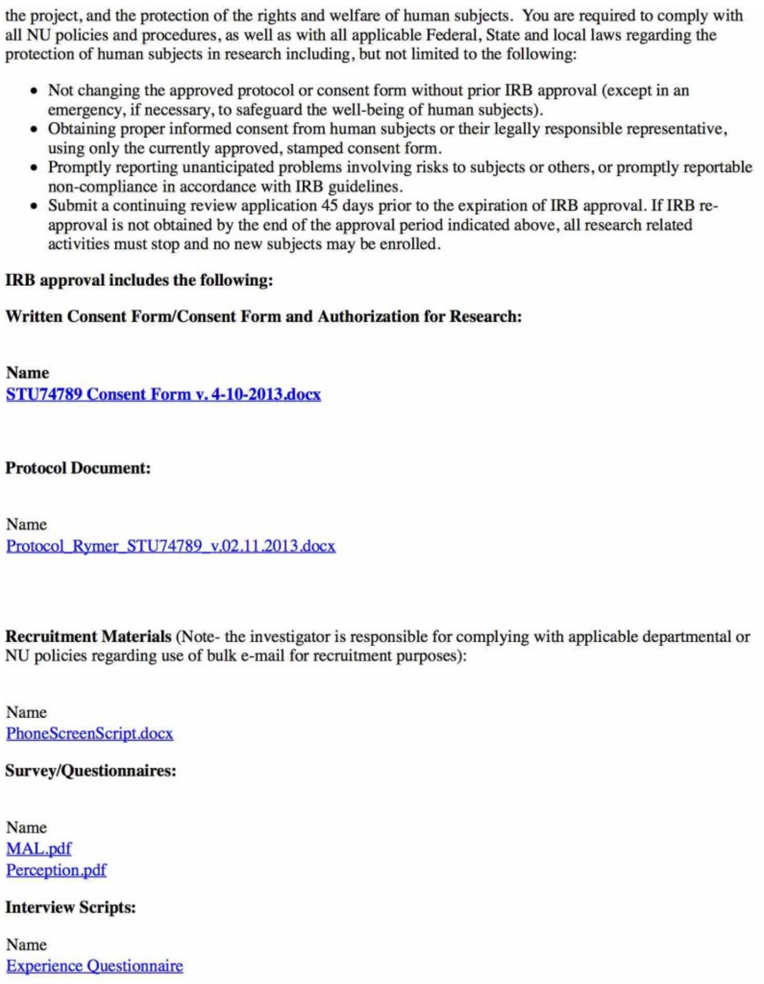
Sponsor Information (Grant #, if applicable):

View	National Institute of Health	#5 R24 HD 050821-09

Submission Considered: New Submission **Submission Number:** STU00074789
Study Review Type: Full IRB Review
Meeting Date: 03/21/2013
Panel: Panel D
Status: APPROVED **Approval Period:** (4/10/2013 - 3/20/2014)

Dear Dr. Rymer,

The IRB considered and approved your submission referenced above through 3/20/2014. As Principal Investigator (P.I.), you have ultimate responsibility for the conduct of this study, the ethical performance of



the project, and the protection of the rights and welfare of human subjects. You are required to comply with all NU policies and procedures, as well as with all applicable Federal, State and local laws regarding the protection of human subjects in research including, but not limited to the following:

- Not changing the approved protocol or consent form without prior IRB approval (except in an emergency, if necessary, to safeguard the well-being of human subjects).
- Obtaining proper informed consent from human subjects or their legally responsible representative, using only the currently approved, stamped consent form.
- Promptly reporting unanticipated problems involving risks to subjects or others, or promptly reportable non-compliance in accordance with IRB guidelines.
- Submit a continuing review application 45 days prior to the expiration of IRB approval. If IRB re-approval is not obtained by the end of the approval period indicated above, all research related activities must stop and no new subjects may be enrolled.

IRB approval includes the following:

Written Consent Form/Consent Form and Authorization for Research:

Name
STU74789 Consent Form v. 4-10-2013.docx

Protocol Document:

Name
Protocol_Rymer_STU74789_v.02.11.2013.docx

Recruitment Materials (Note- the investigator is responsible for complying with applicable departmental or NU policies regarding use of bulk e-mail for recruitment purposes):

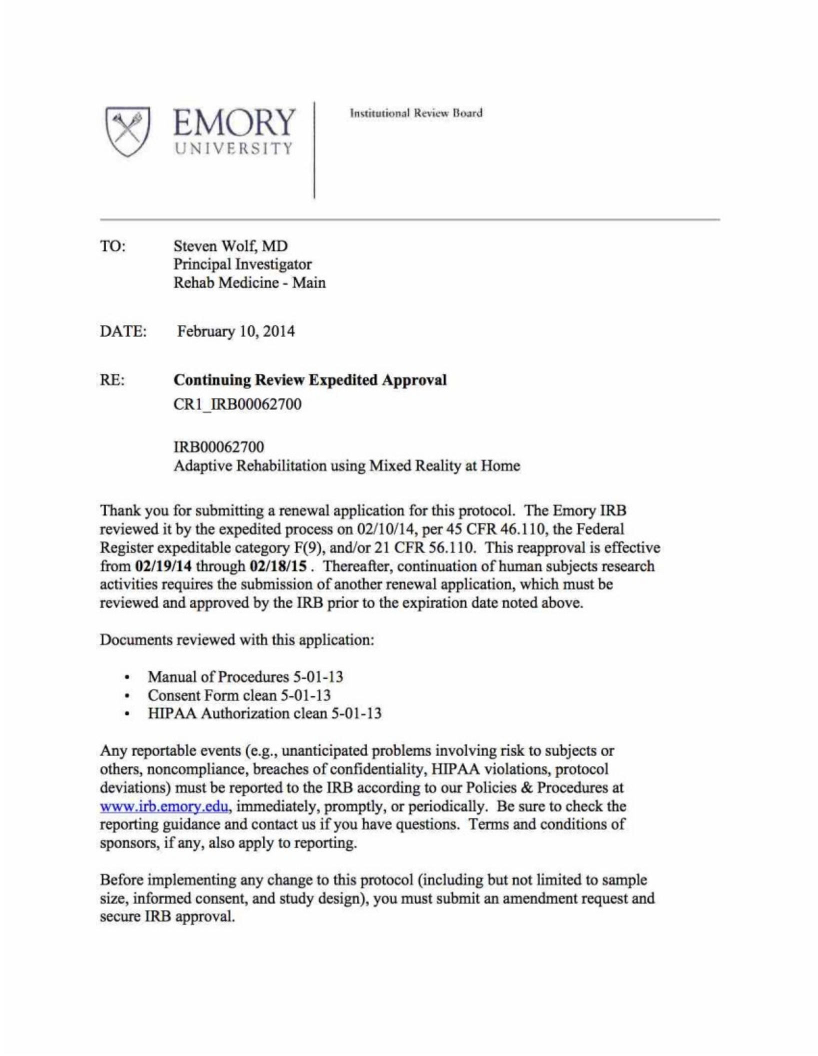
Name
PhoneScreenScript.docx

Survey/Questionnaires:

Name
MAL.pdf
Perception.pdf

Interview Scripts:

Name
Experience Questionnaire



EMORY
UNIVERSITY

Institutional Review Board

TO: Steven Wolf, MD
Principal Investigator
Rehab Medicine - Main

DATE: February 10, 2014

RE: **Continuing Review Expedited Approval**
CR1_IRB00062700

IRB00062700
Adaptive Rehabilitation using Mixed Reality at Home

Thank you for submitting a renewal application for this protocol. The Emory IRB reviewed it by the expedited process on 02/10/14, per 45 CFR 46.110, the Federal Register expeditable category F(9), and/or 21 CFR 56.110. This reapproval is effective from **02/19/14** through **02/18/15** . Thereafter, continuation of human subjects research activities requires the submission of another renewal application, which must be reviewed and approved by the IRB prior to the expiration date noted above.

Documents reviewed with this application:

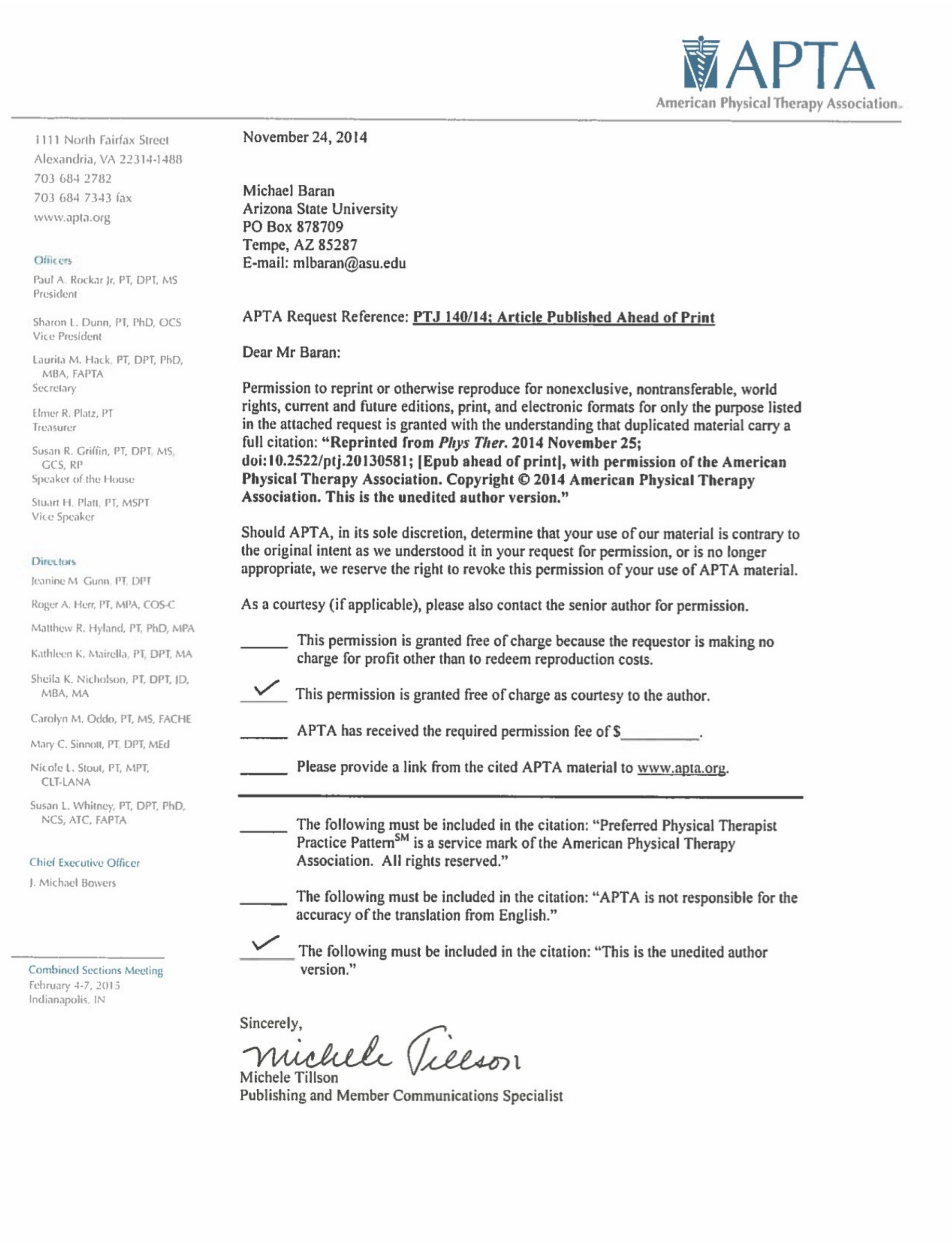
- Manual of Procedures 5-01-13
- Consent Form clean 5-01-13
- HIPAA Authorization clean 5-01-13

Any reportable events (e.g., unanticipated problems involving risk to subjects or others, noncompliance, breaches of confidentiality, HIPAA violations, protocol deviations) must be reported to the IRB according to our Policies & Procedures at www.irb.emory.edu, immediately, promptly, or periodically. Be sure to check the reporting guidance and contact us if you have questions. Terms and conditions of sponsors, if any, also apply to reporting.

Before implementing any change to this protocol (including but not limited to sample size, informed consent, and study design), you must submit an amendment request and secure IRB approval.

APPENDIX D

PHYSICAL THERAPY JOURNAL PERMISSIONS



APTA
American Physical Therapy Association

1111 North Fairfax Street
Alexandria, VA 22314-1488
703 684 2782
703 684 7343 fax
www.apta.org

November 24, 2014

Michael Baran
Arizona State University
PO Box 878709
Tempe, AZ 85287
E-mail: mlbaran@asu.edu

APTA Request Reference: **PTJ 140/14; Article Published Ahead of Print**

Dear Mr Baran:

Permission to reprint or otherwise reproduce for nonexclusive, nontransferable, world rights, current and future editions, print, and electronic formats for only the purpose listed in the attached request is granted with the understanding that duplicated material carry a full citation: **"Reprinted from *Phys Ther.* 2014 November 25; doi:10.2522/ptj.20130581; [Epub ahead of print], with permission of the American Physical Therapy Association. Copyright © 2014 American Physical Therapy Association. This is the unedited author version."**

Should APTA, in its sole discretion, determine that your use of our material is contrary to the original intent as we understood it in your request for permission, or is no longer appropriate, we reserve the right to revoke this permission of your use of APTA material.

As a courtesy (if applicable), please also contact the senior author for permission.

_____ This permission is granted free of charge because the requestor is making no charge for profit other than to redeem reproduction costs.

✓ This permission is granted free of charge as courtesy to the author.

_____ APTA has received the required permission fee of $_________.

_____ Please provide a link from the cited APTA material to www.apta.org.

_____ The following must be included in the citation: "Preferred Physical Therapist Practice Pattern^{SM} is a service mark of the American Physical Therapy Association. All rights reserved."

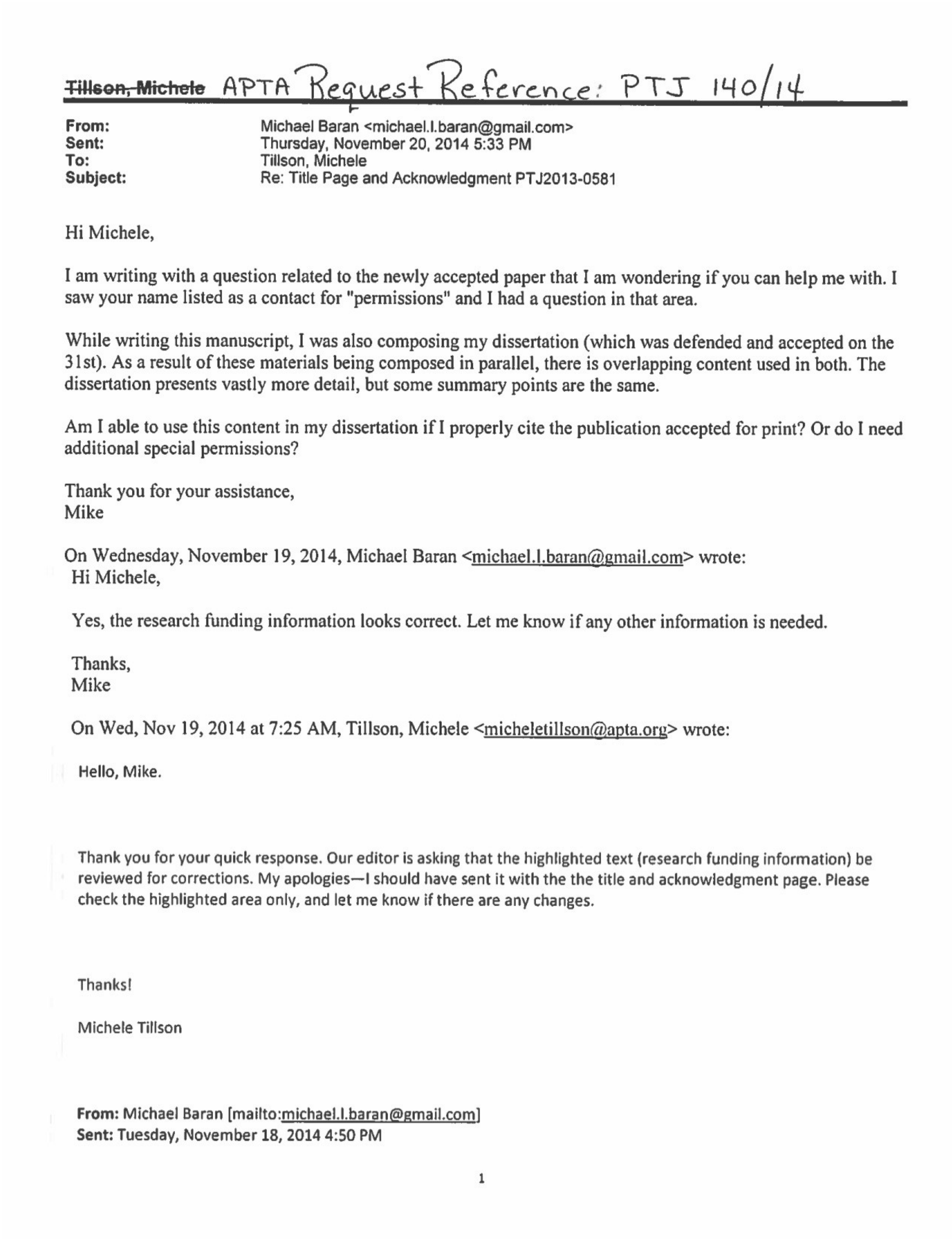
_____ The following must be included in the citation: "APTA is not responsible for the accuracy of the translation from English."

✓ The following must be included in the citation: "This is the unedited author version."

Sincerely,

Michele Tillson

Michele Tillson
Publishing and Member Communications Specialist



~~Tillson, Michele~~ APTA Request Reference: PTJ 140/14

From: Michael Baran <michael.l.baran@gmail.com>
Sent: Thursday, November 20, 2014 5:33 PM
To: Tillson, Michele
Subject: Re: Title Page and Acknowledgment PTJ2013-0581

Hi Michele,

I am writing with a question related to the newly accepted paper that I am wondering if you can help me with. I saw your name listed as a contact for "permissions" and I had a question in that area.

While writing this manuscript, I was also composing my dissertation (which was defended and accepted on the 31st). As a result of these materials being composed in parallel, there is overlapping content used in both. The dissertation presents vastly more detail, but some summary points are the same.

Am I able to use this content in my dissertation if I properly cite the publication accepted for print? Or do I need additional special permissions?

Thank you for your assistance,
Mike

On Wednesday, November 19, 2014, Michael Baran <michael.l.baran@gmail.com> wrote:
Hi Michele,

Yes, the research funding information looks correct. Let me know if any other information is needed.

Thanks,
Mike

On Wed, Nov 19, 2014 at 7:25 AM, Tillson, Michele <micheletillson@apta.org> wrote:

Hello, Mike.

Thank you for your quick response. Our editor is asking that the highlighted text (research funding information) be reviewed for corrections. My apologies—I should have sent it with the the title and acknowledgment page. Please check the highlighted area only, and let me know if there are any changes.

Thanks!

Michele Tillson

From: Michael Baran [mailto:michael.l.baran@gmail.com]
Sent: Tuesday, November 18, 2014 4:50 PM