ISR Institute for Software Research
University of California, Irvine

An Augmented Reality Interface for Game Based Stroke TeleRehabilitation

Arzang Kasiri
University of California, Irvine
akasiri@uci.edu

Walt Scacchi
University of California, Irvine
wscacchi@ics.uci.edu

June 2017
ISR Technical Report # UCI-ISR-17-3

Institute for Software Research
ICS2 221
University of California, Irvine
Irvine, CA 92697-3455
isr.uci.edu

An Augmented Reality Interface for Game Based Stroke TeleRehabilitation

Arzang Kasiri and Walt Scacchi

Institute for Software Research

Donald Bren School of Information & Computer Sciences

and

Neural Repair Laboratory

School of Medicine

University of California, Irvine

June 2017

ISR Technical Report #UCI-ISR-17-3

Overview

We believe game based stroke telerehabilitation (GBSTR) is an effective solution to loss of arm motor control caused by stroke [14]. This report is about applying augmented reality (AR) interfaces to game based stroke telerehabilitation [7, 11]. We believe that by using augmented reality interfaces for our games, we can further improve stroke survivor's recovery rates and engagement with the rehabilitative games [3, 4, 7, 8, 9, 14, 15]. This report is written with a focus on the most recent proof-of-concept prototype we have developed, the Dual Screen prototype.

In this report we will cover necessary background information to understand our work, we will look at other work that has been done in game based stroke telerehabilitation, we will discuss the considerations and decisions for both the hardware choices and game designs, and we will discuss challenges we faced during the development process.

The first section of the report will provide background information to help enrich the reader's understanding of the project that we will discuss. We first define what stroke is and how we engage with it in this project. Then we will define what augmented reality is. We are not going to be using the traditional means of augmented reality interfacing. This will be discussed in the section that defines augmented reality. Next we discuss the past prototypes and game based stroke telerehabilitation systems. These will include the TR Console that is currently undergoing a national clinical trial, the AR1 prototype that illustrated the benefit of decreasing abstraction of the interface, and the Tablet-based AR prototype in which we attempted to apply traditional means of AR to game based stroke telerehabilitation.

The second section will discuss the design of the Dual Screen prototype we made for this project. We will first discuss the choice of hardware devices used and give a description of what they do. Then we will discuss the general considerations we put into designing games for stroke telerehabilitation [4, 14]. After that we will discuss each game we made in depth, providing thoughts on design, a graph of how all the objects in the game interact with each other, descriptions of all the scripts we wrote for the game, and discussion of assets we made for the game.

In the last section we will discuss the challenges we faced throughout the process of this project. These include technical limitations of the devices used and mistakes we made when trying to assemble the prototype we aimed for.

Background

Relevant Information

What is stroke?

Stroke occurs when blood flow to a section of the brain is cut off. This causes damage to the the section of the brain that doesn't get blood. This can result in impaired functionality of the survivor. The most common impairment is the loss of motor control in one's dominant arm [16]. The work we have been doing focuses on rehabilitating survivors in this group.

Working with therapists from the Neural Repair Laboratory in the UCI Medical School, we decided on a core set of features to aim to rehabilitate in our games [14]. The arm can be separated into two sections, the proximal section which is closer to one's body and consists of the shoulder and elbow, and the distal section which is farther from one's body and consists of the forearm, wrist, and fingers. The features then that we aim to bolster are general proximal strength and control, general distal strength and control, gripping strength in both the whole hand and individual finger "pinch" grips, and an added focus on fine motor control in the fingers.

1. Proximal Strength: shoulder, elbow
2. Distal Strength: forearm, wrist, fingers
3. Grip Strength: hand
4. Pinch Grip: fingers
5. Proximal Motor Control: shoulder, elbow
6. Distal Motor Control: forearm, wrist, hand, fingers
7. Fine Motor Control: fingers

These are the areas of focus as specified by therapists from the Neural Repair Laboratory [14].

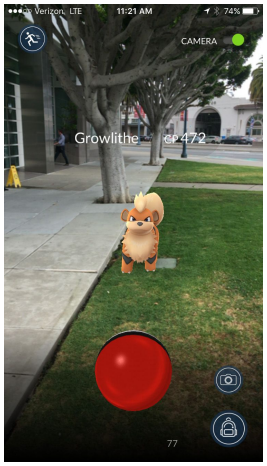
What is augmented reality?

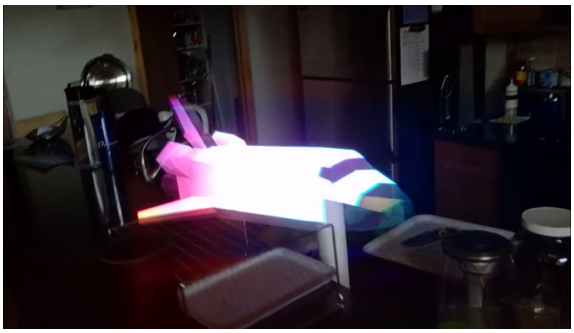
At its most basic level, augmented reality is the mixing of real and virtual objects in perceivable and interactive ways. The most common form of this is a head-mounted display (ex: Google Glass) which let you perceive virtual elements on its screen in addition to the real world through the glasses. Another example use of augmented reality is in the mobile hit game Pokemon Go where you could see a virtual pokemon juxtaposed on a video feed of the real world from your mobile device's camera. In these cases, reality is "augmented" by virtual elements. Our goal with incorporating augmented reality into our stroke telerehabilitation games is to decrease abstraction in the types of interaction, resulting in improved rehabilitation and greater engagement. Interacting with and engaging with a tabletop surface, a form of interaction lower in abstraction, has proven to be better for stroke proximal motor control rehabilitation than the more abstract interaction with a tabletop surface but engaging with a screen in front to the stroke survivor [7].

Additionally, we can use functional objects to augment a stroke survivor's play experience [1]. Functional objects are objects that have a known use and are familiar to the stroke survivor, such as a

kitchen spatula for food preparation or a common tool like a hammer. Having an ingrained understanding built on intuition and plenty of experience is key to the benefit of functional objects [2, 6, 16]. The benefit of functional objects in stroke telerehabilitation is that when doing exercises with functional objects, stroke survivors automatically recall the correct motion and visualization for the task they are using the objects for.

Examples of visual augmented reality:





Prior Research Within UCI

UCI TR Console

The TR Console is a game based stroke telerehabilitation system developed by The Neural Repair Laboratory [14]. It has an arcade like interface consisting of a console composed of buttons, a touchpad, a motion tracking wristband, pinch and grip sensors, a joystick, a dial, and

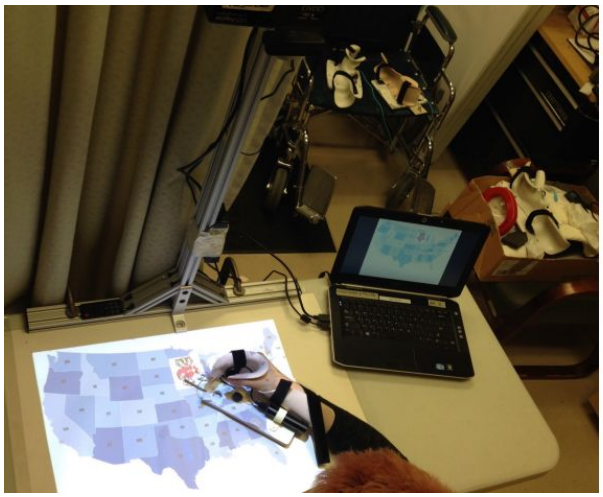
some functional objects: a gun and a hammer. This system was developed with close involvement by stroke rehabilitation therapists. It is currently undergoing a national clinical trial with over 100 enrolled participants.

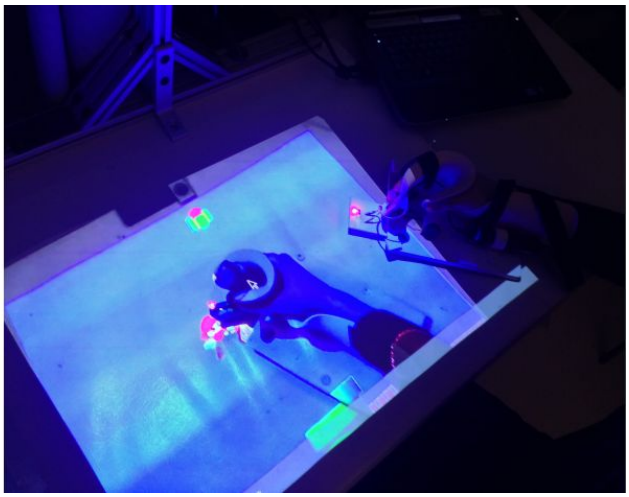
There is an extensive collection of games available on this system all of which use the unique interfacing devices that characterize the system. For example there is a shooting game, a driving game, a game where you aim to generate precise controller input, blackjack, and whack-a-mole to name a few. Additionally, most of these games can work with multiple different devices. Therapists sign in to an online server to regularly assign games to stroke survivors. There they can assign the game and assign specific devices to use on said game.

Where later systems explore different ways of performing or improving upon game based stroke telerehabilitation, the TR console's purpose is to study game based stroke telerehabilitation and see if it has an improved effect on stroke rehabilitation. In preliminary trials, game based stroke telerehabilitation has been shown to have a positive effect on stroke rehabilitation. We will have a more decisive answer when this study is complete.

The TR console does not support augmented reality play. Most of its games are played from a 2nd person, more abstract perspective. In addition the TR console doesn't support social multiuser play.

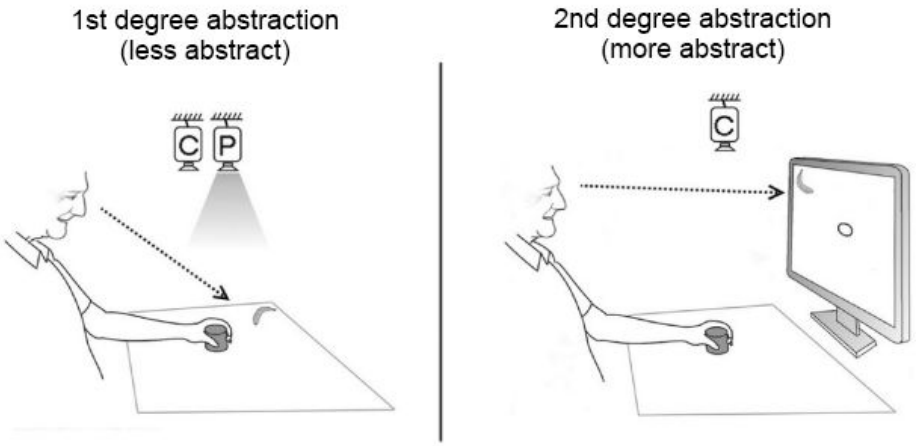
Augmented Reality 1 Prototype





In this first AR prototype system, players have their hand and forearm placed in a brace [7]. They move this brace around a tabletop surface that has a video feed projected onto it. The brace's movements on this 2 dimensional surface are tracked and used for the games. The games in the AR1 all consist of spline tracing. The games require the player to move their arm to follow a guide, all the while drawing out a preset path. This is an activity that is depended on proximal motor control and as such is a good means of cultivating proximal motor control. The path the players end up drawing is known as a spline and is recorded. Comparing the smoothness of a stroke survivor's spline from when they first started using this system (spline is jagged and shaky) to

after they have used it for multiple weeks (spline is smoother) shows how much a stroke survivor's proximal motor control improves by performing these telerehabilitation games [7].

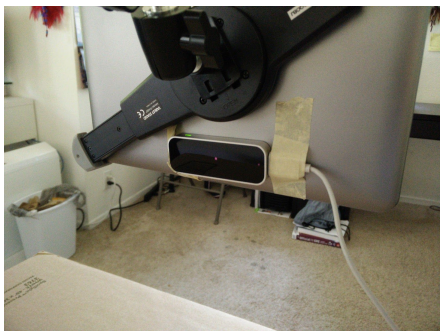


The purpose of this system was to study how different levels of perceiving abstraction effect proximal motor control telerehabilitation. The study compared the system I described above to a version of the system where instead of displaying the game directly onto the tabletop surface the system displayed the game onto a screen located directly in front of the player. The difference is that in the first case the player is looking at what they are doing as they do it, while in the second case the player is looking at only the results of their actions while the are moving their arm around. The first case is an example of 1st degree abstraction while the second case is an example of 2nd degree abstraction. The study found that 1st degree abstraction proved to produce better proximal motor control improvements.

Tablet-based AR Prototype

The AR tablet game based stroke telerehabilitation system was our attempt at making window-based augmented reality, where the stroke survivor looked through a screen (a virtual "window") at a video feed of the real world with virtual objects that they could interact with

integrated into the feed. This is akin to how Pokemon Go used AR (see section “What is Augmented Reality?” above). We chose to incorporate this form of AR as an alternative to using an AR head mounted display. We wanted to avoid using head mounted displays because stroke survivors who have lost motor control in their dominant arm, as most who lose arm motor control do, would have difficulty putting on and taking off a head mounted display on their own and we want our game based stroke telerehabilitation systems to be something stroke survivors can use independently.



The AR tablet system consisted of a tablet connected to the table via an opposable arm with a Leap Motion hand tracking camera attached to the back of the tablet. The tablet was situated such that it was held about 8 in. off the table with the screen facing the sitting stroke survivor. The stroke survivor would reach around behind the tablet where their hands can be tracked by the Leap Motion camera fastened to the back of the tablet (see image to the right). There, the stroke survivor can move their hands about, seeing virtual representations of their hands moving on the tablet screen. Through this tracking, the stroke survivor can interact with virtual objects on the screen in front of them. To demonstrate this system, we built a virtual representation of “Box and Blocks”, a popular grip motor-control rehabilitation exercise and evaluation activity. Pictures of the system and the activity can be found below.

This system was demonstrated to the team at Neural Repair Lab. There it received positive and negative feedback. The team believed that the technology showed promise and that it would benefit from further improvement and refining, but also that presently the hand tracking had problems with accuracy, that the system could be confusing for stroke survivors, and that the arm joint holding the tablet is too complicated for a stroke survivor. Ultimately we decided to shelf pursuit of this design because of the concerns the Neural Repair team raised. The biggest concern was that current tablet arms are too complicated for stroke survivors suffering from hindered arm motor-control to manage on their own.

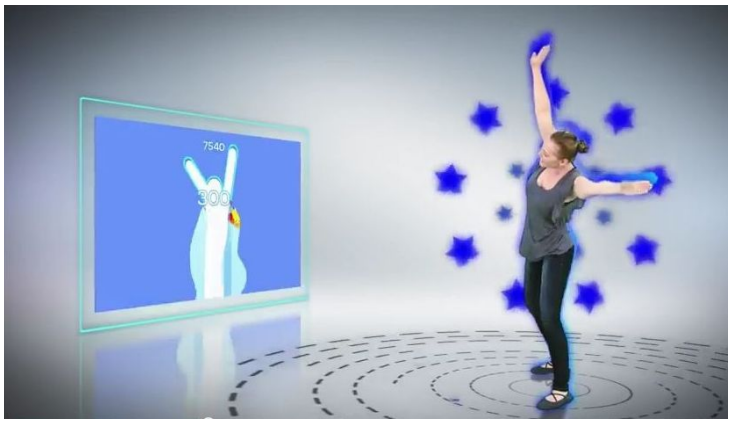
Other Game-Based Rehabilitation Solutions

When planning our work on the Tablet-Based AR prototype, we researched game-based rehabilitation solutions implemented by other research groups. We looked at other groups’ work so that we can get learn about what aspects of game based rehabilitation have already been solved and so that we can get a broader perspective [2, 6, 8, 10, 11, 12, 15]. By looking at other research work we can see varied approaches to a problem that can help spark ideas. From our search we found that game-based rehabilitation is implemented in a variety of different ways. The most common of which being through the

means of large-scale assistive technologies, a variety of glove-based interface device devices, and the use of Microsoft's Kinect body tracking camera. There are also a variety of custom built stroke rehabilitation transducing interface devices that are developed for game based stroke rehabilitation.

Large scale assistive technologies are devices that are large, complicated, and manufactured for a specific type of rehabilitation exercise. They provide support and assistance to survivors while the survivors make the specific rehabilitation movements. This is either in the form of movement support or balance support. Though an effective means of training, the large scale assistive technologies are not a feasible solution to widespread stroke rehabilitation. This is because large scale assistive technologies are complicated and expensive to produce rendering them outside the price range of an average stroke survivor. In addition, because of their bulky form factor, large scale assistive technologies are not an effective solution to telerehabilitation. Not all stroke survivors will have enough space in their homes to accommodate such a large device. Examples: ArmeoPower, ArmeoSpring, ReoAmbulator, and LokoMat.

Glove-based devices are user interface devices with a glove-like form factor that survivors wear and interact with to facilitate stroke rehabilitation via computing device. Because stroke survivors may not have the required level of motor control to put on gloves, glove-based interface devices are not viable for independent stroke telerehabilitation. If provided with assistance putting on and removing gloves, they can be an effective solution to stroke rehabilitation. Examples of glove solutions are Rapael Smart Glove, Music Glove, and Hand Mentor. The Music glove tracks the formation of pinch gestures by the stroke survivor using it. These observed pinch gestures are used in different games provided by the maker of music glove. The Rapael Glove tracks the stroke survivor's hand position and rotation and how much they are gripping. Gloves are good at tracking finger movements and gestures and also hand rotation direction and velocity, but gloves on their own do not provide information on overcompensation in shoulders and torso position. In addition, information on arms must be extrapolated from hand velocity and rotation direction.

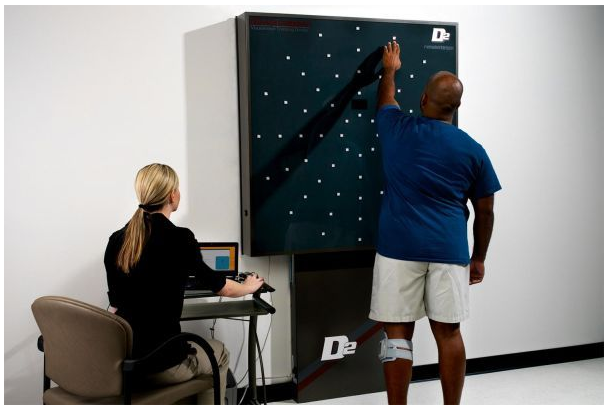


Microsoft Kinect-based stroke rehabilitation games take advantage of the Kinect's body tracking functionality to track gross arm movements in space. Rehabilitation games are based around survivor's body positions and movements. Examples of stroke rehabilitation games that use the Kinect are Dance Wall, VirtualRehab, and Recovr. The kinect on its own does not provide any means of haptic feedback because it is a camera that observes the stroke survivor from a distance. This is disadvantageous because haptic feedback has been shown to have positive benefits to stroke rehabilitation. In addition, because the Kinect is a body tracking

camera, it can only track larger movement, such as arm swings or arm position. It is unable to track details such as hand gestures and finger position. As such, Kinect-based rehabilitation games have not been used for grip and hand dexterity rehabilitation. The Kinect is, in comparison to large scale assistive technologies and other custom interface devices, an inexpensive and convenient off-the-shelf solution for implementing game based stroke rehabilitation. The sale Microsoft Kinect has been discontinued, but there are still other Kinect-like devices for sale. An example of which is the crowdfunded up and coming VicoVR.

So far we have discussed the most common categories, but there are still some rehabilitation systems that fall outside these classifications. There are rehabilitative games that use head mounted displays, a variety of wall based interfacing devices, and other custom-designed transducing devices that are used in rehabilitation games.

Examples of stroke rehabilitation game systems that use head mounted displays include the vHAB project made by students from the University of Washington, Tryomothion's VR rehabilitation system, and the work of the NeuroRehabLab at the University of Madeira. We chose not to include head mounted displays in our augmented reality systems because some stroke survivors may be unable to put on and take off a head mounted display on their own. We want stroke survivors to be able to independently use the telerehabilitation systems we develop to their full capability. This means that head mounted displays are not viable for independent telerehabilitation. This does not mean that head mounted displays are not viable for stroke rehabilitation though. Some projects are attempting to use head mounted displays in a supervised environment to create more engaging experiences in an attempt to improve rehabilitation.



Wall-based interfacing devices include the BTS Nirvana, Dynavision, and the Treax Pads that can be wall mounted or placed on the floor. The last two systems both involve the stroke survivor reaching out and pressing parts of the device while it is propped on a wall. The BTS Nirvana instead has the stroke survivor reach out towards a large screen without actually touching it. All of these systems specialize in training proximal strength and control, without any distal or hand training.

There is yet to exist a perfect solution to game based stroke stroke rehabilitation. All of these systems in addition to our own prototypes have their benefits and disadvantages. The Dual Screen prototype for example lacks the ability to measure and hence train grip strength. Additionally it only provides limited tactile feedback. Alternatively the Dual Screen prototype can support a fairly wide variety of rehabilitation exercises, including proximal control, distal control, and finger control. This is an attribute lacking in several other devices we found. It is common

with custom interface devices and with large scale assistive technologies that the system is only able to train a specific exercise or a small group of similar exercise. Microsoft Kinect-based systems are also limited to proximal control, being unable to track hand and wrist movements. Additionally the Dual Screen prototype is designed to be a telerehabilitation system, meaning that it can be used independently by a stroke survivor with occasional checkups and communication with a therapist. The stroke survivor does not need to be supervised to use the Dual Screen prototype. This is also a feature lacking in several of the systems we found. Systems that require the stroke survivor to wear an item, such as a glove based interface device or a head mounted display, all require supervision or assistance to use.

Playful Social Interaction: Another facet to GBSTR

Though the primary work we did is on augmented reality applications to game based stroke telerehabilitation, we have also researched applying playful social interaction through means of multiuser play and a playful chat client to game based stroke telerehabilitation to bolster engagement and combat depression and decreased self-worth.

Stroke survivors often perceive their ability to participate and interact with others as being decreased. As a result of this, social relationships between stroke survivors and their family, friends, and others in their groups and communities may be perceived by the stroke survivors as diminished. Now feeling uncomfortable in their former relationships, survivors may distance themselves from their social support groups such as their family or community, leading to social isolation. This self-isolation can lead to self-deprecation, loss of identity, and depression, which in turn lead them to slip into depression which in turn makes it harder for the survivors to improve.

To combat this we research playful social interaction as a means of increasing social participation to mitigate and reduce social isolation and feelings of depression. The goal of playful social interaction is to improve the quality of life of stroke survivors and their family and support groups. To implement playful social interaction, we built proof of concept software for a networked multiuser stroke rehabilitation game and a networked chat client for stroke survivors to communicate with each other. Both of these programs are designed to function as add-ons for the TR Console (though they are not actually being used in the clinical trial).

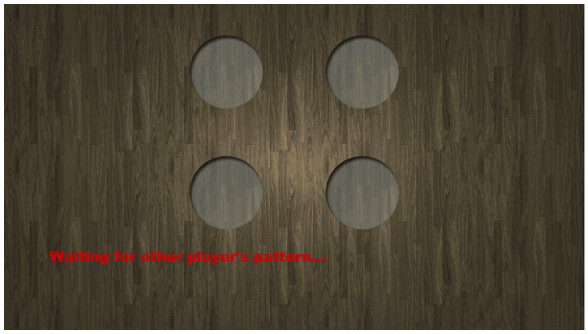
To facilitate the networked interaction for our games we made a simple python server hosted on amazon aws. The server provides group-making and message relay functionality. Game clients request a type of game when connecting to the server. The server then pairs the client to a group of other clients looking for the same type of game. During the game, the server acts as a relay between clients, sharing information to facilitate the multiuser game play.

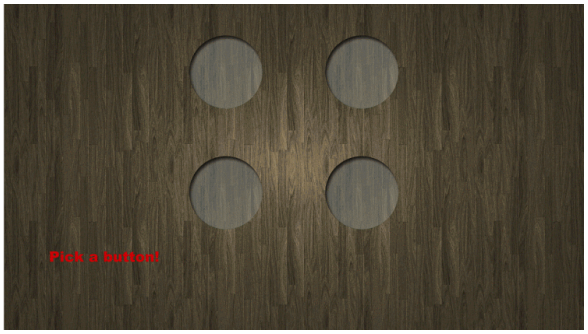
Networked multiuser play with the *Simon* game

The present game based stroke telerehabilitation systems available are limited to single-user play. This does not directly help to mitigate or reduce social isolation and does not encourage social participation. To resolve this, we want to develop games that incorporate online, recreational multi-user play for small groups of 2 to 4 participants. Participants in these games can include stroke survivors, family of the stroke survivors, healthcare providers, and any other members of the stroke survivor’s support groups. These multi-user games can either be local

onsite games where stroke survivors come together in a common space to share a console and play, or the games can be remote networked play where stroke survivors connect with each other and play stroke rehabilitation games online over the Internet.

To demonstrate the concept of a multi-user stroke rehabilitation game, we developed a remote networked play game designed to function with the TR Console. The game we made was a modified version of the *Simon* game where each player takes turns repeating a pattern and adding a new step to the pattern on their turn. The players keep taking turns until someone makes a mistake.



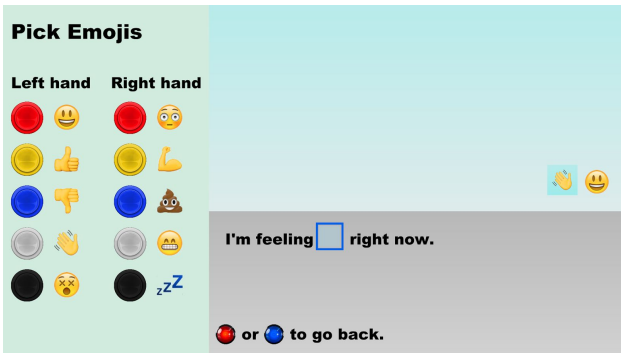


When designing a competitive multi-user game, there is always the concern of balancing user's skill so that a highly skilled user such as a high functioning individual is not paired up with a low skilled user such as a stroke survivor suffering from inhibited motor control. If this pairing were to occur, there wouldn't be a fair competition, rather the higher skilled individual would dominate the lower skilled individual resulting in a boring game for the higher skilled individual and an unpleasant experience for the lower skilled individual. But when the participating players are all at a similar level of skill, the competition can be engaging can cultivate a sense of familiarity and rivalry between the participants, which can help combat feelings of social isolation.

Playful social communication through Emoji Chat

When faced with a loss of hand and arm motor control, stroke survivors may not be able to use traditional human computer interface devices such as the mouse and the keyboard. In addition to this, stroke survivors that have speech aphasia will have trouble communicating through a telephone audio chat. All of these situations hinder stroke survivor's' ability to communicate socially.

To overcome these challenges, we developed the Emoji Chat, a proof-of-concept custom chat client based on a *Madlibs*-style form of sentence formation that uses the stroke-survivor-friendly interface devices of the TR Console [5]. The Emoji Chat enables stroke survivors with limited motor control to utilize simple gestural inputs to select sentence templates and populate them with emojis that best represent their socio-emotive self expression. Using the Emoji

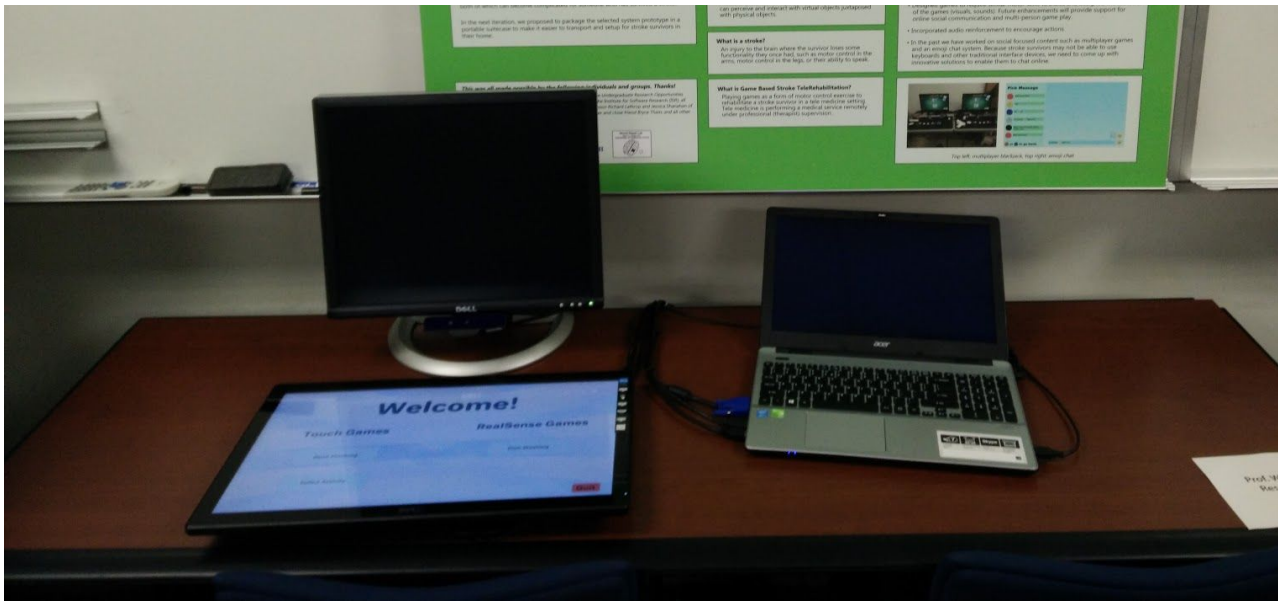


Chat system, the stroke survivor is able to communicate online with others. By doing this, we are able to “level the playing field” between people of different levels of functionality. Both high functioning and low functioning stroke survivors are able to assemble sentences and phrases to communicate. In addition, stroke survivors with speech aphasia and those that speak different languages can communicate using the Emoji Chat.

We chose to use emojis in this chat system instead of only using words because of their playful creativeness, versatility, and potential capability for researching social participation and social isolation of a stroke survivor. An emoji is a small, pictorial representation, ideogram, or emoticon that conveys a complex cultural meaning and socio-emotive expression. As the saying goes, “a picture is worth a thousand words.” Emojis are capable of representing a variety of complex ideas and emotions. Because of this they are a versatile form of communication, being able to communicate many different concepts in different situations. Take for example the thumbs up emoji, it can be used to represent concepts such as “confirmation”, “good”, “approval”, and “up.” On the other hand, this ability to represent a variety of complex ideas can also result in ambiguity. This ambiguity is the core component to the playful creativeness created by the use of emojis. The ambiguity can result in creative, fun, and leisurely ways of expressing oneself. Lastly, a stroke survivor’s choice of emoji may indicate their socio-emotive state or self assessment across multiple communications over time. This is a potential avenue of further research that is available through the Emoji Chat as a result of the use of emojis.

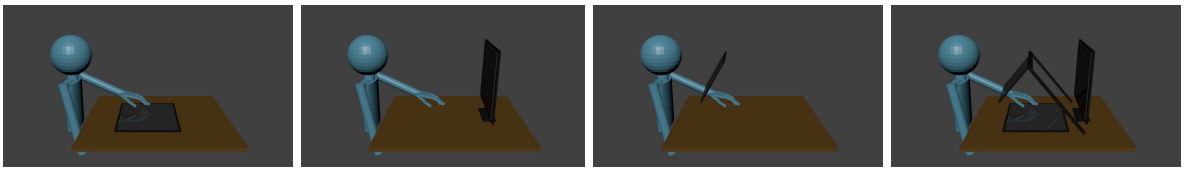
Conceptual Design

Hardware

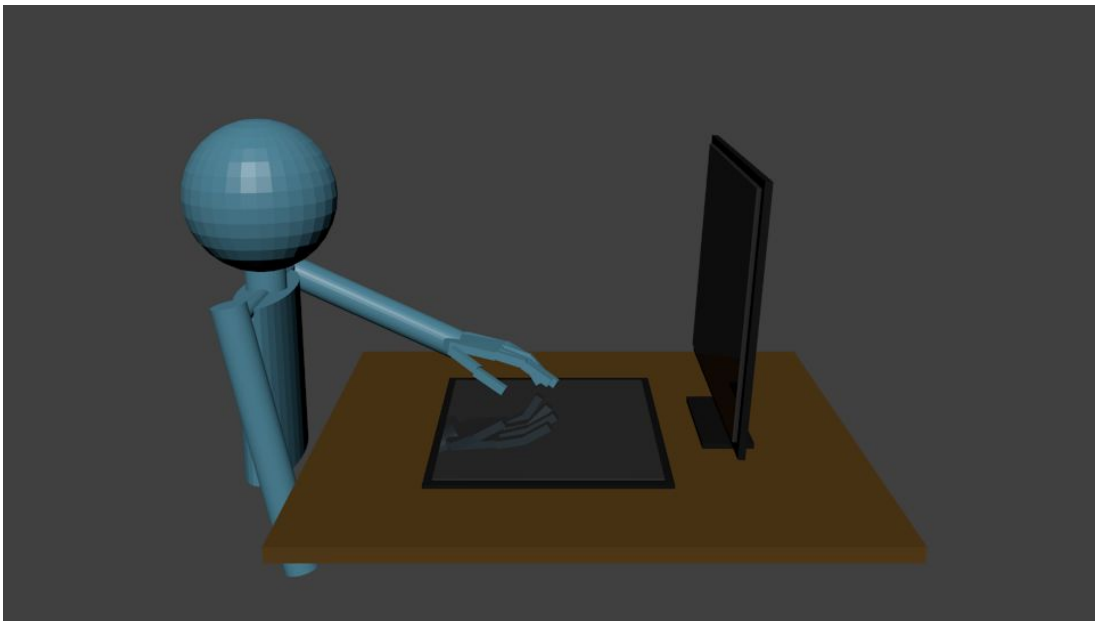


For our Dual Screen prototype, we opted to use two screens, one of which has touch capabilities, and an Intel RealSense camera. Because the two traditional means of augmented reality, head mounted displays and phone / tablets are both too complicated for a stroke survivor who has lost arm motor control to handle, we are required to find more creative solutions for creating an augmented reality experience. In the past we tried to attach a tablet to a flexible arm (see “Tablet-Based AR Prototype” section in “Background” section), but that too turned out to be too complicated. There was another system that was developed in the past that utilize augmented reality via means of an interactive tabletop. As such, what we ended up on deciding to make was a system that harnesses an improved version of this interactive tabletop surface (ie: a touchscreen) and we also chose to include the Intel RealSense depth sensing camera. We had already been researching the use of depth sensing cameras in the Tablet-Based AR project and we believed that even though the Tablet-Based AR prototype isn’t viable for game based stroke telerehabilitation at this time, the depth sensing technology used in the prototype could still be useful. The reason we didn’t add any other devices is because we believe the touchscreen and the RealSense together are already quite versatile. We also wanted to keep costs down. Our goal wasn’t to create a $10,000 system, but rather one that most people could afford. Lastly we believe there is potential for both these devices in stroke rehabilitation. As such, we want to put time into researching the design potentials of both, rather than getting bogged down with lots of peripheral devices.

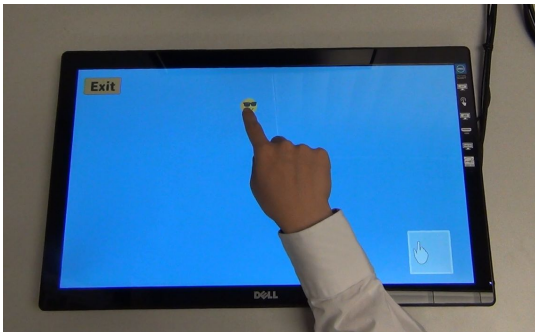
Alternative Hardware designs we considered but enventually ruled out:

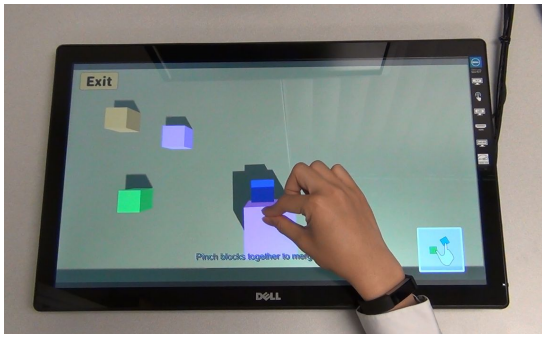


Hardware design we selected to pursue:



Touchscreen





The touchscreen we are using for the Dual Screen prototype is the Dell S2240T 21.5-Inch Touchscreen LED-lit Monitor. Because Unity at the present time does not support touch input in desktop applications, we needed to use the open source TouchScript framework developed by Simonov Valentin to acquire high level interfacing support while working with Unity. TouchScript provided built in gesture recognition that we utilized while developing games that involve

touchscreen interaction. Though it is convenient to use, there is a perceivable drop in touch-input sample rate when using the framework. Down the line we will likely implement our own, more performance focused, framework for touch-input.

Intel RealSense

The intel RealSense is a depth-sensing camera that can be used to track a user's hands and face. The particular version of the RealSense that we used in this project is the Intel RealSense SR300. It sports both a standard RGB color camera and an infrared (IR) sensing camera that registers an IR mesh that is projected by an IR emitter on the RealSense device. By analyzing the displacement of the mesh, the RealSense can form a 3 dimensional mapping of surfaces facing it. Using this map, the hand and face tracking software included in its software development kit (SDK) is able to get an accurate reading of the placement and gesture of a user's hands and placement and expression of a user's face. For our research we only utilize the hand-tracking abilities of the RealSense camera.

Software wise, the RealSense is integrated into our project by means of a collection of Unity resources in a provided framework that is included as part of the RealSense SDK. In this framework you find pre-written scripts for straight-forward integration of the core features of the SDK. For example there is a TrackingAction.cs script that simplifies the process of having an in-game object follow the movements of the user.

Software

In this section we will discuss the games we designed and made for the Dual Screen prototype. It is recommended that you watch the video at the beginning of each game's section to get an idea of what the game looks and plays like. It would be ideal for you to try the games, but without a touchscreen and Intel RealSense, the games would not be possible to play.

When designing augmented reality games for stroke telerehabilitation, we took into account instrumental activities of daily living (IADL), functional objects, and varying degrees of perceiving abstraction [1, 6, 10, 12]. Instrumental activities of daily living are activities such as social participation (shaking someone's hand), meal prep and clean up, and care of pets / others that are as the name implies, instrumental to daily life. We designed games around exercises that are used in these IADLs and we also themed some of the games around these activities. The purpose of this themeing is to convey to stroke survivors that the ability to perform these activities is not out of reach.

Another concept we focused on while designing our games was the inclusion of functional objects. This was discussed earlier in the section defining Augmented Reality, but we will briefly discuss it again here. Functional objects are physical objects that can be held and interacted with. Using a functional object helps stroke survivors regain their motor control quicker than without.

By performing familiar gestures with familiar objects, stroke survivors can recall what it feels like to perform a familiar gesture, helping guide their rehabilitative exercises. As an example, holding and flipping a physical spatula recalls memories and thoughts of the flipping gesture that was once performed by the stroke survivor, helping give them a clear understanding of the motion they are now trying to do.

Lastly, we kept in mind 1st, 2nd, and 3rd person play when designing our augmented reality telerehabilitation games. We specifically selected our devices and designed our prototype to promote 1st person and 2nd person play, so we need to make sure that we design games that meet that criteria. That is to say, we have not and will not make a game where the stroke survivor will be required to interact with the bottom screen while watching the upper screen.

Games

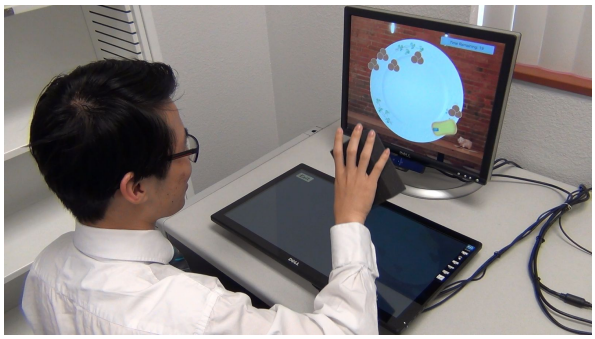
So far we have developed 3 games that utilize the Intel RealSense and 2 games that utilize the touchscreen. One of the RealSense games, the Sandwich Making game, turned out to be not viable at the present time to include in the prototype. This is discussed further in the Challenges Faced subsection of the Assessment section of this report.

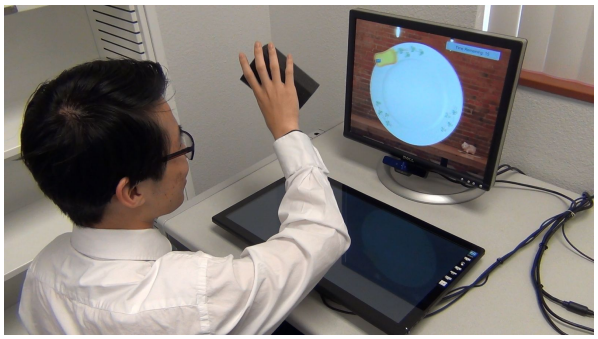
Presently, we do not have a game that utilizes both the Intel RealSense and the touchscreen together. That isn't to say that we don't plan on having such a game in the future. We are currently exploring different game designs that would be able to combine both the devices in one game.

The Dual Screen prototype initially loads into a menu screen that acts as the central hub of all the games. When a game is done, the stroke survivor can return to this menu. From the menu the stroke survivor can access any of the available games, or exit the application. For this purpose, all the games share two core scripts that serve administrative purposes in the games and facilitate navigation between the games and the menu. To avoid redundancy, these are not listed in each game's section in this report. Instead they will be listed right here:

- UseMultipleDisplays.cs
    - Checks to see if all displays are active. Unity needs to activate displays to be able to use them in the games. If the displays are not already active, they are activated.
- ExitToMainMenu.cs
    - Holds a GoToMainMenu() method that is called when the "Exit" button is pressed.

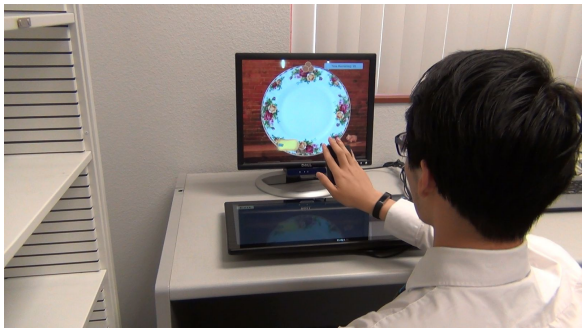
DISH WASHING GAME:





Introduction:

The dishwasher game is a game where the stroke survivor uses proximal (upper-arm, shoulder) motor control and hand grip strength to grip and hold a sponge and to move said sponge, guiding a virtual sponge displayed on a screen in front of the them to clean a virtual plate also displayed on that screen. The stroke survivor moves their physical sponge left and right and up and down, guiding the movement of the virtual, 2D sponge. The virtual sponge mirrors the position of the held, physical sponge. As the virtual sponge travels across the screen, it cleans the virtual plate under it. Whenever the sponge comes in contact with a dirt spot on the plate, the dirt is purged. Once the plate is free of all its dirt, it will scroll to the left, to be followed by a new dirty plate scrolling from the right. At the end of the game when the timer runs out, the number of dishes that have been cleaned is displayed on screen for the player to see. This number could be broadcast to a therapist if needed.



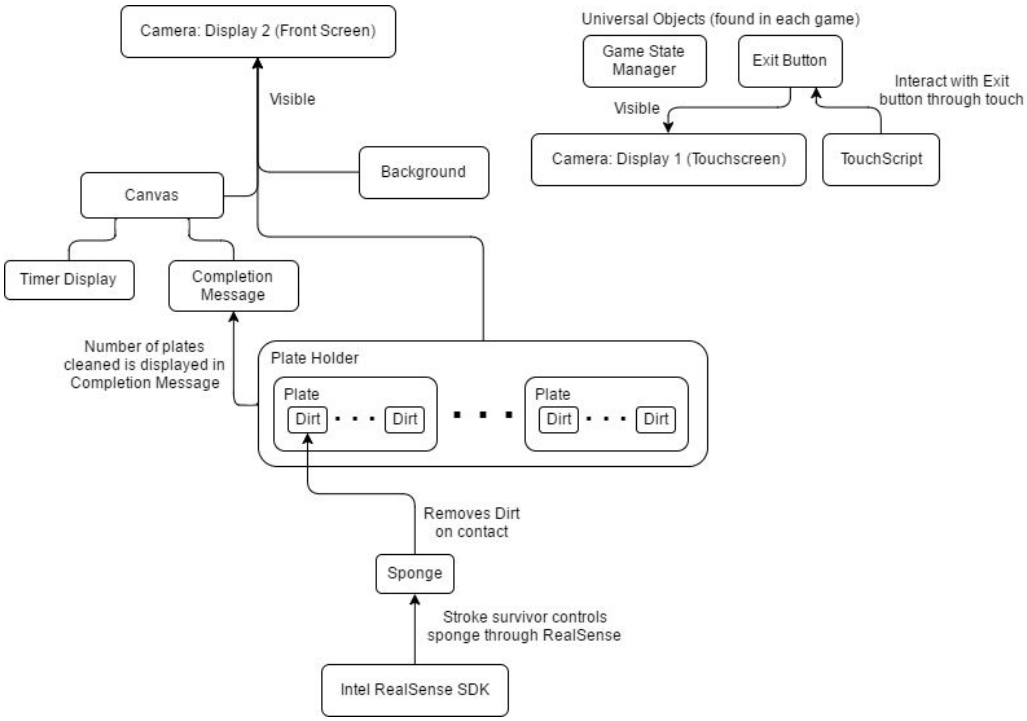
Level Design:

For the purposes of demonstrating the Dual Screen prototype's concept and design and the Dish Washing game's core mechanics, we made a timer based level in keeping with the current style of games present on the TR Console. In this level, the stroke survivor cleans a never-ending series of plates with randomly generated amounts of dirt on them until a timer runs out. This is not a hard requirement though. We can instead make pre-programmed sequences of plates with manually defined quantities and placements of dirt. This way, we can create a series of unique levels. We can also optionally add a timer to these levels where the stroke survivor will need to complete the level within a time limit to be considered successful. In addition, we could add variety to the game by include moving dirt. This would require more precise motor control and a degree of planning and prediction. The stroke survivor would need to be able to predict where a piece of dirt will be and then plan how to get there in time to come into contact with it. We could also include an opposite of dirt, something that the stroke survivor will be required to not come into contact with to succeed. Finally, we could having plates of different sizes. This wouldn't add any new challenge to the game, but seeing something new and different helps fight off feeling of "sameness" and redundancy.

Technical:

We use the TrackingAction.cs script provided in the RealSense SDK to facilitate the virtual sponge following the real sponge. TrackingAction.cs is used for tracking hands, but it can, reasonably accurately, be used to track an object being held by a hand. In the future, we will implement this tracking ourselves for greater accuracy, but for a proof of concept demonstration, TrackingAction.cs is fine.

Relational Graph between Game Objects:



Scripts we wrote for this game:

- Timer.cs
    - Is attached to the GameStateManager game object. Takes in a number of seconds and counts down till 0 seconds remain. When the time has run out, Timer.cs ends the game and triggers the completion display showing the number of plates washed.
- PlateManager.cs
    - Creates and manages the plates that need to be washed. Is attached to a holder object containing all the active plates. Keeps track of a current and next plate, scrolling both of them when the current plate is clean. Once the current plate is cleaned and has left the screen, it is removed and a new plate is added. Also keeps track of the number of plates that have been washed.
- DirtManager.cs

- Is attached to the plate game objects. Creates and manages the dirt objects attached to the plate. This script spawns all the dirt objects when the plate is created and then keeps a counter of how much dirt remains on the plate.

- Cleaner.cs
  - Is attached to the virtual sponge object that is controlled by the stroke survivor. On collision with a dirt object, the dirt object's GetCleaned() method is called, telling it to be cleaned.

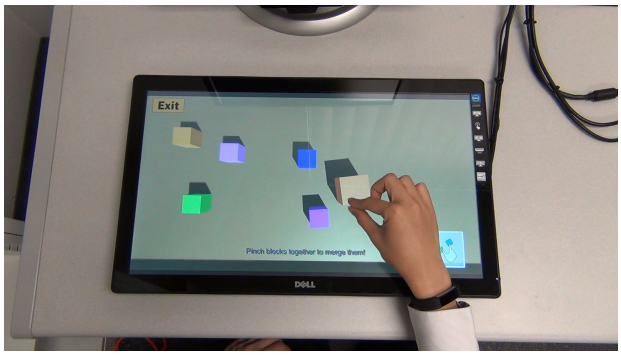
- Dirt.cs
  - Is attached to a dirt object that is attached to a plate. Has a GetCleaned() method that decrements the counter in the parent plate's DirtManager script and also removes the dirt object.

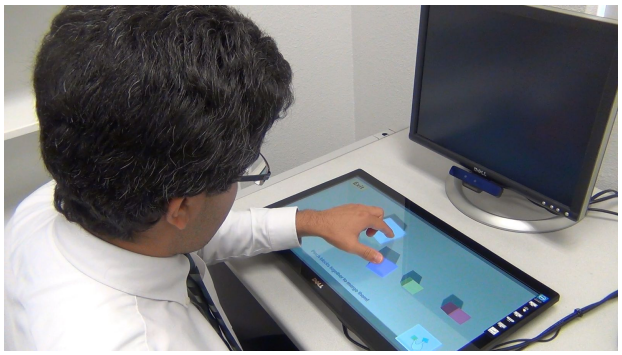
Assets:

The only visual asset we created for this game is the dirt sprite. The background image, plates, and sponge sprite were all found online through google image search. All of which, aside from the background image required minor cleaning and reformatting from jpg to png (for transparency).

There are no audio assets used in this game. In the future, it would be beneficial to add in a sound effect for sponge-dirt contact, a sound effect for successfully cleaning a plate, and a song to play in the background.

BLOCK COMBINING GAME:





Introduction:

The Block Combining game is a game where the stroke survivor uses finger motor control to pinch blocks together. This game utilizes the Dual Screen Prototype's tabletop touchscreen. To play the game, stroke survivors touch and drag one block with their index finger and then, while touching the first block with their index finger, they touch another block with their thumb. They then drag these two blocks together, in the process making a pinching gesture with their index finger and thumb. When the stroke survivor's index finger and thumb are touching, the two blocks will merge together, forming a larger block with a color somewhere in between the colors of the two parent blocks. The game ends when all available blocks have been combined into a single, large block.

This game relies on an element of trust between the therapists and the stroke survivors playing the game. We cannot enforce that the index finger and the thumb are what is touching the blocks, for all we know, the stroke survivor can be using two separate hands to bring the blocks together. This will require the therapists to convey the importance of performing the correct gesture to the stroke survivor who will play this game and it will require the stroke survivor to be responsible when they play. If, for example, the stroke survivor were to play the game with two hands instead of an index finger and a thumb, they would miss out on rehabilitating the pinching gesture this game works to develop.

Level Design:

For the purposes of demonstrating the Dual Screen prototype's concept and the Block Combining game's core mechanics [4, 14], we made a simple level where the stroke survivor combines 7 different blocks into a single large block. Going forward, we can perform an assortment of changes and additions to this game to create a variety of different types of levels to keep stroke survivors interested and engaged.

One different level design would be to add in an element of cognitive challenge, where each block holds some property and when blocks are combined the new resultant block's property would be derived from its two parent blocks. The goal then would be to combine specific blocks in a specific order to produce a desired resultant block. For example, each block could have a number displayed on it. When the blocks are combined, the numbers are added together to form the new block's number. The goal then, would be to create a block in the desired number. As

another example, we can have the resultant block be the average size of the parent blocks. In this case, the goal could be to produce a block of a specific size.

Alternatively we can put a time limit on the level. In that case, the stroke survivor would have to complete the level's task in the allowed time to succeed. This would serve to increase the challenge of whatever task they are performing, be it just a motor control exercise, or motor control plus cognitive challenge.

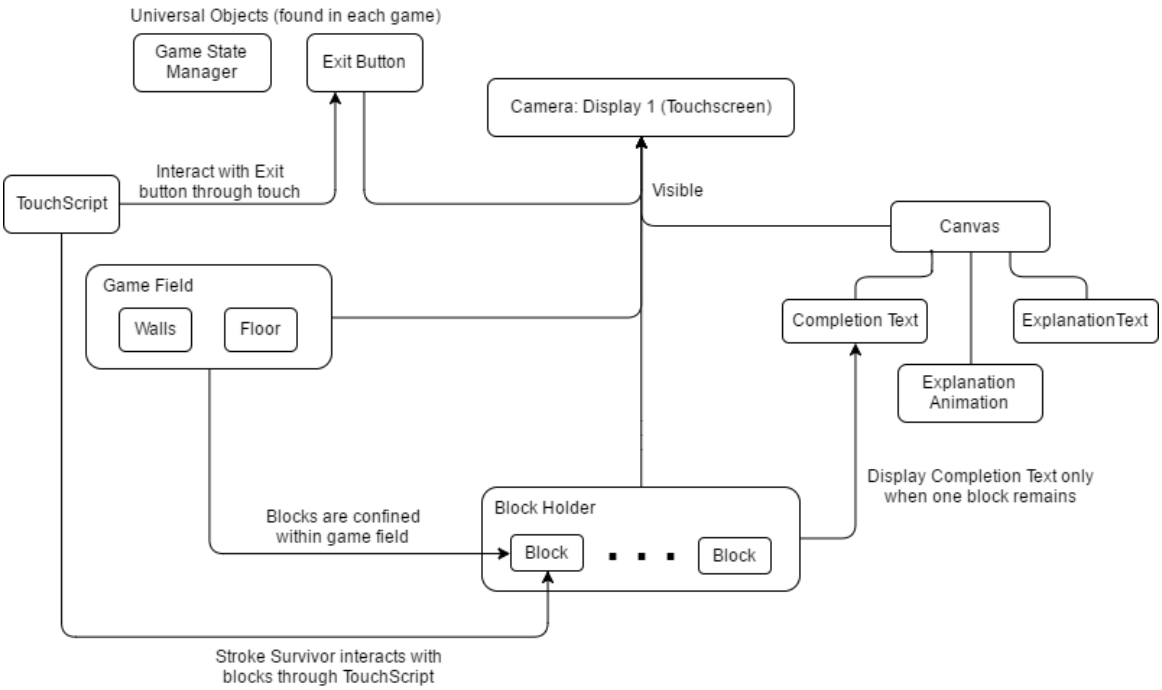
Another, more innocuous option would be to add in physical obstacles to the level space. In the current demonstrative level, the combinable blocks all exist in a open field bounded only by walls surrounding the border of the screen. To add variety, we can add walls into the middle of the field, creating an obstacle the stroke survivor would need to trag combinable blocks around. Admittedly this is nothing more than a minor cognitive challenge, but it would add a degree of "newness" to the level, keeping the stroke survivor from getting caught in a feeling of staleness.

We presently have auditory enrichment in the Block Combining game. Different, satisfying sound effects are played when blocks are touched and when blocks are combined.

Technical:

As mentioned in the touchscreen subsection of the devices section, we used the TouchScript framework for Unity as a means of interfacing with the touchscreen. TouchScript provides a variety of supported gestures as part of its framework. For the Block Combining game, we used the provided Transform Gesture to move blocks around and the Press Gesture to play the on-touch sound effect. As with all the other games, we used a UI gesture to register a tap the the exit button displayed in the top left corner of the touchscreen.

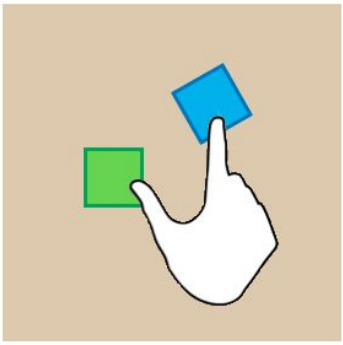
Relational Graph between Game Objects:



Scripts we wrote for this game:

- Combinable.cs
    - Attaches itself as TransformGesture event handler. When a collision between the block this script is attached to and another block occurs, it checks if both the blocks are being touched and if the touches are close enough to warrant a combine. If they are, it spawns a new block, plays the combination sound effect, and removes the parent blocks.
- Draggable.cs
    - Attaches itself as a TransformGesture event handler. When a transform gesture occurs to the game object this script is attached to (ie: a stroke survivor tries to drag a block), this script applies the same transform position from the gesture to the game object it is attached to, resulting in the object following the touch input (drag by touch).
- CheckAllBlocksJoined.cs
    - Is intended to be attached to a game object that acts as a holder for all the blocks in the scene. The script checks every frame if the number of child objects (blocks) is 1. If it is, then the game is considered complete and the completion text is displayed.
- SFXOnPress.cs
    - Attaches itself as a PressGesture event handler, and when a PressGesture is triggered, it plays the sound effect from the AudioSource connected to this instance of SFXOnPress

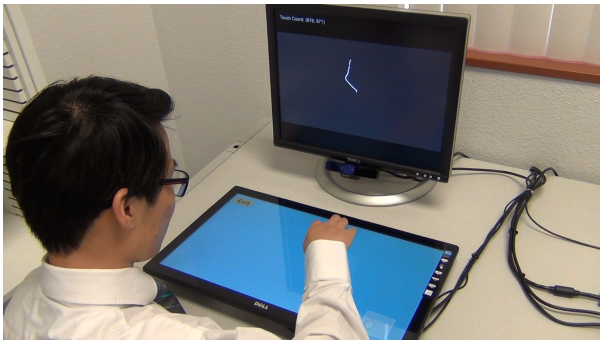
Assets:

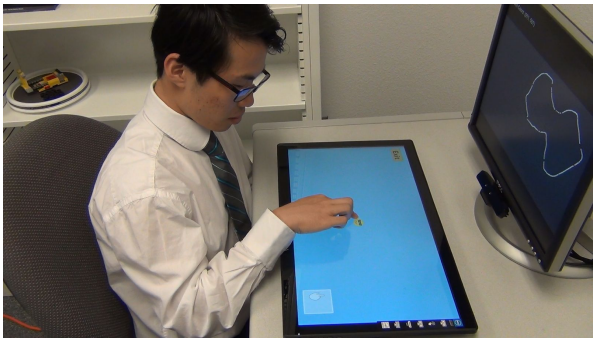


The only visual asset we made for the Block Combining game is a tutorial animation showing the core action that the stroke survivor needs to perform. This is included to the right (animated gif). The rest of the visual assets seen in this game are default Unity assets (colored blocks, font).

Audio assets consisted of two sound effects found on freesound.org. There is a plop sound for then a block is tapped and a bloop sound for when blocks are combined.

SPLINE ACTIVITY GAME:

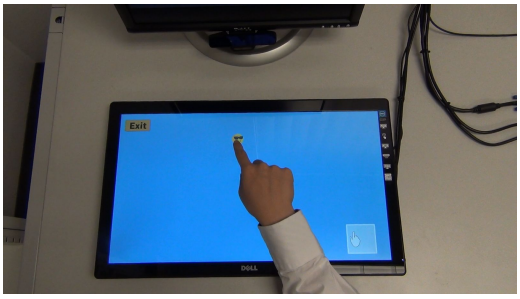




Introduction:

The Spline Activity game is demonstrative game with the purpose of asserting that this Dual Screen prototype can achieve the capabilities and hence is at least as good as the AR1 prototype. As mentioned in the AR1 section of earlier projects, AR1 is an augmented reality game based stroke telerehabilitation system that is build for spline tracing. All the games for the AR1 involve drawing splines. As such, this Dual Screen prototype, which is shown to also be able to trace splines, is at least as good as the AR1 prototype.

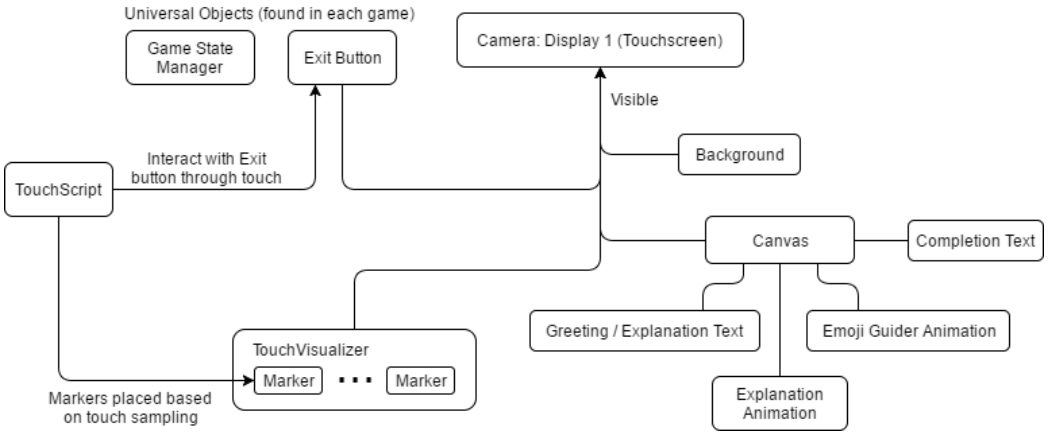
Though the current Spline Activity game prioritizes efficiency to convey its capabilities, this isn't to say that we can't make a more interesting, "gamey" spline game. Now that we have shown that the Dual Screen system has the ability to trace splines, we can make spline activities with more of a narrative to them, or spline activities that have interesting visual elements worked in to keep the stroke survivors interested. In fact, down the line we can port over the games from the AR1 system, saving time designing the games and making the assets.



The Spline Activity game exercises proximal (upper arm, shoulder) motor control for the stroke survivors when they are moving their hand to follow the moving marker.

Technical:

Relational Graph between Game Objects:

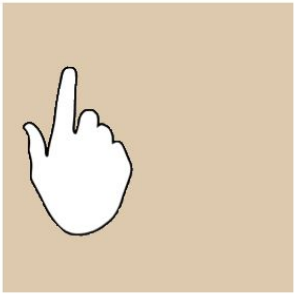


Scripts:

- Modified MyTouchVisualizer.cs from TouchScript framework:
    - Modified so that instead of creating preset marker objects as its own children, it now makes copies of a given marker object and the objects become children of a given holder object. These changes allow greater graphical control of how the touch visualizations look.
- DrawOnTexture.cs:
    - This wasn't used in the Spline Activity game, but it was written while we were developing the game. This script will be useful for making a tracing game in the future. Or it could also be used for a drawing activity.
    - Is intended to be placed on a plane object and be given a template Texture2D object to place on the plane.
    - Attaches itself as a TransformGesture event handler. When a TransformGesture occurs to the the object this is attached to, the pixels at the centerpoint of the gesture change color. The point of contact can also be tracked at this point. This can be used for general drawing and "color in the lines" drawing, which requires more precise motor control.

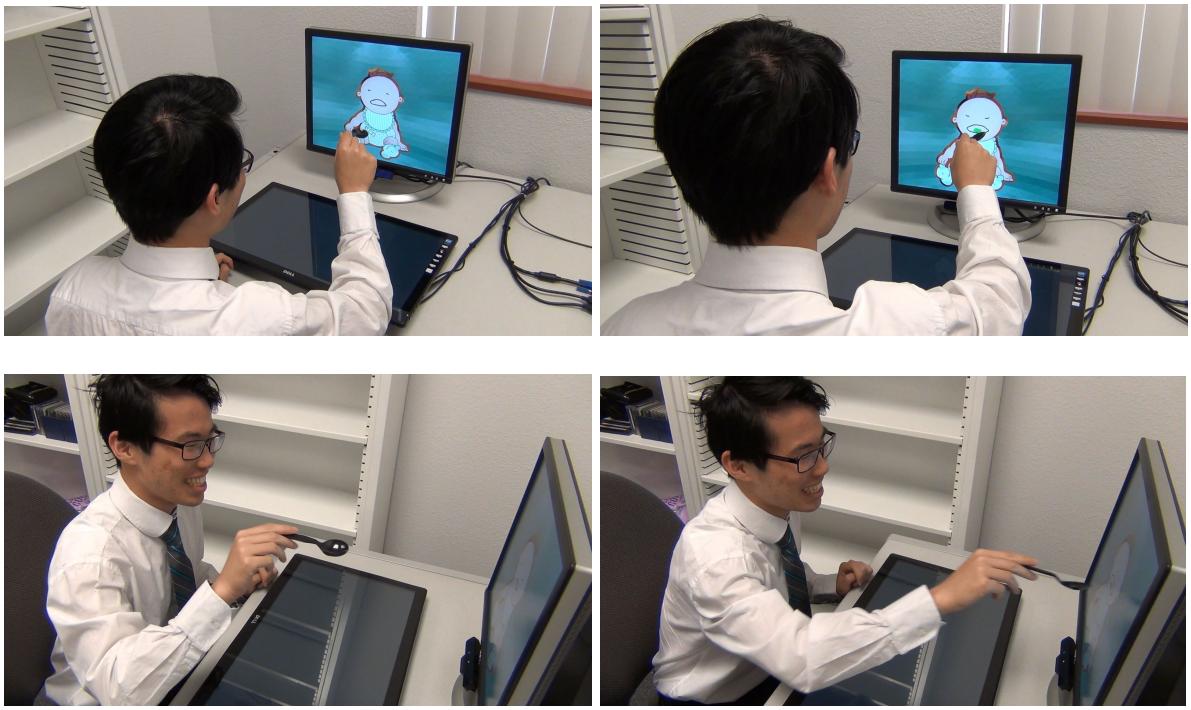
Assets:

This game has no audio assets and only has a single emoji sprite (seen to the right) and a tutorial animation. The emoji sprite needed to be cropped and cleaned up before being used in this game. The Spline Activity game is on the minimal side of



things, as far as assets are concerned. The spline path in this game is actually implemented as an animation, so technically it could qualify as an asset, but we would consider this animation more as a mechanic, a technical element, rather than an aesthetical element.

BABY FEEDING GAME:



Introduction:

The Baby Feeding game is the third game we made that utilizes the Intel RealSense. In this game, the stroke survivor uses proximal (upper arm, shoulder) motor control and finger grip strength to hold a spoon and extend the spoon forward and backwards to feed a virtual baby displayed on the front screen.

At random time intervals, the normally relaxed baby begins to feel hungry and cry. Depending on how close the stroke survivor's outstretched hand holding a spoon is, the baby will either stop crying and open its mouth slightly, open its mouth, open its mouth wide, and when the spoon bearing hand is extended fully, the baby receives a virtual parcel of peas and begins to chew, returning to its relaxed demeanor afterwards.

This game can inspire memories or thoughts of caring for a child, a form of responsibility that some look back fondly upon. For stroke survivors who are feeling depressed due to a decreased self-worth, giving them the chance to perform this responsibility, even in the confines of a virtual game, can be rewarding and rejuvenating.

Level Design:

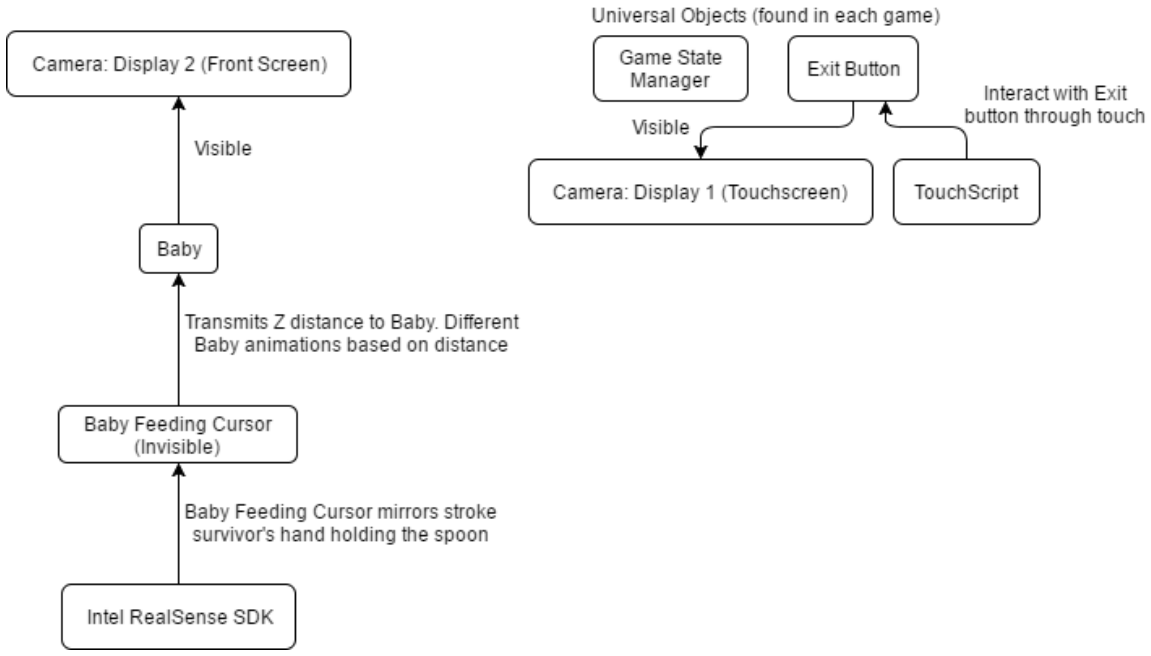
There are multiple different additions or changes we could make to this game to create a variety of different levels, each with a unique challenge or engaging element. Some features are: managing multiple babies, intensity of crying increases over time, having to do gestures such as

shaking the spoon to humor a baby before feeding it, adding in a food jar that the stroke survivor would need to “refill” at, and adding different types of baby food for feeding different types of babies. Most of these fall in the category of adding cognitive challenge to the game, while a few require added motor control. They all add an element of variety and “newness”, ensuring that the chances of a stroke survivor getting tired and bored of the Baby Feeding game will not come soon.

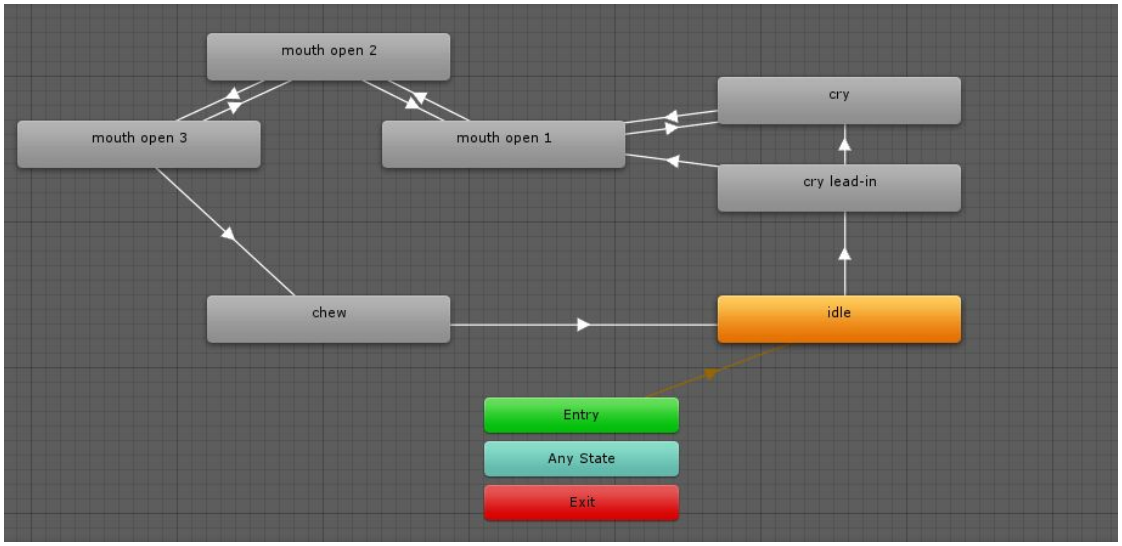
Technical:

Like with the DishWashing game, we used TrackingAction.cs to track the stroke survivor’s extended hand that is holding the spoon. In the future it will be more ideal to program our own means of tracking for functional objects in which we can track the objects specifically, or at least accepts a general blob, rather than a tracking function that specifically is looking for hands.

Relational Graph between Game Objects:



Animation Controller Graph:



Scripts we wrote for this game:

- BabyStateManager.cs
    - Is intended to be attached to a baby game object. Randomly starts the hungry crying animation when the baby is idling. Also acts as an intermediary between the AnimationController and the BabyFeedingCursor, whose distance from the baby define which animation should be playing.
- BabyFeedingCursor.cs
    - Intended to be attached to an object that also has the TrackingAction.cs script attached to it. Because of the TrackingAction script, the object BabyFeedingCursor is attached to will move around, entering and leaving the bounds of a baby. With the current demo, there is only one baby so the cursor reads itself as always being within the bounds of that baby. When it is in the baby's bounds, it transmits its Z position value (forward-backward, to the stroke survivor) to the baby. The baby uses this position to decide what animation to play.

Assets:

Because this game relies heavily on animations to convey how the baby is feeling (and hence the state of the game), we needed to make several unique animations representing different baby states. There is a graph of each state above called Animation Controller Graph. Each one of those states has a unique animation attached to it. "idle" has a simple, smiling baby animation. "cry lead-in" is the transition animation between smiling and a proper crying loop. "cry" is a repeating "wah wah" animation designed to go with the accompanying sound effect. "mouth open" 1, 2, and 3 all are simple animations like idle except each conveys a different level of spoon-to-baby proximity. When the spoon gets closer, the mouth gets wider. After the spoon has

made it all the way, the “chew” animation will play showing the baby receiving the food and then chewing it.

As a result of needing specific animations, we needed to make our own sprites to use in these animations. We found a generic cartoon image of a baby on Google image search to use as a base. After that we drew on our own faces for each of the sprites used in the animations. Some of these sprites are listed below for demonstration purposes. When drawing the facial expressions, we tried to capture the correct feeling for each sprite and also to make sure the feeling was clear and easy to recognize. If this means exaggerating the size of the mouth when the baby is waiting for food, then so be it.

Lastly we included baby sounds as a form of enrichment and to aid in the believability and connection to the virtual baby character. The baby makes crying sounds when crying and makes an “ahhh” sound when waiting for food with its mouth open. Like the other sounds used in this project, they were acquired from freesound.org.

Assessment

Challenges

Technological Limitations

RealSense:

Though the Intel RealSense is still a useful camera with a promising future, right now it is held back by technical limitations keeping it from realizing its full potential. The most glaring of which is the quality of gesture recognition and tracking done by the RealSense SDK. There is a fifth game we made that wasn’t included in the Software section above.

The second RealSense game we made is the Sandwich Making game. In this game, the stroke survivor would reach out to the screen in front of them. There is a hand shaped cursor that mirrors the stroke survivor’s hand’s side-to-side and up-down position on the screen. On the screen there is four piles of ingredients and an empty plate that needs to be filled. The stroke survivor needs to move their hand in front of a pile of an ingredient and make a grabbing motion (ie: make a fist). They then need to maintain this fist as they move their hand from the ingredient pile to the plate. They release the fist over the plate to drop the ingredient on the plate. Recognizing this type of grabbing gesture is supported in the RealSense SDK, but after trying to make the game we realized that the accuracy was simply too low for this game to be viable. When playing the game, it frequently drops recognition of a grabbing gesture. That means, a

stroke survivor would grab an ingredient and begin to carry it to the plate, but then as they are transporting the ingredient the game would recognize the false loss of the grabbing gesture and drop the ingredient. From our own internal testing we came to the conclusion this game is too awkward and unreliable for actual use at this time, but in the future when the RealSense's tracking software becomes more accurate, we might be able to use this game.

In addition to the problematic gesture tracking of the Intel RealSense, we frequently found ourselves wanting for a wider field of view on the RealSense's camera. The official specifications of the field of view are H: 73, V: 59, D:90. In practice, these did not prove to be as wide as we would like. They only really cover the space directly between the stroke survivor and the front display screen. It is not uncommon for the RealSense to drop tracking when playing because the stroke survivor's hand went outside of this space or got too close to the screen, hence leaving the cone of visibility. We hope that in later versions of the Intel RealSense, the dimensions of the field of view will be addressed.

We are also held back by the lack of proper object recognition and tracking. As mentioned in the specific games' sections, the tracking of functional objects we are currently performing is actually the tracking of hands that happen to be holding functional objects. Tracking the hands results in lower accuracy with occasional drops in the tracking (because the held object is occluding the hand). In addition, by tracking the hands we are unable to do any interesting interactions specific to certain objects. In the future we will need to develop a means of object tracking using the Intel RealSense.

Touchscreen and TouchScript:

The open source framework TouchScript for the Unity game engine has been incredibly helpful for our implementations of touchscreen games made in Unity. That said, there is a pronounced drop in sampling rate manifest in the Unity touchscreen games we developed using TouchScript in comparison to interacting with native Windows applications. Now we don't know for sure if this drop in sample rate is caused by Unity, which is known more for its usability than its efficiency, or by TouchScript, but we do know that TouchScript samples once every frame within the game. We suspect that the fact that the sample rate is tied to the game's frame rate is the cause of the decreased sample rate. This is something that required further investigation.

Challenges with assembling our desired Dual Screen prototype model

Our goal for the Dual Screen prototype is to package it in a suitcase like form-factor, much like the Nintendo DS gaming device (pictured to the right as a visual reference), sans buttons. The point of this is to make the prototype self contained and easy to setup for non-technical, potentially elderly users. The goal is to have a system where they just plug in a single cord, open up the system, and then get right into the rehabilitation games.

Packaging the TR console like this for easy transportation and setup by non-technical users is an idea that the people at UCI's NeuroRepair Lab have been considering for a while now. They haven't been able to test this feature out yet because they have more pressing matters to work on, but

being an offshoot, experimental branch from the project leaves us in the prime position to try out interesting ideas like this and test their viability.

To make this system, we needed a small computer to run the game software. Up until this point all the processing was done on a personal laptop belonging to one of the project members. We initially planned to buy a single board computer such as the Raspberry Pi or the Lattepanda. We realized though, that none of the single board computers were built to support multiple monitors. Because the multiple monitors are a core feature to the prototype we are making we decided not to get a single board computer at the time. Instead we bought a mini computer, the Minix Neo Z83-4. The Minix is a more powerful upgrade on single board computers, being designed as a personal PC device.

Here is where the first problem occurred with assembling our desired prototype. It turns out that the RealSense does not support any Intel Atom processor, Intel's processor for smaller devices and Internet of Things applications. The processor in the Minix Neo is a type of Intel Atom processor. As a result, the Minix is unable to use the Intel RealSense camera. The depth sensing RealSense provides is a core feature to our prototype system so as a result we were unable to use the Minix. We have, for the time being, returned to using a personal laptop to run our prototype.

The second challenge we faced was selecting the chassis. For this we bought a standard carry-on suitcase that was as small as can be while still including enough room to hold the components. This turned out to be a poor choice of container. The suitcase was too flexible and the hinge was simply not strong enough to support the heavy monitors we needed to install into the suitcase to make it the Dual Screen system. In the end, primarily because of this suitcase we were unable to assemble the portable model of the Dual Screen prototype.

Future Enhancements

Going forward, our main goals are to refine the Dual Screen prototype, incorporate playful social interaction elements into the Dual Screen prototype, and take part in trials with real stroke survivors to evaluate the use of the Dual Screen prototype. Refining the Dual Screen prototype will entail expanding the game library available on it which in turn entails adding some games that use the RealSense and the touchscreen together, adding more games that use functional objects, and adding games to cover as many different exercises as possible with the devices present in the Dual Screen prototype. We are considering using QR code tagging on functional objects in upcoming games to facilitate object recognition and tracking. It is not an effective solution nor is it a lasting one, but the present abilities of the RealSense are just not good enough for tracking arbitrary objects.

We believe the Dual Screen prototype is a good platform to study playful social interaction on. As such, we plan to incorporate the playful social interactive elements we researched earlier on into the Dual Screen prototype. This means we will add local and networked multi-user games and a revamped Emoji Chat client.

Lastly, after refining the Dual Screen prototype, we will consult with the therapists at the Neural Repair Lab to see if we can set up some initial screenings of the Dual Screen prototype with real stroke survivors to observe the system's usability and get the stroke survivor's feedback. If all

goes well, we will attempt to get funding to perform a clinical trial in parallel with the national trial currently taking place for the TR Console.

Conclusions

When faced with the loss of arm motor control, stroke survivors are left unable to perform daily activities they once took for granted. This can rob stroke survivors of their independence and self-worth. To combat this, we are researching augmented reality applications to game based stroke telerehabilitation. We believe game based stroke telerehabilitation can be an effective means of helping stroke survivors regain lost arm motor control in a fun and encouraging environment. In addition to this, we believe that by incorporating augmented reality elements into the games, we can make them more engaging for the stroke survivors who play them. Additionally, we can support positive recall and motion visualization by using functional objects to augmented stroke survivors' rehabilitation games.

Other implementations of game based stroke rehabilitation tend to either lack a breadth of rehabilitation capability or to lack the ability for stroke survivors to use the rehabilitation system independently. We sought to address both these issues when making the Dual Screen prototype. We selected our devices to be able to account for proximal, distal, and fine control. Additionally we designed the prototype to function as a telerehabilitation system, allowing the stroke survivor to be able to operate it independently.

When we set out, we aimed to make a versatile telerehabilitation system that utilized IoT actuators and sensors in addition to functional objects to create a system that could provide fun and engaging rehabilitation games for stroke survivors. We aimed to package this system in a "plug and play" suitcase form-factor for ease of use for potentially technologically inexperienced stroke survivors. We were able to achieve the core of what we set out to do: create a system that used IoT devices to create fun and engaging games for stroke survivors, but we encountered challenges with the incorporation of functional objects and with packaging the system in a suitcase form-factor. Though we did incorporate functional objects in two of the games we developed, their use was limited by the Intel RealSense's lack of a robust object tracking library. The suitcase form-factor is something we were unable to realize. We attempted to build it, but were faced with engineering challenges that we were unable to overcome. We were able to gain a better understanding of what is required from our failure for the next time we attempt to build this form factor though.

Our research has shown the existing potential for augmented reality applications to game based stroke telerehabilitation. We have illustrated the current limitations to the available technology, while also showing their present ability and future potential for game based stroke telerehabilitation. This was done through the production of multiple proof-of-concept prototypes produced by use over the span of our research. We showed the current limitations present in a tablet-based augmented reality "window", we showed the potential applications of social communication in game based stroke telerehabilitation, and we have shown a means of applying augmented reality through minimizing abstraction and including functional objects to game based stroke telerehabilitation.

Acknowledgements

This project benefitted from research support from the Undergraduate Research Opportunities Program (UROP) at UCI, along with collaborations and interactions with project team members In the Neural Repair Laboratory directed by Dr. Steve Cramer in the UCI School of Medicine working on the *Telerehabilitation in the Home Versus Therapy In-Clinic for Patients With Stroke,* ClinicalTrials.gov NCT02360488.

References

1. Adams RJ, Lichter MD, Krepkovich ET, Ellington A, White M, Diamond PT. Assessing Upper Extremity Motor Function in Practice of Virtual Activities of Daily Living. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*. 2015;23(2):287-296. doi:10.1109/TNSRE.2014.2360149.
2. Bower, KJ, Louie,J, Landesrocha, Y, Seedy, P, Gorelik, A, Berhardt, J. (2015). Clinical feasibility of interactive motion-controlled games for stroke rehabilitation. *J Neuroeng Rehabil.* 2015 Aug 2;12:63. doi: 10.1186/s12984-015-0057-x.
3. Chou, YK (2015). *Actionable Gamification - Beyond Points, Badges, and Leaderboards.* Octalysis Media, Fremont, CA.
4. Cooper, KM and Scacchi, W, (Eds.), *Computer Games and Software Engineering*, CRC Press, Taylor& Francis Inc., Boca Raton, FL (2015).
5. Evans, V. (2015). Are Emojis Becoming the New Universal 'Language'? *Newsweek Online*, 09/18/2015.
6. Givon, N, Zeilig, G, Weingarten, H, Rand, D. (2016). Video-games used in a group setting is feasible and effective to improve indicators of physical activity in individuals with chronic stroke: a randomized controlled trial. *Clin Rehabil.* 2016 Apr;30(4):383-92. doi: 10.1177/0269215515584382. Epub 2015 May 7.
7. Hondori, HM, Khademi, M, Dodakian, L, McKenzie, A, Lopes, CV, Cramer, SC (2016). Choice of Human-Computer Interaction Mode in Stroke Rehabilitation, *Neurorehabilitation & Neural Repair*, 30(3), 258-265
8. Hung, YX, Huang, PC, Chen, KT, Chu, WC (2016) What Do Stroke Patients Look for in Game-Based Rehabilitation: A Survey Study. *Medicine (Baltimore).* 2016 Mar;95(11):e3032. doi: 10.1097/MD.0000000000003032.
9. Kapp, KM (2012). *The Gamification of Learning and Education: Game-based Methods and Strategies for Training and Education*, Pfeiffer, San Francisco, CA.
10. Pietrzak, E, Cotea, C Pullman S (2014). Using commercial video games for upper limb stroke rehabilitation: is this the way of the future? *Top Stroke Rehabil.* 2014 Mar-Apr;21(2):152-62. doi: 10.1310/tsr2102-152.
11. Putrino, D (2014). Telerehabilitation and emerging virtual reality approaches to stroke rehabilitation. *Curr Opin Neurol.* 2014 Dec;27(6):631-6. doi: 10.1097/WCO.000000000000015
12. Rand, D, Givon, N, Weingarten, H Nota, A Zeilig, G (2014). Eliciting upper extremity purposeful movements using video games: a comparison with traditional therapy for

stroke rehabilitation. *Neurorehabil Neural Repair.* 2014 Oct;28(8):733-9. doi: 10.1177/1545968314521008. Epub 2014 Feb 10.

13. Scacchi, W, (2015). The Future of Research in Computer Games and Software Engineering, in K. Cooper and Scacchi, W, *Computer Games and Software Engineering*, CRC Press, Taylor & Francis Pubs. (2015).
14. Scacchi, W (2016). Game-Based Stroke Telerehabilitation: Challenges in Scaling to National Clinical Trials. *2016 UK-US Games for Healthcare Workshop*, Drexel University, Philadelphia, PA, 22 March 2016.
15. Seo, NJ, Kumar, A, Hur, P, Crocher, V, Motawar, B, Lakshminarayanan (2016). Usability evaluation of low-cost virtual reality hand and arm rehabilitation games. *J Rehabil Res Dev.* 2016;53(3):321-34. doi: 10.1682/JRRD.2015.03.0045.
16. Winstein CJ, Stein J, Arena R, Bates B, Cherney LR, Cramer SC, Deruyter F, Eng JJ, Fisher B, Harvey RL, Lang CE, MacKay-Lyons M,Ottenbacher KJ, Pugh S, Reeves MJ, Richards LG, Stiers W, Zorowitz RD; American Heart Association Stroke Council, Council on Cardiovascular and Stroke Nursing, Council on Clinical Cardiology, and Council on Quality of Care and Outcomes Research. (2016). Guidelines for Adult Stroke Rehabilitation and Recovery A Guideline for Healthcare Professionals From the American Heart Association/American Stroke Association, *Stroke.* 2016 Jun;47(6):e98-e169. doi: 10.1161/STR.0000000000000098. Epub 2016 May 4.