Wii-B-Fit Adaptive Wii Remote

Seth Black, Craig Leitterman, Jamie Nease, Canh Sy, Mike Tran
Computer Engineering Program
California Polytechnic State University
San Luis Obispo, CA
wii-b-fit@googlegroups.com

Abstract—It is difficult for people with quadriplegia to get the physical activity necessary to stimulate brain activity and prevent diseases associated with the paralysis. Individuals with the condition can often feel alienated from peers due to limited abilities to participate in games and social activities. There aren’t many activities that allow for independence in their lifestyle. Wii-B-Fit’s mission is to give people with paralysis the ability to participate in activities that they would not otherwise get to do, while creating a more fun, social, and independent lifestyle. The goal of our project is to create a universal design that is applicable to all levels of quadriplegia, and allows for Nintendo Wii game-play with minimal assistance required for setup. This paper discusses the steps taken to achieve this goal, going into depth about the technical aspects of our project. We go over the design, implementation, and results of our transformation from a regular Wii remote into a fully adaptive remote.

Index Terms—accelerometer, adaptive switches, IR sensor, quadriplegia, wii remote, wii sports

I. INTRODUCTION

Many video game consoles have been hacked to provide adaptive abilities to those who are paralyzed, but few also offer a workout routine and social setting. The Wii was created with the intention of providing a social atmosphere for a video game platform, as well as a way to move around and get exercise. These two aspects make it different from any other video game console on the market, and are what make it ideal for a person with quadriplegia to play. Unfortunately, these aspects also make adapting the console more difficult than others.

Adapting the remote to someone without any arm or leg movement required using face sensors in place of the ordinary buttons, and placing the accelerometer on the head so that head movements correspond to arm movements. Then, we had to boost the accelerometer readings so that weaker head movements can be read as ordinary arm movements that the console expects. We wanted to utilize the stock remote as much as possible in order to avoid reimplementing the Bluetooth communication between the remote and the console and to avoid learning that particular protocol, since there is no documentation of it online. This reduces the chances for error while making the overall design simpler and more intuitive.

To make all of this possible, we needed to add extra computing power and to make minor modifications to the remote. We use an ATMEGA1284P microcontroller to intercept the accelerometer readings, amplify them to sufficient values, and to output them back to the remote. We needed the movement sensors, the accelerometer and IR sensor, to be placed upon the user’s head to detect movements. This meant removing the IR transmitter and the accelerometer from the remote, and placing these on top of the head in a way that wouldn’t interfere with the user’s head movements or cause any neck pain. We chose to place these on an ordinary baseball hat that was comfortable and fit snuggly on the client’s head. We have long wires that extend along the side of the hat and down the back of the client’s head, so that they are out of the line of sight. We have chosen to include a large push button for our client and a puff sensor to map to the two most commonly used buttons on the remote (‘A’ and ‘B’). We use velcro attached to the puff sensors tubing and the hat so that the clients can adjust the tube to the proper position in front of their mouths. We have also provided connections for adaptive buttons for each of the four D-Pad buttons. This way, the client may choose any switches they would like to be used to control the menu and for directional control in games. We also provide a flex switch and four push buttons so that the client has a variety of switches to choose from.

The game we have chosen that provides simple movements and minimal button presses is Wii Sports, focusing on bowling and tennis. Both games will give the client an ample workout, while also creating a fun social setting. Theoretically, our system will be capable of playing any Wii game that does not require Wii remote add-ons, such as the nunchuck. The nunchucks can not be used, because we do not have two body parts to mount accelerometers to that have independent motion. However, the rest of the Wii Sports games require the same buttons and motions as the ones we are designing for.

This document contains detailed information on the design and testing of our adaptive Wii remote. Section II. discusses prior work done on the subject, Section III. talks about our approach to the design, Section IV. details the tests we ran, Section V. displays the results, and we conclude the paper in Section VI.

II. RELATED WORKS

A. Gesture Recognition with a Wii Controller [1]

The authors of this project discuss the ways that gesture recognition using accelerometers can be applied directly to the Wii Remote. They describe a way for a user to train the application into recognizing specific gestures using a hidden Markov model. They study arbitrary gestures performed by the user using only one hand. This project relates to ours in certain aspects. We originally thought we were going to recognize gestures made on the Wii remote, but we ended up using a
much simpler algorithm. This was a great starting point for seeing the kind of work required to interpret movements.

One of the key techniques that make this project effective is the system pipeline created to analyze the accelerometer data. The pipeline consists of three stages: quantizer, model, and classifier. Before the data is sent into the pipeline, it is first filtered. The filtering is divided up into two stages, where the first stage discards any vectors which indicate an idle state. The second stage of filtering gets rid of any vectors that are very similar to the previous vector. This creates a gesture with only characteristic vectors that can be used for recognition. The first component in the pipeline is the quantizer, which clusters and abstracts the data so that it is ready for the next stage in the pipeline. This is the model phase, which uses a hidden markov model to train the application to be able to recognize gestures that the user wishes to use. The classifier then takes the possible gesture options and narrows it down to one choice.

In evaluating their process, they used only one woman and five men, none of whom were disabled. They had five predetermined gestures that the used in their studies and that they had the users train into the application. These gestures were a square, circle, roll, Z, and tennis serve. In their results, they found that it correctly recognized gestures, on average, 90 percent of the time. The recognition rate also varied between the different gestures, tending to be more accurate for more unique gestures. The two most accurate gestures were the Z and the tennis serve. Accuracy also varied between users, meaning that some users might not have been reproducing the gesture in the way that they trained it to.

B. Detecting Gesture Force Peaks for Intuitive Interaction [2]

This project focuses on using simpler gestures in place of “haptic force gestures” currently in use by the Wii console. This system will allow for more intuitive gestures to be used to navigate Wii menus and play the games.

The authors discuss the difference between the intended and rebound peak of force in a gesture. The intended peak comes from a force in the direction the user wishes for the console to pick up on, where the rebound peak comes from the deceleration of the remote back to stable. The current console cannot distinguish between these two force peaks. The goal of the paper is to distinguish between these two force peaks, and cancel out unwanted peaks. This kind of recognition is a large part of our project. When boosting accelerometer values, we end up getting unwanted peaks read and interpreted as movement in the console.

They hope that instead of using advanced gesture recognition, simple flicks could be used for navigation. The authors state that to swing a tennis racket in Wii Tennis, a force two times greater than gravity must be used. They use a technique which identifies the average resting position of the remote, and recognizes the start and end of a gesture by the times when the remote is in its resting position.

The study they performed involved a two-item menu, where the user would select an item using the typical pointing and then using gesture flicks. On average, the gesture pointing was faster and more accurate, but not by much. This could be due to the fact that the menu was so small, so navigating with the pointer was not very difficult.

III. Theory/Application

A. System Overview

Our system consists of three main components: the user, the microcontroller, and the Wii remote. This can be seen in our block diagram in Figure 1. The hat is placed on the user’s head and a lapdesk placed on his lap. The accelerometer is placed on the hat’s bill, with long wires extending out of the back of the user’ s head and down to the breadboard. The Wii remote is mounted in a black box which contains the microcontroller and breadboard. The accelerometer on the head replaces the accelerometer on the Wii remote, so the wires go to the pads on the remote that correspond to those connections.

The lapdesk holds all of the adaptive buttons used by the client during game play. It contains large push buttons that can be used for all six buttons that we have adapted on the remote. These are the four directional D-pad buttons, the A button, and the B button. For users without arm movement, they can use the puff switch and the flex switch. The puff switch is mounted to the hat, and the flex switch can be mounted to a tripod. The puff switch goes in the mouth, and the flex switch may be mounted above the eyebrow or cheek, or below the chin, and only requires a slight muscle movement to be activated. The D-pad buttons are mounted to the sides of the head rest on the user’s wheelchair.

The microcontroller and breadboard are contained in a small black box so that the connections are not altered during game play. We utilize an STK500 development board in order to access the ports of our microcontroller more easily. The breadboard also contains our voltage converters. All that is accessible in the black box is the power switch and cord to turn the system on. The box must be placed on a table next to the user while playing the game, so that the long wires extending from the box to the head can reach all the way to the head.

![Fig. 1. Block Diagram of the system](image-url)
B. Accelerometer Theory: What constitutes a swing?

Very early on in the design of the project we noticed that head movements could not register results in the Wii games we were testing with. Even a test subject with a healthy strong neck could only produce very low power swings and bowling speeds in games. We would have to modify the accelerometer readings being sent to the game software to allow individuals with quadriplegia to participate. One of our goals early on was to determine what constitutes a swing in tennis for Wii Sports. We had two hypotheses as to how the game software interprets a swing. One, the software looks for a geometric movement over a specific period of time or two, the software looks to see if an acceleration threshold was met or exceeded in a specific direction. It was apparent from our testing that having the software interpret movements over a specific period would be difficult to compute from a game point of view. We also noticed we could play games with the flick of our wrist as opposed to using full arm motions. We decided to test and verify our second hypothesis: acceleration thresholds.

With this in mind we designed a series of tests using Brian Peeks Wii test software [3] to determine the amount of acceleration in g’s required to make a swing in the game. We would later find out the software was not quite right in its measurement of acceleration. It showed readings above 5G but the Wii remote’s accelerometer can only sense up to 5G in any direction. However these early tests still allowed us to figure out a multiplier to boost the accelerometer readings by in order to compensate for the weaker head movements. This would be based on our client’s head swing vs. the required amount to register a swing. We determined we would need to boost the signal by at least two times the normal amount. Another problem with the test software was that values were displayed in G of acceleration and we needed to break that down into voltages.

The accelerometer we used for the project is the one similar to the Wii remotes, the ADXL330 [4]. Since Brian Peek’s test software’s measurement of acceleration was imprecise as stated above, we had to figure out what voltage threshold could produce a swing. We first removed the Wii remote’s accelerometer. Second, we utilized multiple power supplies to simulate an individual swinging the remote. From the ADXL330 datasheet, we knew the accelerometer had a zero G rest state of 1.5V or half the input voltage and it read accelerations at 0.3V/g. We started by hooking up 1.5V on both X and Y and 1.8V on Z (0.3V higher due to fighting gravity). We first tried modulating a single direction X but with varied results. No consistent swings were shown, even at maximum voltage modulation. We turned up Y’s base voltage to approximately 1.5 G or 1.95V and then modulated X from 1.5V to 3V. This yielded consistent swings and showed that a swing was not just a function of single direction but rather a combination of multiple directions. We also noticed that if you kept the voltage high without resetting it to or passing by the rest state you would not get multiple swings. This is smart on the developer’s part because when a person swing’s a racket in real life they return to a rest state or pass through a rest state on their way to the next swing. In the case of the Wii remote, we needed to change the voltage down to around 1.5V and back up to 2V-3V for a forehand swing or down between 0V-1V for a backhand swing on a right-handed Mii character.

C. Software Design and Accelerometer Modulation

Based on our theory and preliminary testing, our software algorithm would be relatively simple. An accelerometer is placed on the bill of a hat worn by the user. The values from the accelerometer are read by the microcontroller that has onboard analog to digital converters (ADC). The signal coming in would only be boosted on the output if it was outside the rest state. We did not want to be boosting or modulating voltages that were near the normal rest state as this would cause the game not to function properly. If we were to boost the rest state voltages, we would in turn be boosting gravity and causing the console to perceive motion when there was none. Since 1.5V is sent in at a rest state, we could not double that and send it out. It would just feed the Wii remote bad readings and, in this case, max readings at rest state. Instead, we convert the voltages coming from the accelerometer to a 10-bit value. So 1.5V converts to 500 in digital value, which is our rest state. The conversion from voltage to digital value is 100 per 300mV. So a voltage of 1.8V converts to 600 in digital value.

Equation (1) gives details to make the conversion from voltage to a 10-bit value.

\[
\text{readings in } V = \frac{100}{300mV} \times \text{digitized value} \text{ (1)}
\]

Equation (2) gives details of how readings from the accelerometer get boosted. So when the accelerometer reading comes in, (1) converts the voltage to a 10-bit value and (2) boosts the 10-bit value. If \textit{boosted value} is out of range of 10-bits, then the boosted value will be at the maximum or minimum value for 10-bits. For our design, we set \textit{boost} to 8 and \textit{at rest} to 500. We also did not want to amplify the accelerometer readings too drastically, therefore we decided upon amplifying the difference between the current measurement and the nominal resting value. We then multiply this value by our boost, add it to the nominal at-rest value, and output that value.

The voltage that has been modulated digitally is then sent out of the microcontroller to three parallel digital to analog
converters (DAC), one for each axis. This signal is then routed to the traces where the Wii remote's original accelerometer leads used to be. The DACs are controlled by separate output ports in the development board, utilizing the chip select and write pins on the DACs.

![Boosting algorithm design](image)

**Fig. 2.** Boosting algorithm design

**D. Adaptive Switches**

Since individuals with quadriplegia have limited to no dexterity in their hands, we have to use adaptive switches to allow them to play the game. Most games primarily use the A and B button and sometimes the D-pad. Upon taking apart the Wii remote we noticed when a button is pressed, it closes a circuit on the board using a conductive material and is registered by the remote and, in turn, the console. Most adaptive sensors, such as the puff sensor and large push button, have a 1/8th inch stereo jack interface. This interface can be broken down into two leads using a female interface port. Each lead is then soldered onto a separate side of the pad and pressing the button down or puffing into the tube you can close the circuit and register a button press. This allowed us to use a wide variety of adaptive switches for multiple clients with varying ranges of quadriplegia. In total, the device allows for plug-and-play ease by rearranging sensors to meet the exercise regiment and/or playability preference of the client. We have left the rest of the buttons on the remote intact, so that whoever is setting up the games can turn the remote on, pause or end the game, and get back to the home screen. Our adaptive switch interface has effectively made our system universal and usable by all people with disabilities.

**E. Physical Head Mount**

Our physical layout went through a number of iterations and ideas. We knew that the targeted user has most of their range of motion in their head and neck. From this, we deduced we would need to mount the device on a hat of some sort, while making sure to place the accelerometer as far away from the center of the head as possible. This ensures that the accelerometer is placed as far away from the focal point (center of users head) in order to produce the largest changes in acceleration. We started with the obvious, by proposing to place the whole Wii remote on top of the users head. This produced a number of troubles. Although the Wii remote without batteries is not necessarily too heavy, it is awkward, long, and does not fit tightly to the bill of a hat. We then proposed a more sleek and elegant design where only the IR and accelerometer were placed on the bill of the hat. This proved to be a more complex design and we knew if something did not work we could fall back to the previous solution. The IR, however, became a problem and never worked when removed from the Wii remote. This led to two possible final solutions: 1) adapt the whole Wii remote and make it as light as possible or 2) only place the accelerometer on the bill of the hat. By placing the whole Wii remote on the hat, we would have more functionality for the system and allow navigation of main menus and games that required the use of an IR. This, however, brought us back to our original problems with this design. The button jacks created an uneven weight distribution and more strain on the neck. We went with the alternate solution with just the accelerometer because it was sleeker, lighter, and it was possible to play all the games we had set forth in our proposal. We included four buttons for the D-pad to allow the client to navigate the menus as the IR normally does, so that no functionality is lost.

The accelerometer is mounted on a breakout board that is securely fastened onto the bill of the hat with nylon screws. The wires from the board are routed away from the user with fabric straps. There is also a section of Velcro on the other side of the hat to allows the user to attach a puff sensor to it.

**IV. TESTING**

Testing consisted of three parts: button testing, IR testing, and accelerometer/software testing. These also included testing the engineering requirements that were created when we first began designing the project, and are scattered throughout all of the component testing. Once we tested each component individually, we began to test the product as a whole.

**A. Button Testing**

Preliminary tests showed that the buttons and switches work when plugged into a 1/8” headphone adapter that was in turn soldered to the remote. Our final testing ensured that the client can use the buttons effectively during menu navigation and throughout the games. To quantify this result, we pressed each button 10 times to ensure 100% functionality. We had the client navigate the menu, and play both of the games and recorded and addressed any errors in button presses.
B. IR Testing

Although we did not end up using the IR in our final design, we have included the tests we ran here as well as the results of these tests in the next section to show the process we took. To test that the soldering and rewiring of the Wii remote into the finished product was done without error or damage, we used continuity tests on each of the eight leads to ensure that all leads were soldered properly and nothing had been short-circuited. We connected each lead from the board to its counterpart lead on the IR to ensure that there was a connection between the two solder joints. After this, we connected each lead to every other lead on the IR to ensure there were no extra connections that were not supposed to exist. We also examined each solder joint carefully under a microscope to ensure that there were no physical shorts between any of the leads.

C. Software Testing

Final testing for our software was used to determine a multiplier for the signals such that the client can comfortably play the games. The goal for our accelerometer adaptation is to simply boost the signal received, so that the client must still develop skill in order to create the proper swing. We are not trying to map every movement to a valid swing, we are simply putting more power behind the swing. To ensure that this is correct, we tested a variety of swings on the new accelerometer. Since we are familiar with the different types of swings associated with the games, we can compare our new results to the one’s we have achieved when playing with the hand. The game should be intuitive to play, the only difference will be the power necessary to achieve a swing. Once we have done this, we allowed the client to play the game and make sure that their power allows for an adequate range of swings.

D. Final Testing

When each component has proved to work to our standards, we tested the product as a whole. First, we tested that the dimensions and weight of our product fell within the range we decided upon in our engineering requirements. This included weighing the hat component of the device and ensuring that it was less than one pound, and measuring the external circuitry on top of the hat and making sure that no dimension extended farther than three inches off of the head. We then placed it on the client’s head and made sure that it did not produce any unnecessary strain, and was generally comfortable to wear and the client was able to move their head freely. To test its durability, we made sure the headgear stayed snugly in place during all game movements. We had the client swing his head 10 times, and made sure that it stayed in the same position on the head. To test our intuitive design, we set up the device on a new user and verify that he could learn how to play in under 10 minutes. We also wanted to guarantee an adequate workout for the client using our system. This involved having him play the two games, and verifying that he swung his head at least 10 times per game. Finally, we made sure that the device was fun for the user. We queried the user for any feedback, and made adjustments accordingly.

V. Results

Our results proved our system is effective in playing Wii Tennis and Wii Bowling with only the use of the head. The results of our component testing showed that the buttons and software produce the desired outcome, but the IR sensor does not. Final testing showed that the IR is not necessary for game play or menu navigation, and that even without its presence, the system still behaves according to the requirements.

A. Button Results

When testing our buttons, we ran into a few problems. After soldering all the leads, we noticed that one of the buttons seemed to be short-circuited. We resoldered that button and tested again. We got it working shortly thereafter. After that, we ran into a few other problems with buttons not working or leads getting short-circuited again. We carefully examined the 8mm jacks and noticed that some of our wires extended too far and in certain angles would short circuit. We fixed this by shortening the exposed wire that caused the short and this fixed the issue. We used staking compound on our remote after we were satisfied with its behavior to ensure that none of our solder joints would break off or short again in the future.

B. IR Results

We successfully got the IR removed from the remote, soldered onto a breakout board, and connected back to the board through long wires. When we turned on the remote to test how it worked, the IR did not register on the screen. When we ran continuity tests, each of the IR leads was connected to its counterpart on the remote. Two leads, however, appeared to be short-circuited to every other lead on the IR. We thought this could mean that these two leads were common grounds, or that they could be short-circuited somehow within the IR. To test this theory, we ran the same continuity tests on our control remote. We could not replicate the results we got on our test remote, and we could not figure out where the short-circuits were. After examining the solder joints under a microscope, we concluded that no joints had been soldered together and shorted. This meant that the problem was somewhere within the IR, and the problem must have arisen during the desoldering process. In the end, we decided to leave the IR out of the final design since it was not necessary for game play. We included the buttons for the D-pad to ensure that the clients could navigate the menus.

C. Software Results

Our software algorithm required a lot of fine tuning based on client preference. All of our results are positive and the client can intuitively play each game. There were some problems encountered along the way. In our original boosting algorithm, where we did not sum the magnitude of each of the axes to compare against the dead zone, the game would recognize certain resting positions as repetitive strong swings. We had to do some careful adjustments to ensure that resting positions were not read as motion. The system provides a wide range of swings based on the power of each swing, and the most
powerful swing does not require extensive head movement that could cause injury. The software requires some practice to perfect the motions that the console is expecting, but this is true of any Wii game, even when using the hand.

D. Final Results

Our final testing validated the rest of our engineering requirements. The system is simple and easy to learn, lightweight and compact, does not extend off the head too much, and provides an extensive workout routine. Our client can play easily, and is very happy with the final product.

VI. CONCLUSION

Our product did not end up the way we had originally intended, but it still meets the specifications and goals we set out at the beginning of the project. We have, to the best of our abilities, mimicked the Wii remote’s functionality and provided an adequate system for a person with quadriplegia to play. We have provided the utilities for a universal design, and left our design open for future groups to make further implementations.

The IR is intact in our final remote, so future groups can try to remove it and attach it to the hat like we had originally intended. Further improvements would be to make the head mount device wireless so that head movement is not limited by the long wires extending down to the remote. The box could also be made wireless so that a power cord does not have to be attached to the wall. We have made our software so that it sensitive enough to be played by someone who has severe disabilities, and anyone with more strength as well. However, a nice feature would be to allow a sensitivity adjustment to be tuned to each individual client. Our client in particular has a strong neck, so he could lower the sensitivity to give himself more of a challenge. Any client with less strength could raise the sensitivity as needed to play comfortably.

These adjustments target some additions that make it more sleek, but do not affect the underlying functionality of the system. We have the core functionality set up so that it meets all of our requirements effectively. We have packaged our system so that it cannot get altered during game play and so that it looks and feels complete and simple, regardless of the underlying functionality.

VII. ACKNOWLEDGMENT

Thanks to the Central Coast Assistive Technology Center (CCATC) for providing adaptive sensors for loan, our EE technician Jaime for removing one of the IR sensors from the Wii Remote, Kevin Taylor and the kinesiology department for the motivation of the project, and Mike Ward for user input and testing.

REFERENCES