Mario Fun
IML4HCI Final Project Report

Kushal Joshi, Jake Liebow, Jorit Pfleiderer, Naif Alharthi

The Idea and Mission Statement

→ What problem are you solving?

Accessible Gaming varies dramatically from game to game, and system to system. One game that most people easily recognize is that of Super Mario, a side scrolling action adventure game. Our vision begins with understanding how to make playing a 2D game like such, an easier and better experience for a person with a disability. With nearly 1 in 5 Americans suffering from some kind of disability our aim is to make sure we make a way to play a side scroller accessible not only to one person or small group of individuals, but instead to a very large number of people. Our goal and aim, to bring a somewhat, unadulterated experience to those who have a disability but enough motor functions to do some basic tasks. We aim to integrate various types of controllers to play games like Super Mario, Sonic and nearly all other Side Scrolling Games.

→ What is the context / circumstances surrounding the problem?

We want to combine our two passions of gaming and helping those who have some sort of disability. We strive to create a experience that allows those with disabilities a way to enjoy games that are simple enough to pick up and play, but have an experience that makes it even more enjoyable for anyone who may have a disability.

→ How will the customer / sponsor benefit from your solution?

Our focus is on those individuals who have a physical disability of some sort. It is understandable that people who have a physical disability are don't enjoy the gaming experience as it is catered to non-disabled people, using Machine Learning and various controllers, we would like to create a system that makes side scrollers perfectly catered to each disability for each individual.

→ Why is your project feasible?

The controllers we are looking to use are the MicroBit, the Myo Band and the Tobii Eye Tracker, all of which we have used in previous projects. We have a grasp of the controllers and integration of side scrolling games with OSC messaged is something we believe we can tackle. Everything we have set forth seems feasible and can be tackled in a orderly fashion

→ Why is machine learning a good fit to your problem domain and product vision?

Every person and every disability is different, and every disability looks different on a per person a basis. Understanding this and taking into account, we look to Machine Learning to provide an experience that the user would cater for themselves in a simple and elegant manner and thus creating the perfect experience for themselves.

Challenges you think you will have to overcome

→ What kinds of challenges will you face?

Deciding on building an intuitive controller will be a challenge in itself. But our main delima/challenge comes in the form of allocating time and manpower, should we either creating a sub-par experiences with all the various controllers, or a great experience with just one kind of controller. Our last challenge will lie in making a GUI that both looks and feels intuitive enough for our target audience.

→ learning a new language or languages

The languages we look to use, at least one of more of our group members has experience with and therefore we see little trouble in this forte.

→ interfacing with hardware

All the controllers/hardware we have proposed to use have been used in the past by our team and the only challenge will be smartly mapping them to make great usable controllers.

→ dealing with limitations on the platform (e.g., iOS/Android)

We will be focusing on the Windows Platform for a majority of the time and shouldn't run into any problems on that front.

→ Calibration

Because our experience will be unique to each user, we will require a calibration on a per person basis. Making sure that this process of setting up is both intuitive and enjoyable is key for a great all around user experience and a challenge for us to combat.

The technologies you think you will need

→ What is it going to take to engineer the experience?

To engineer the experience we are going to need to focus on how we can take video game features and incorporate them into a device that feels natural. While training should be done by the user to better the experience for them, controllers still need to be efficient and explanatory enough to be used as a controller intuitively.

→ Are you going to need hardware?

Hardware wise we are focusing on MicroBit, Myo Band and Tobii Eye Tracker, all of these we aim to make controllers in our project. Though we may need to understand tackling all of these might not be the best idea and instead concentrating on one will yield the best results.

→ Are you going to need language tools / IDEs?

For languages we will use Python predominantly and for IDEs will rely heavily on Visual Studio and Visual Studio Code.

→ Algorithms used and why

We used a neural network to train our data as it had the least amount of noise associated with it.

Risks to failure

→ How could your project fail? E.g.,

While our risks of failure are very minimal we see our largest risk to failure being not implementing controllers in a correct fashion and getting to caught up on implementing all controllers over just one very well.

→ Technology doesn't work (hardware or software)

We don’t think we will face any technology hurdles on the hardware front as we are familiar with the technologies and have used them in previous projects.

→ Not enough time to implement

Getting caught up on implementing all controllers over just one very well, is our largest hurdle in terms of not having enough time to implement.

→ Problem too hard to solve

The hardest task we think will be making a GUI interface that is simple and intuitive enough for the user to use. Also getting a program talking to Wekinator to train for models and run the Wekinator program may be a difficult task.

Ethical questions

→ Distraction (for vehicular applications)

- Obviously we would not recommend using our software while operating a car or other forms of dangerous/heavy machinery
- Should not be used for actions of abuse or harassment, should not allow the user to create and harmful content or material.

→ Security (what will happen if your database gets hacked?)

- Nothing, we don't store relevant information except for maybe username, and password information if we even deem it necessary to implement a database, which at the moment we don't.

→ Hardware failure (drone flies into orphanage picnic)

- As a worst case scenario if our hardware fails then the user would not be able to play the game. Hardware failures in our proposed project are not life or death situations but would ruin the experience for the user.

Open Source and Closed Source Components

- **Wekinator -** http://www.wekinator.org/ (free, open source software originally created in 2009 by Rebecca Fiebrink. It allows anyone to use machine learning to build new musical instruments, gestural game controllers, computer vision or computer listening systems, and more. The Wekinator allows users to build new interactive systems by demonstrating human actions and computer responses, instead of writing programming code. (http://www.wekinator.org/)
- MicroBit - (http://microbit.org/) is an ARM-based embedded system. The board measures 4 cm × 5 cm and has an ARM Cortex-M0 processor, accelerometer and magnetometer sensors, Bluetooth and USB connectivity, a display consisting of 25 LEDs, two programmable buttons, and can be powered by either USB or an external battery pack.The device inputs and outputs are through five ring connectors that form part of a larger 23-pin edge connector. (https://en.wikipedia.org/wiki/Micro_Bit)
- Tobii Eye Tracker - An eye tracker knows what a user is looking at which makes it possible to gain insight into human behavior or interact with computers. (https://www.tobii.com/tech/technology/what-is-eye-tracking/)
- Myo Band - The Myo armband is a wearable gesture control and motion control device that lets you take control of your phone, computer, and so much more, touch-free. (Myo Gesture Control Armband | Wearable Technology by Thalmic Labs)

The Timeline

→ The Concept Phase

I. What are your user's requirements?

-Creating an experience that the user feels immersed in the gaming experience is key for our project. The user expects a gaming experience that is flawless in both setting up and playing with few bugs or hiccups.

II. What are the product concepts?

We conceptualize a product that makes it easy for a wide variety of disabled people to play games and each experience being unique to each individual, with multiple options for controllers they could use.

→ The Planning Phase (Items should be done prior to following date)

1. Deciding on exact parts of the project, after deciding what exactly we would like our project to look like.
2. Purchase/Find the physical parts for our product
3. Incorporating Machine Learning into the product, what it may look like, how it changes and learns from user experience.
4. Creating a physical prototype as well as begin coding to make project a reality
5. Iterate on step 4 until product is completed and learn from feedback on what can make the project better.
6. Have a finished product that is presentable and can be controlled with easy setup and infinite replayability.

→ The Design & Development Phase Timeline

3-19-18	TIME SPENT ON PREVIOUS PROJECT IDEA (TOUCHPAD)
3-26-18	SPRING BREAK
4-2-18	Build Prototype, concentrate on computer/emulator interactions. (Document progress in a changelog.md, with an explanation of each person's contribution. Include pictures and videos.)
4-9-18	Iterate on prototype, get the computer to speak with the emulator or website alongside wekinator. (Document progress in changelog.md, with an explanation of each person's contribution. Include a link to a demo video on youtube. Have class demo ready)
4-16-18	Iterate on prototype, implement hardware and see which items work best with the system so far.. (Document progress in changelog.md, with an explanation of each person's contribution. Include a link to an updated demo video on youtube. Be prepared to offer hands-on demosin class.)
4-23-18	Begin final iteration, implement a GUI to tweak wekinator settings, using either Python or Java. Document progress in changelog.md, with an explanation of each person's contribution. Include a link to an updated demo video on youtube. Work on the draft of final report on peer review system.

4-30-18	Prepare for final presentation and polish anything that may need to be polished, concentrate on one controller for the final demo (most likely the MicroBit). 13-minute final presentations. 7 minutes of talk + 3 of demo + 3 minutes of questions.

What you accomplished (including information about what is open source.)

Starting with our original project idea and proposal, our TrackPad idea started as a great idea connecting music and those who lack musical ability. As we began Spring Break we quickly realized that with the lack of musical understanding and ability on the team made this project unachievable. Towards the end of Spring Break we realized that we needed to pivot to a new project idea, one that would also take into account the fact that we had only a few weeks left of school, something manageable. We decided to shift gears to something we all loved, video games, more specifically we shifted to 2-D Side-Scrollers. A side-scrolling game, side-scroller or 2D is a video game in which the gameplay action is viewed from a side-view camera angle, and the onscreen characters generally move from the left side of the screen to the right (or less commonly, right to left) to meet an objective. (https://en.wikipedia.org/wiki/Side-scrolling_video_game). While side scrollers are simple games that require simple controller configurations to play, we wanted to tackle the accessibility problem of not everyone being able to playing with a traditional keyboard or controller. Our aim became to to provide a user experience that is unique to every player and every disability and also realized that machine learning would be perfect for this situation. Every player would train the computer how they want simple controls like left, right, up and down to be in the physical world, and then with this training examples they are able to play from a variety of side-scrollers. Our first week we were able to get wekinator talking to a online based side-scroller using some processing code, it was very successful and got us excited for all the other stuff that may be possible. OSC messages were sent out, mimicking keystrokes and the online based emulator would be able to move the Mario character. We decided to start by sending in OSC messages with a face tracker, being able to move the Mario character using the X-Y coordinates of our face. From classmate feedback that week, we learned people loved the idea and wanted to see more! The next week we moved to implementation of controllers such as the Micro-Bit, Myo Band and Tobii Eye Tracker. We were successful in getting each on talking to Wekinator with processing code. This week we learned that it was hard to implement 3 different technologies as controllers and do a good job at it, as there varying factors that almost caused some controllers to be better than others. That week we demoed the Micro-Bit and were told by classmates that it was best to concentrate on just the Micro-Bit and not worry about the other controllers. We shifted all focus to the Micro-Bit and wrote all processing code and did all testing around the Micro-Bit. The week after we began working on the code for the GUI for users to train the different motions of the controller, the GUI sending OSC messages to Wekinator to do so. We began to run into lots of issues, cramming to make the GUI with the limited weeks we had left. When we finally finished we were able to create a GUI processing and was very user friendly and took care of training wekinator. We were able to create the GUI in Processing, though we did run into the problem of it not being able to open up wekinator upon launch (which needs to be done manually). Our finished product was something we were proud of and those who tested our finished product out were amazed at how well it worked. While it is not 100% refined, given more time we would have loved to refine the GUI further, implement more controllers, and even try to add a second controller for some local multiplayer action.

These are snippets from our Changelog, and help show what we did on a week to week basis in a better, easier to read format:

April 2nd 2018	-Pivot from original project idea, have now moved to combatting accessibility problems for players playing 2D side scrollers -The idea aims to combat the issue of some players having physical difficulty playing these sorts of games -Our aim is to provide a user experience that is unique to every player and every disability and thus Machine Learning would be great to combat this problem -Wekinator takes in various inputs and sends out OSC signal (which mimic keystrokes) -For our in class demo we prepared playing the popular side scroller Mario, and used X-Y coordinates of our face to move the character as well as make Mario Jump
April 4th 2018	-This week our emphasis grew on incorporating multiple devices to play these side scrollers -We worked directly with the MicroBit Controller as well as the Tobii Eye-Tracker to make devices you can play side scrollers with -We also discussed ways to send keystrokes to a emulator, allowing for the playability of all 2-D Side Scrollers, not just online based 2D games -We talked about how we need to create a program to send and train Wekinator, deciding what the UI may look like, as well as what the user experience would look like for that program
April 11th 2018	-Video with content: https://youtu.be/5_Sw3kIZUX0 -Updated Repo with folder containing feedback -Worked on wireframing a GUI to train wekinator -Will be also working on creating the GUI based trained for Wekinator over the weekend. We most likely are looking at HTML based items at it would be easy to send packets of data through a port into wekinator this way

	-After last weeks feedback we are concentrating on refining the MicroBit experience and then moving to other technologies, we want to make the MicroBit experience refined and perfected before diverting our attention elsewhere.
April 18th 2018	-Video with content: https://youtu.be/6d6kb41GQ70 -Scrapping the HTML GUI interface, shifting to higher order language like Java/Python -Making MicroBit communicate with computer wireless -Updated Repo with folder containing feedback -Concentrating on making the whole experience better, looking at other games to incorporate (we have been using Mario but would like to extend more options to the user)
April 29h 2018	-Finished Final Project -Worked specifically on the refining existing features on the project -MircoBit fully wireless and uses Java to now send messages to wekinator -GUI built to train wekinator for users with little to no experience with wekinator -Worked on Final Proposal and Video (to be submitted on May 4th)

Final Project Report

Goals

Our goal began with understanding that Accessible Gaming varies dramatically from game to game, and system to system. Our vision begins with understanding how to make playing a 2D game like such, an easier and better experience for a person with a disability. With nearly 1 in 5 Americans suffering from some kind of disability our aim is to make sure we make a way to play a side scroller accessible not only to one person or small group of individuals, but instead to a very large number of people. Our goal and aim, to bring a somewhat, unadulterated experience to those who have a disability but enough motor functions to do some basic tasks. We aim to integrate various types of controllers to play games like Super Mario, Sonic and nearly all other Side Scrolling Games.

Design (UI/UX)

Our final designed, while not as polished as we wanted it to be, turned out to be very decent. We created a responsive GUI that allows the user to select what type of training they want. The user can customize what controls are to be mapped using our controller. The GUI then sends the inputs to Wekinator automatically. Unfortunately this whole process wasn't without its drawbacks and tradeoffs needed to be made. The controlP5 library works well with backend, but is not as customizable as other methods. We decided to build the User Interface into the same file as our backend, which in the grand scheme of things works, but is messier than splitting the backend and frontend up. We also explored alternatives including Python GUI libraries, C++ Framework, but at the end of the day our implementation was most logically and was easier to talk to wekinator to. Our User Interface also requires very little physical inputs, and sets up training autonomously once the required types of controls are entered.

Describe your engineering work

After a long tedious 5+ weeks we were able to create a final product we were proud of. We developed a lot of our code in Processing. Our file home.pde acts as both frontend and backend which ended up working best for us. This file is responsible for setting up the wekinator environment, and it learns and assigns output values to key presses. For Machine Learning we relied on a neural network to train our data. The reason we decided to go with a neural network was that it had the least amount of noise. The process of Machine Learning begins with taking a series of accelerometer data and maps certain accelerometer values to certain actions, these actions include Run, Jump, etc. The actions correlate to what actions would be reciprocated in the side scrolling game and would vary from game to game depending on which game you decided to play. Our feature engineering included the Accelerometer values from the Micro Bit controller. While our machine learning could have expanded further had we had time to implement more controllers, we were only able to fully implement the MicroBit controller.

Results

Our system takes however many controls you need to play a game, and walks you through the training process to map each of these controls to a real distinct motion or position recorded by the microbit. We wanted to make the Training process as simple as possible and succeeded somewhat. The set up for this program can be accomplished in 14 easy guided steps, no technology background is required to use this technology. The controller has been found to be very reactive and accurate when classifying inputs to outputs however, we still have yet to find a decent mechanism for running while jumping and hope to address this minor issue in the week to come.

Lessons Learned

Our group learned a lot more about wekinator and also learned a thing or two about team dynamic. We gained a deeper understanding of Machine Learning as well as GUI building, which we all were new to prior to this project. Overall this project was very insightful and a great resume builder as well as a great way to understand more about Machine Learning.

Future Work

Given more time we our team believes we could have done a lot. First and foremost cleaning up existing code would have been the main goal. Refining the UI to make it a seamless experience would have been key. Refining the GUI to provide an easier way train wekinator, also making the GUI very intuitive with lots of user testing. Gamification features for better UX and elaborating on the existing UI would be a great way to further engage the user. We also had started with the goal of being able to implement a variety of controller based on a person disability. Implementation of variety of hardware for more functionality, given more time it would have been awesome to incorporate more hardware so more users could play games. These pieces of hardware include the Tobii Eye Tracker, Myo Band and a Webcam.

Bibliography

Wekinator, www.wekinator.org/.

> As a source throughout the semester, Wekinator has proven time and time again to be extremely useful for Machine Learning. We look to it again for our final project as a tool to incorporate Machine Learning into our project. The website itself provides detailed documentation on the Wekinator system as well as lots of various examples to helps understand inputs and outputs better.

MicroBit, http://microbit.org/.

> https://makecode.microbit.org/ on the MicroBit site helps us get code up and running on microbits with ease. The MicroBit website also provides tons of documentation on the MicroBit.

Processing, https://processing.org/

> Allowing us to view various documentation about Processing and Forums filled with information about writing programs with processing. Our GUI was processing based and this site was key in helping us understand processing as a whole.