Mobile Healthcare & Machine Learning Final Report : Gravity Rush

Sophia AHMED-ASHFORD, Ryan ARMIGER, Ifeanyi CHINWEZE,
Rebecca ELKINGTON, Jack PEARSON, Omar WOLFE HUSSEIN

Abstract—We present a mobile app solution for the rehabilitation of muscles following injury - Gravity Rush. Gravity Rush is an arcade style game controlled by two surface electromyography (sEMG) electrodes placed on the target muscle, and connected to a mobile device via Bluetooth providing input data for a convolutional neural network that classifies the data into two different actions. These actions are used to control the game; the aim of which is to navigate a rocket through space, avoiding asteroids and black holes. The game connects to a web server where each user's high score is stored so it can be displayed on a leaderboard. Players can also build a friends list from other user's to enhance competition. The impact of gamification on the willingness of patients to use the therapies was investigated by comparing data collected on the Gravity Rush game against a simple control app which just tracks muscle movements. It was found that the use of gamification did cause a statistically significant increase in patient willingness, although there are limitations to these findings.

Fig. 1: Gravity Rush : Menu screen.

1 Introduction

Many people suffer from limb motor problems as a result of both neurological conditions such as cerebral palsy, or physical injuries like broken bones and ligament injuries. Current long term rehabilitation treatments have been deemed inadequate [1] due to a lack of resources to properly sustain long term treatments. Consequently, patients are not given the appropriate length of treatment for their condition. For example, fewer than 30% of the stroke patients who need it receive a six-month treatment plan [2] for their rehabilitation.

A solution to this issue is the introduction of home-based therapies that patients can complete independently from their own homes. This reduces the stress on healthcare providers and allows patients to rehabilitate from the comfort of their own home. However, home-based therapies introduce the problem that patients may lose motivation to continue. Patients tend to stop their rehabilitation plans early given a lack of enjoyment and motivation to continue, given the small incremental improvements shown in long term plans. One solution to such a problem is the "gamification" of therapies, making them more enjoyable thus increasing the motivation for patients to complete the course of the rehabilitation.

This report documents a game, Gravity Rush, which is controlled by muscle actions and designed to be used for home-based muscle rehabilitation therapy. The system consists of two surface electromyography (sEMG) sensors connected to the patient's arm, which communicate with a game run on a connected iOS device, such as a tablet or mobile phone. The game is played by making specific hand gestures that control the game mechanics of moving a rocket through space. The hand gestures intended by the user are inferred in real time by the signal data received from the sEMG sensor array.

2 Hypotheses

Using the Gravity Rush platform we are able to explore the efficacy of sEMG based games on muscle rehabilitation as well as the the effect of using "gamification" on the motivation of user's to rehabilitate. By conducting user tests and surveys to gather data, we can evaluate the changes brought about by using a "gamified" version of the tool compared to a tool that doesn't contain game elements. Section Section 5 outlines the full experimental process. The hypothesis to be investigated is as follows:

H1 The introduction of a game-based, competitive, training tool increases the amount of time spent on rehabilitation exercises compared to a traditional, non-game, non-competitive tool.

3 Literature Survey of Similar Work

Lyons et. al. [3] presented a game based solution for muscle rehabilitation that was controlled by sEMG sensors. Their device was a large desktop unit that processed the sensor data which it then communicated to a home computer. Pre-existing Java games (such as Space Invaders) were modified so that they could be easily controlled by muscle activity. The games that were chosen were ones that only require single key presses in normal operation - and so in the sEMG context, only require single muscle contractions. When tested with a group of adults with cerebal palsy, muscle strength increased by 1 point on a scale from 0-5 and there was a 125% increase in range of motion in the users compared to a control group. Gravity Rush follows similar underlying concepts however our implementation differs, as detailed in Section 4.

4 System Overview

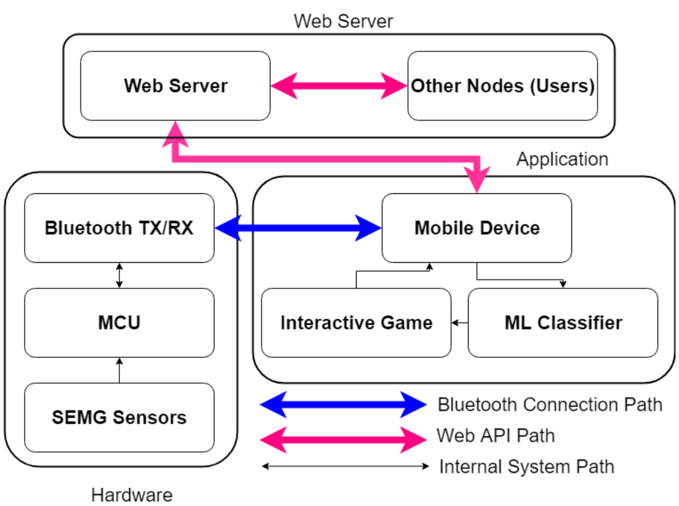


Fig. 2: System architecture diagram.

The system is comprised of three main subsystems, detailed in Figure 2. The hardware system contains sEMG sensors which are read and processed by a microcontroller (MCU). This is then connected to the mobile application via a Bluetooth connection. The user plays the game on the mobile application using both the sEMG sensors as the controller and the phone's touch screen. From the sensors (which are connected to the area around their elbow) the player can steer the rocket ship left and right by curling their wrist and clenching their fist respectively. The user can then make the rocket ship speed up by tapping the screen and slow down by swiping downwards.

For the sensor actions, the game interprets which move the player is intending to make using an ML algorithm which classifies each sensor reading into either 'none' (to indicate the player is not trying to make a move), 'left', 'right' or 'clench'.

Game data is then stored to firebase cloud web server which is used to create a leaderboard feature and allow users to add each other as friends.

4.1 Hardware Sensing Platform

4.1.1 Hardware Design

The hardware sensor uses two sEMG electrodes to measure local bio-potentials on the target muscles. The sensor units used are the MyoWare Muscle Sensors [4] which provide an analogue voltage correlating to the muscle activity, Figure 4. This voltage has been rectified and amplified by an adjustable gain in order to deal with noise. The signal is then read by an ATmega328p [5] via an Analogue-to-Digital Converter (ADC). The ATmega processor runs the Arduino bootloader [6] to simplify the development process. Communication between the hardware platform and the phone is achieved using Bluetooth Low Energy (BLE) via the BlueFruit LE UART Friend, a BLE module that communicates with the ATmega using the Universal Asynchronous Receiver-Transmitter (UART) protocol.

A platform prototype was developed using an off-the-shelf Arduino Uno board to prove the functionality of the system and then a custom Printed Circuit Board (PCB) was created to reduce the form-factor of the final system, Figure 3. Unfortunately, due to time restrictions, the custom PCB was not used in the experiments.

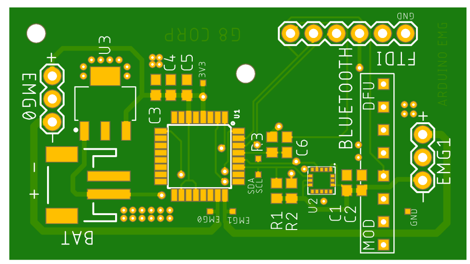


Fig. 3: Rendering of the custom Printed Circuit Board designed for the hardware sensing platform.

4.1.2 Firmware

The Arduino runs the Gravity Rush firmware that takes readings from the sensors, packages the readings and then transmits them to the BlueFruit module that encodes the messages into Bluetooth messages and broadcasts them to the phone. The Arduino platform was chosen as it allowed for faster development times, however some limitations of the 8-bit AVR chip were found. The most significant issue resulted from a lack of Static RAM (SRAM). The ATmega328p has 2k bytes of SRAM, enough to hold 500 integer values in memory. The initial plan was for the hardware platform to collect a time series of readings from the sensors and to send these values in batches every 100 ms. However, there is not enough SRAM to hold a series that long so instead the readings were processed and transmitted as they were made.

The Bluefruit module was configured as a Human Interface Device (HID), effectively turning it into a Bluetooth keyboard for the phone. This meant that the messages were encoded into Unicode before being transmitted, which was then decoded by the phone.

4.2 Machine Learning Classifier

4.2.1 Anatomy

In our two sensor setup, one set of sEMG pads is placed on the inside of the forearm near the elbow and the other on the outside, across the muscles associated with finger movements, again near the elbow as shown in Figure 4.

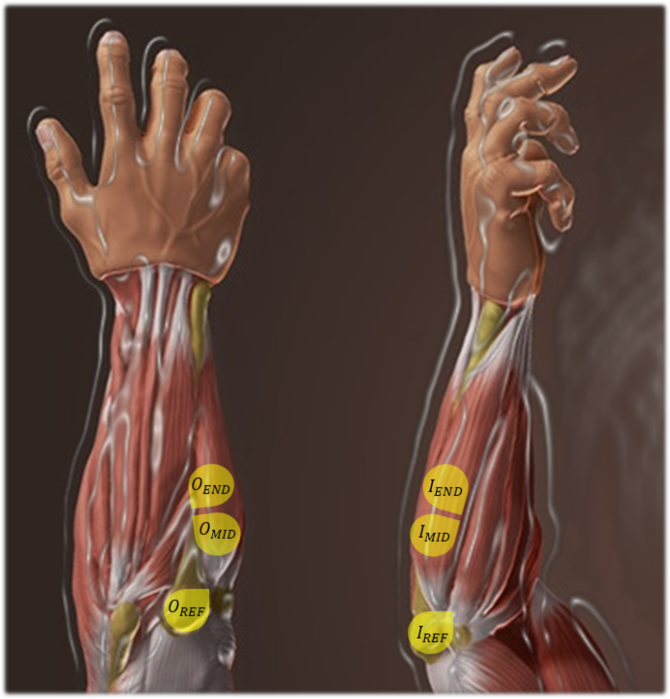
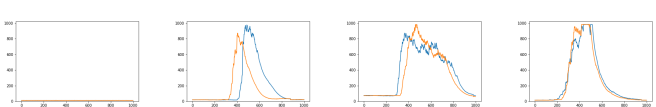


Fig. 4: Chosen positions of two sEMG sensors on the the inside (I) and outside (O) of the same right forearm.

Both sEMG sensors require three electrode placements - END, MID and REF as highlighted in Figure 4. The sEMG signal is most clear when the REF electrode is placed near a bone, thus the placement of electrodes I_{REF} and O_{REF}. In the simplest example, our classifier was trained using data collected from specific actions:

1) `0b00` None - No Movement.
2) `0b01` In - Curling the wrist towards the user.
3) `0b10` Out - Curing the wrist away from the user.
4) `0b11` Clench - A firm clench of the hand.

These four actions were chosen to make best sense of the two chosen sEMG sensor positions. Action 2) strongest activated sensor I (Figure 5b), while Action 3) strongest activated sensor O (Figure 5c) and Action 4) strongly activated both sensors (Figure 5d). A classification of *None* was necessary to give a steady state for no action taken place, providing almost a calibration of the sensors when training the model, as seen in Figure 5a.



(a) None. (b) In. (c) Out. (d) Clench.

Fig. 5: Example waveforms of sEMG sensors $I : blue$ and $O : orange$ for the each possible classification.

The resulting classifications can be imagined primitively as a two bit number with sensors I and O corresponding to an On/Off classification on bits 0 and 1 respectively. This simple idea was originally tested as a threshold classification problem, calibrating the threshold manually to ensure a clear, distinguished reading between the actions before proceeding with a more complex model. Using only thresholding for classification does reduce computational complexity and work to a good standard for a two sensor input stream, however it does not extrapolate well to the greater resolution acquired from more sensors.

4.2.2 Convolutional Neural Networks

The machine learning (ML) model chosen for the final design is a convolutional neural network (CNN). The filtered outputs from layers of a CNN are capable of extracting local features and patterns from input data and thus, CNNs are used heavily in the classification of continuous signals. The specific model chosen will extrapolate to any amount of sEMG sensors, up to a bound set by the processing capabilities of the mobile device chosen. The architecture used in our application is shown in Table 1.

Layer Type	Dimensions
Input	$(1, N, f)$
Convolution 2D	$(1, N, 1024)$
Activation ReLU	-
Batch Norm	-
Pooling	$(1, 1, 1024)$
Dropout (50%)	-
Convolution 2D	$(1, 1, 512)$
Activation ReLU	-
Batch Norm	-
Pooling	-
Dropout (60%)	-
Flatten	(512)
Dense	(128)
Activation ReLU	-
Dropout (40%)	-
Dense	(4)
SoftMax	(1)

TABLE 1: Model architecture.
$N : Number\ of\ sEMG\ sensors$
f : $Sampling\ frequency\ (Hz)$

The CNN used in our application consists of two-dimensional convolution layers. The one-dimensional equivalent was available, but extracted local patterns from the multi-channel data stream less successfully. The 2D option was better suited to handling the input sEMG stream described in Subsection 4.2.1. Training data for our CNN was collected from a member of the group by attaching the sEMG sensors, in the configuration described, to their right forearm. The subject was instructed to perform one of Actions 1-4 in response to a visual stimulus (flashing LED). The response was recorded at a sample rate of 1KHz over one second intervals and sent to a Python script running on a PC. The measurements were formatted and written to `csv` files in the form that they would be used on the mobile device.

There were ≈ 850 samples taken, with an equal split across the four classifications, to be used for training and validation. Using ≈ 750 samples for the training set, the model was trained with early stopping enabled, reaching roughly 25 epochs. Early stopping was used due to the size of the training dataset and to avoid over-fitting the model. At the end of the training process, the model summarised in Table 1 had a validation accuracy of 83.4%, which was deemed sufficient given time and resource constraints. This accuracy, is improved upon using methods discussed in Subsection 4.2.3.

Other groups who have performed similar sEMG-based experiments have built databases with more than $15,000$ samples [3], [7], [8]. Comparatively, the ≈ 850 samples our group collected is very low and limits the ability of the model to generalise its classification approaches to a larger population. Additionally, these groups commonly record sEMG signals with 8-12 electrodes [9] to generate inputs to a classification model with higher spatial resolution. With more detail, classification models become accurate and can reach a validation accuracy in excess of 90% [9], [10].

4.2.3 Mobile Processing

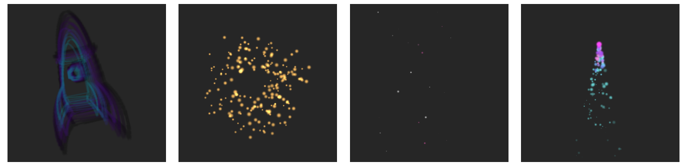
For our prototype we use the *iPhone 6* which accommodates two sensor inputs. The data received from the Bluetooth connection are `utf-8` encoded bytes which are comma and line ending separated. Once the packets are processed into an integer format and packed into a FIFO buffer, we then proceeded to classify the action after each new packet. Originally we chose to use the full sampling rate of 1KHz for classification rate. This was too computationally heavy for the phone despite having dedicated graphics processing. The constant running of the model prediction slowed the game to the point of freezing and sometimes crashing. Progressively lower sampling rates were tested eventually settling for $40Hz$, which despite initial thoughts actually had a better classification accuracy than the 1KHz model of 86.9%. The models were simply retrained on the same data, varying sampling rate by integer factors of the original 1KHz so to not introduce skew to the sample timing.

Using the lower sample rate of $40Hz$, and following the previous method of classifying one second data snippets on each new received sample was computationally acceptable for the *iPhone 6*. The processing of game mechanics and display were no longer hindered however, due to the 86.9% accuracy of the classifier, misclassification while playing the game was very common and made the game very difficult to control. To solve this issue the resulting classifications from each new data input were saved in a separate array and the most common classification over that period is taken as the current action in the game. This smooths out the classification error and works well for even lower classification accuracies which could occur quite easily depending on the ratio of actions classified to number of sEMG sensors.

In practice the classification of Out was regularly misclassified as In. At a first glance of Figure 5b and 5c, there is a clear difference between the peak magnitudes of the separate sEMG sensors. One issue with the use of the convolutional network is the convolution of the two graphs will still come out as fairly similar. This paired with inconsistencies found throughout the Out class dataset may explain this issue. For the final design we chose to use only the three working classifications, $None$, In and $Clench$.

4.3 iOS Game

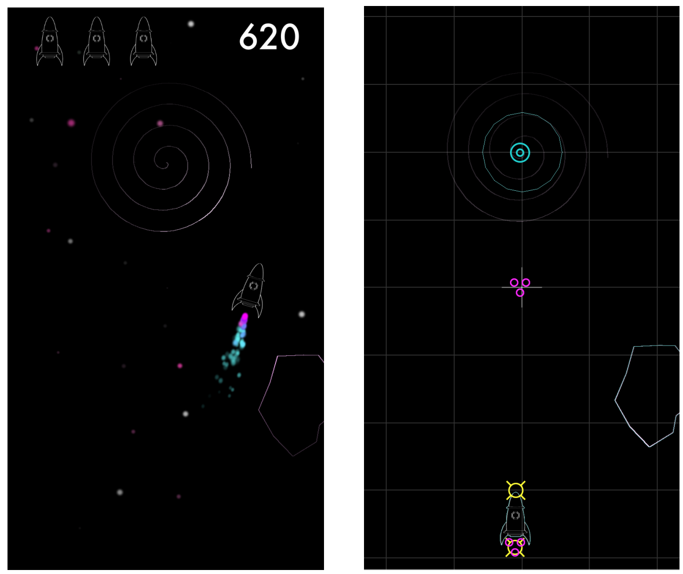
The game design takes inspiration from the classic arcade game asteroids. Figure 7a shows the elements a player encounters during gameplay. Like in asteroids, the player will be able to rotate the spaceship left or right, and move it forwards. Additionally, there will be randomly generated asteroids which decrement the life counter should the ship collide with them. Unlike the asteroids game, the aim is to move up the screen for as long as possible, increasing your score. Additionally, there are also black holes which suck the spaceship towards them if it gets too close and



(a) Black hole death animation. (b) Asteroid collision explosion. (c) Background star/space effects. (d) Spaceship engine thrusters.

Fig. 6: Emitters used in the iOS game.

if the player is caught in the middle of one, it will count as an instant death and will move straight to the game over screen. As the player progresses, the difficulty increases, as the width of the path the player has tor travel in is decreased, by increasing the size or frequency of the obstacles.



(a) Image of player perspective. (b) Image of level design screen.

Fig. 7: Comparison of development and player perspective.

The game mechanics were chosen based on a few key metrics, primarily accessibility and implementation time. To appeal to the widest audience possible the game must be accessible and appealing. Simplicity improves the ability for users of different ages to understand and engage with it. Many of the mechanics such as the player controls have been common in games for decades. A game which lends itself to a simple scoring mechanism also allows different users to compare their progress with greater ease, inciting a level of competition that encourages use of the app.

The iOS framework SpriteKit was used to create the game due to experience using native iOS frameworks over other tools such as Unity. Because SpriteKit integrates well with other iOS frameworks, it allows simpler integration with other frameworks such as Core ML, which is used to convert and run the machine learning model that classifies actions. SpriteKit is also optimised heavily for iOS devices, the final game running at a consistent 60 frames per second on an iPhone 5s, a 6-year-old phone.

The goal of the game is to progress up the screen and avoid obstacles. The camera or perspective of the scene was bound to the players vertical position and the space the user can traverse horizontally is limited to the screen width. This provides a simple format that is familiar to users of other mobile games. The final view seen by the user is visible next to a development screen in Figure 7.

A key part of the game design is the use of the physics engine provided by SpriteKit. This allows the use of radial gravity fields surrounding the black holes, visible as blue circles in Figure 7b, as an interesting game mechanic. It is also used to define the controls for the spaceship. To increase the velocity of the spaceship a user taps on the screen, which applies force to the spaceship in the direction it is facing. Increasingly applying force increases the velocity further, and changing the direction the spaceship is facing and then tapping applies force in that direction. To control the direction of the spaceship, the action classifier provides a callback to a function that rotates the direction the ship is facing.

As the player progresses they must avoid obstacles in the form of asteroids and black holes. Collision detection is applied to these and triggers decrementing lives or ending the game, respectively. Collision detection is applied to an alpha mask of the objects within the game, ensuring it is pixel perfect and completely fair for a player. The assets used to display the asteroids and black holes were randomly generated in python.

Particle emitters have been used to enhance the style of the game, such as those shown in the background stars, spaceships engine flame and explosions upon collision with an asteroid. These are shown in Figure 6, and as pink circles in Figure 7b.

Dynamic lighting has been added to the spaceship, a white light to the front and purple light at the rear, shown in yellow on Figure 7b. Obstacles not within the range of this light are not visible, as visible in figure Figure 7a, providing an extra level of atmosphere and challenge to the game.

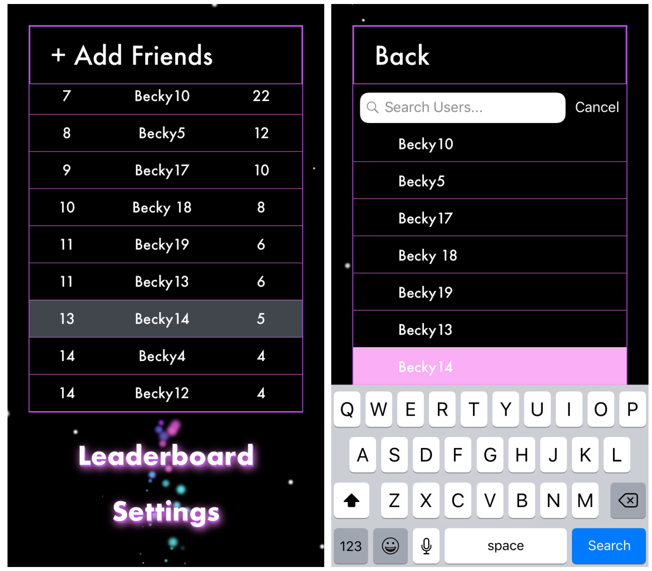
4.4 Web Server and Leaderboard

The benefits of gamification to engagement largely come from competition, and competition against other players can be just as useful, if not more so, than competition against just your personal best. To this end, all user's usernames and high scores are stored on a firebase web server, and a sub-set of these are displayed in a leaderboard.

Firebase was chosen as the web server because creating one from scratch would be unnecessarily time consuming for our purposes and give us less time to develop the concepts relevant to the experiment. Furthermore, it was chosen over similar purpose-built solutions such as amazon AWS and infinityfree because of its user interface which allows for easy database visualisation and thus is ideal for prototyping and testing. This being said, it would not be a good solution for a commercial or widely used app because there are limits to how much free storage it provides for the purposes of this project it was the ideal solution.

Every time a new high score is achieved in the app, the highscore field in the database for the user with the value of username that corresponds to the username stored in the apps 'UserDefaults' is updated.

The up-to-date leader board (Figure 8a) is displayed at the end of each game and can be accessed at any point via the app's homepage. If the user is in the top 5 of all users then the top 20 user's score are displayed in the leaderboard table. If they are outside the top 5 then the top 5 users, the 5 above them, the user themselves, and the 5 below them are displayed with rows of '...' to indicate where there are breaks in the rankings. It was decided not to just display all players because it could be difficult for the user to find themselves and they could become disheartened seeing themselves visually so far down.



(a) Leaderboard. (b) Add Friends Screen.

Fig. 8: Leaderboard and Add Friends Menu.

To enhance the competitive element of the app design, users are also able to add friends who either they may know from in person rehabilitation classes or may have a personal vendetta to beat a given user's score. To do this they can click the add friends button which will repopulate the table with a list of all usernames stored on the server (Figure 8b). They can then use a search bar to search for a username and add one by clicking on the row in the table. The names of added friends are shown in a different colour and can be un-added just by clicking on them again, at which point they will revert to the standard colour. The list of added friends is stored locally in the app's UserDefaults and all friends are shown on the leaderboard with '...' rows to indicate if there are gaps between users.

5 Experimental Procedure

Fig. 9: Control application for user experiments.

In order to investigate the hypotheses, a control application (Figure 9) was created which allows the user to practice their muscle movements without gamification. This control app features a left and a squeeze counter, which increment as the corresponding gesture is made. Effort was put into styling the user interface of this control app so as to not make the test subjects think we wanted them to prefer one over the other.

5.1 Initial Experimental Methodology

In the original experimental methodology, a test subject uses both the control and the game applications sequentially and their responses to each are measured and compared. The test user is informed the purpose of the study is to help determine the best technology assisted method for at home technology and that they will be asked to use and rate two mobile apps. They are then connected up to the sensors and given the control app to play with: they are instructed how to use it but not which exercises to do or when to stop. The same then happens for the game app.

Whilst this is happening, the amount of time the test subject spends on each game is measured as a way of determining how much more engagement and interest gamification provides.

After this portion of the test the user is presented with a Google survey that assesses their experience. The survey response is anonymous and filled in without the test supervisor looking at it, to make the test subject more likely to be honest with their answers.

The survey questions below were asked about each app and users could select their answer from: Not at all, Not very, Somewhat, Quite and Very.

1) How would this app affect how many exercises you do in a session?
2) How would this app affect how often you do exercises?
3) How useful do you feel the app would be for rehabilitation?
4) How intuitive is the app?
5) How fun is the app?

It then asks the user to rate the following, selecting their answer from one of: Very poor, poor, Satisfactory, Good and Excellent.

1) The game mechanics.
2) The game leaderboard features.
3) The game concept.
4) The exercise detection system.

Finally it asks the user if they would consider using the following methods to rehabilitate a muscle, selecting their answer from: Not at all, Maybe and Definitely.

1) Muscle controlled game.
2) Muscle tracking app (i.e. the control application).
3) Tracking by hand/ on paper.
4) Not tracking at all.

5.2 Secondary Experimental Methodology

Unfortunately, the machine learning algorithm was trained on a very limited data set from only one participant and thus, for many others users the model struggled to correctly classify their gestures. This led to the game becoming unresponsive for them in testing, causing them to become frustrated with the game in a way that is not representative.

Few tests were conducted in the way described above due to the performance of the ML model. As such, a second experiment was derived that does not require the test subject to actually use the device. While this might give less reliable results, it does allow more responses to be counted since we are no longer limited by the time it takes to conduct tests and to find participants who can be physically present.

This second method of testing features a 2-minute video explaining the aim of the project and demonstrating both applications in operation. After having watched the video (available here [11]), the test subject is asked to fill in a short survey.

This survey asks them to score both apps out of 5 (1 being lowest and 5 being highest) against the following criteria:

1) How likely would you be to use it for at-home muscle rehabilitation?
2) How enjoyable do you feel the app would be to use?
3) How would you rate the app's design?

There is also a field for additional comments and thoughts to help inform our future work plan.

5.3 Experimental Population Sample

The core of the population sample is based around students of Imperial College London, as this is where the development of Gravity Rush took place. For the Initial Experimental Methodology there were 12 participants, 6 were able to control the game themselves using sEMG, whilst the remaining 6 were shown a demonstration of the game using muscle control and then used alternative control to play themselves. Of these 12 all were aged $18 - 25$ and there were 7 males and 5 females.

For the Secondary Experimental Methodology there were 40 participants, the form was distributed online (via Social Media) in an attempt to expand the population sample. Due to data protection regulations it was decided the form would be anonymous and so the exact demographics of the sample are unknown.

6 Results

6.1 Initial Experimental Methodology

If the sensors did not work on a test subject, they became frustrated and responded negatively. To resolve this problem, halfway through the in-person tests the methodology was changed to simply have them observe the apps in use, and play them using touch screen controls instead of the sensors. Other than this, the same quantities were measured, and the participants were asked to fill out the same survey at the end.

The measured data for the participants who were connected to the sensors is shown in Table 2.

Participant No.	Sensor Performance	App 1 Time	App 2 Time	Time Difference
1	Good	0:45	4:25	3:40
2	Poor	0:55	3:15	2:20
3	Fair	2:05	4:25	2:20
4	Poor	1:05	1:30	0:25
5	Good	2:30	4:52	2:22
6	Poor	2:56	3:24	0:28

TABLE 2: Measurements from User Testing.

Every single test subject spent more time playing the game than using the control app, which lends weight to the hypothesis that gamification does increase engagement. On average, participants spent 1:56 more minutes on the game than on the control app and the two for whom the sensor performance was good averaged 3:01 minutes longer.

These 6 participants and a further 6 who were just shown correct operation and then asked to play around with the apps (as described above) then also filled in the survey described in Subsection 5.1.

The qualitative scale of responses (i.e. 'not at all' to 'very') was converted to a quantitative scale for better analysis by equating 'not at all' to 1, 'not very' to 2 and so on. The same was done for all qualitative feedback scales in the survey. Table 3 shows the average scoring of each game across the various questions asked.

Question	App 1	App 2
How would this app affect how many exercises you do in a session?	3.9	4.0
How would this app affect how often you do exercises?	4.0	4.1
How useful do you feel the app would be for rehabilitation?	3.8	4.0
How intuitive is the app?	3.5	3.6
How fun is the app?	3.3	3.5

TABLE 3: Survey Responses from User Testing.

In every criteria the game scored higher than the control app, although not by a large amount. It is worth noting that this data set is likely biased against the game as it includes the data points from participants for whom the sensors worked poorly. Based on notes taken during the user testing process these users found the lack of responsiveness more frustrating when using the game than the control app, although it should be noted that this is not a drawback of the game since the investigation into the concept can reasonably assume fully working technology.

A paired, one-tailed t-test was conducted on the first two questions to assess the statistical significance of the findings.

The p-value for the first question was $p = 0.361$ which means that assuming the null hypothesis (i.e. that gamification will not make the user want to do more exercises per session) there is a 36% chance of getting these results. This suggests that the results are not statistically significant and our dataset does not lend sufficient evidence to confirm the hypothesis.

For the second question the p-value was also $p = 0.361$ which similarly means the dataset does not provide enough evidence against the null hypothesis. For both of the tests done, increasing the number of people sampled could well yield a more statistically significant result.

Participants were also asked to rate, among other things, the game mechanics, the game concept and the leaderboard features, giving them average scores of 3.6, 4.1 and 3.6 respectively. For context on the qualitative scale that participants were using, a score of 3 corresponds to satisfactory, and 4 corresponds to good, suggesting a broadly positive response to these criteria.

Finally, the participants were asked what solutions they would 'Definitely', 'Maybe' and 'Not at all' use, should they have to rehabilitate a muscle, the results for which are shown in Table 4.

Solution	Definitely	Maybe	Not at all
App 2 (Game)	9	3	0
App 1 (Control)	6	6	0
Tracking by hand/paper	2	4	6
Not tracking at all	1	5	6

TABLE 4: Additional Survey Responses from User Testing.

The results suggest that Gravity Rush was the most popular option amongst the testers as more people were confident in its use as a rehabilitation solution. This is a good indication of successful concept.

6.2 Secondary Experimental Methodology

The second method is the video and survey described in Section 5. This was shared using various social media platforms and in the end received 40 responses. The results of the survey are shown in Table 5.

Question	App 1	App 2
How likely would you be to use it for at-home muscle rehabilitation?	3	3.9
How enjoyable do you feel the app would be to use?	2.65	3.95
How would you rate the app's design?	3.2	3.9

TABLE 5: Responses from the Online Survey.

It can be seen from these results that participants scored the app higher than the control app, and to a larger degree than for the in-person tests since the bias from sensor difficulties has been removed.

The same t-test as for the first method was conducted on the first survey question, which yielded a p-value of 0.00000387. This means that the result is statistically significant at the 5% (and indeed the 1%) level since there is about a 0.0004% chance of getting these results if the null hypothesis (that gamification doesn't make you more likely to use the app) is true. The fact that people are more likely to use the app confirms the overarching hypothesis laid out in Section 2, that people will spend more time using the app.

The other questions also yield interesting and useful results, by demonstrating that participants overwhelmingly felt the game was more enjoyable and had a nicer design than the control. This suggests that it is a quality application and that the concept for the game is a good one that achieves its aim of being fun to play.

There was also a section for miscellaneous comments which was filled out by 12 people. Two of the comments felt that it was confusing and prohibitive to patients with certain conditions to have the game controlled both with the sensors and the touch screen, and suggested that all movements should be made using the muscle. This is a feature that could certainly be implemented, but would require more sensors to get reasonably accurate classification. Another interesting suggestion that was made was actually to incorporate the control app into the game by having a report of the number of movements made available. The majority of the comments were positive feedback about the game and negative feedback about the control app, which both emphasises the result and suggests that we were successful in making the control app seem like a legitimate application that was being worked on and investigated with equal weight to the game.

6.2.1 Limitations

While this is a statistically significant result it is also one with many limitations. Firstly, since they did not actually play the game, participants may be falsely reporting on how engaging it is. It is plausible that upon actually testing they may find it more dull and frustrating or more engaging and usable than they would imagine from just watching a video of it.

Secondly, the fact that this survey was only spread via social media and the social circles of the group are made up of mainly young people means that realistically the results only hold true for a segment of the population. It is possible that for more elderly people they would not appreciate the added technical complexity of the game and this may actually make them less likely to perform the rehabilitation exercises. While this may not seem like a problem since all the existence of the app would do is allow patients a greater choice there is a danger in situations such as this where there are limited resources, that a treatment that works well for a lot of people is pushed to be the standard for everybody. This nuance in needs is something that currently stretched healthcare services may not be able to account for, so a proper conclusion should reflect the thoughts of a more fairly distributed population sample.

7 Future Work

7.1 Hardware Sensing Platform

Future development of the hardware platform would allow for the issues found with the current implementation to be solved. Primarily, extra time would allow for the custom PCB to be assembled and so the hardware elements would benefit from a reduced form factor. Additionally, given further development opportunities, the processor would likely be changed to higher performing device such as an ARM Cortex-M0 which would reduce the limitations on SRAM availability along with increasing processing power. This increase in memory and processing capability would allow for the data to be split into chunks at the hardware end and the data could be transmitted in batches rather than in the streaming method currently used, reducing overheads on the mobile device.

7.2 Machine Learning Classifier

As discussed in Subsection 4.2.3, the current solution only facilitates two in-game controls. The addition of more sensors would provide greater resolution to the convolutional classifier model and could potentially achieve classification of specific finger movements in the case of wrist and forearm exercises that this project has focused on. Given the integration of more sensors, the inputs can be trivially added to the ML model (discussed in Subsection 4.2.2) and the game could be entirely controlled by the users muscle movements, as opposed to the current solution of using the free hand to accelerate and decelerate. Additionally, the model would need to be trained with better data. This would be achieved by collecting thousands of gesture recordings from a wider pool of subjects. The overall impact on the model should be a boost in the performance and robustness between users.

7.3 iOS Game

Currently, levels are designed and placed by hand, however in future they could be procedurally generated, producing infinite varied levels of increasing difficulty. Other game mechanics could be introduced such as enemies or items to collect. To increase the accessibility of the system, ideally the app would eventually be ported to android as well.

7.4 Big Data

In order to accommodate multiple muscle injury rehabilitation requirements the iOS platform and the integrated server would safely store data on specific muscle movements in order to train the classifier to better distinguish between various different sEMG configurations on different parts of the body. This implementation of big data storage would also provide a basis for a separate application to be used by physiotherapist to track their patients progress. A different web server solution would have to be used to host this much data. There are many dedicated 'big data' servers with better pricing plans and infrastructure for high data volumes, so one of these would have to be selected.

8 Conclusion

Overall, Gravity Rush has been shown to successfully demonstrate the feasibility and usefulness of using sEMG sensors in a mobile game context for muscle rehabilitation. The system developed has accomplished the goals of integrating sEMG sensors with a Machine Learning Classifier and a mobile device game along with the social aspects of a leaderboard and friends lists.

In-person user testing was carried out, and while the conclusions lent weight to our hypothesis the sample size was not large enough to constitute statistical significance. This portion of user testing also highlighted the importance of expanding the training data set (detailed in future work) before this could be marketed as a commercial product. A user survey was then created which confirmed that users would be more likely to use the game for at-home rehabilitation than a control application, which confirms the initial hypothesis.

Further development, in particular the addition of extra sensors should be continued to explore the beneficial impact of increased input detail for the classifier.

References

[1] W.-K. Song, K.-H. Cho, J. Y. Jeong, Y. Kim, J. Y. Kim, and J.-H. Shin, "Reaching contralateral target by chronic hemiparetic stroke survivors using active-assisted/active exercise with 2D/3D visual feedback," in *2015 IEEE International Conference on Rehabilitation Robotics (ICORR)*, pp. 913–918, IEEE, 8 2015.

[2] Martin Barrow, "The NHS is failing stroke patients and society - Raconteur," 2017.

[3] G. M. Lyons, P. Sharma, M. Baker, S. O'Malley, and A. Shanahan, "A computer game-based emg biofeedback system for muscle rehabilitation," in *Proceedings of the 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEEE Cat. No.03CH37439)*, vol. 2, pp. 1625–1628 Vol.2, Sep. 2003.

[4] Advancer Technologies LLC, "Myoware Muscle Sensor."

[5] Microchip Inc., "Microchip ATmega328p 8-bit AVR Microcontrollers."

[6] Arduino, "Arduino.cc Arduino - Bootloader."

[7] M. Atzori, A. Gijsberts, C. Castellini, B. Caputo, A.-G. M. Hager, S. Elsig, G. Giatsidis, F. Bassetto, and H. Müller, "Electromyography data for non-invasive naturally-controlled robotic hand prostheses,"

[8] Atzori M, Gijsberts A, Castellini C, Caputo B, Mittaz Hager A, Elsig S, Giatsidis G, Bassetto F, Mller H, "Data from: Electromyography data for non-invasive naturally controlled robotic hand prostheses," 2014.

[9] W. Geng, Y. Du, W. Jin, W. Wei, Y. Hu, and J. Li, "Gesture recognition by instantaneous surface EMG images OPEN," *Nature Publishing Group*, 2016.

[10] M. Atzori, M. Cognolato, and H. Müller, "Deep Learning with Convolutional Neural Networks Applied to Electromyography Data: A Resource for the Classification of Movements for Prosthetic Hands," *Frontiers in Neurorobotics*, vol. 10, 9 2016.

[11] Rebecca Elkington, "Mobile Healthcare Project." `http://y2u.be/S2NJjpnO7Fc`, Mar 2019. [Online; accessed 19. Mar. 2019].