Technological Feasibility Analysis

October 12th, 2023

Team Name: Motus Methods

Sponsor: Dr. Zachary Lerner

biomotum

Mentor: Saisri Muttineni

Team Members

Caroline Fye

Payton Cox

Eli Barela

Tomas Jauregui

Table of Contents

Introduction

All parents look forward to the day their baby has their first steps; it is an amazing milestone and indicates a healthy brain. Around this age, however, some parents discover their child has a brain development disorder known as cerebral palsy. Cerebral palsy is the most common motor disability in children and a lifelong disability that incurs a financial burden of over $1 million per affected child. They will go to lengthy physical therapy sessions that, without seeing progress, become discouraging and hard to follow through with. This is why Motus Methods is creating a video game that connects to rehabilitation, making it a fun and worthwhile endeavor.

Currently, pediatric physical therapists help the children through:

- Physical therapy sessions can be long and straining on their mental and physical health
- Prescription of passive leg braces to increase mobility that can be uncomfortable and bulky to use.

While both of these are helpful, they often do not lead to long-term mobility. To children, physical therapy can be mentally and physically straining, ultimately creating an undesirable experience.

Biomotum is an NAU technology company co-founded by Zachary Lerner “with a long-term business objective to provide rehabilitation professionals and patients intelligent products that increase independence and mobility”. Biomotum already has robotic leg braces known as exoskeletons that connect to an app to assist in data retrieval. These legs also can be used for assistance and resistance training; however, they are mostly used in a lab setting for now. Biomotum is looking for a solution to eventually use these exoskeletons in a pediatric clinical environment.

Biomotum has partnered up with the Motus Methods NAU capstone team to create an iOS application that connects to these robotic legs with an engaging game that aids in walking rehabilitation training. As stated before, there is a functional, yet simple iOS app Biomotum has created that “controls the device (enter patient details, set up a session, calibrates the device,

sends torque control commands) and streams sensor data (number of steps, battery data, torque set points, etc.) to an AWS database".

Using this as a starting point, Motus Methods will create a product with the following features that will help provide solutions to the rudimentary aforementioned challenges:

- A game, based on the users' ideas that can create an enjoyable physical therapy experience.
- This game will also be able to track the training progress over time to see improvements or areas of struggle, which can be beneficial to pediatric physical therapists for these children.
- Motus Method's stretch goal is to also implement an AI/chatbot that is a virtual assistant to the children, that will be able to respond to the children's conversations accordingly; for example, if they want to talk about any struggles they have with their disability.

Biomotum and Motus Methods will be working diligently with an iOS application and game development technology in order to make these ideas come to fruition and help children with motor disabilities worldwide. However, not every project is able to be implemented smoothly without challenge. These challenges are outlined to examine different solutions to develop this game with confidence.

Technological Challenges

Game Engine:

For the game to run efficiently, there are certain game development tools that we need. First, we need a way to configure a gameplay loop and have things load before and after that loop. It will have to render assets and audio that correspond to certain elements inside the game. The assets that are in the game must have the ability to be animated so that users can relate to the character more. The final thing that is needed inside a game engine is a collision detection system so actors can interact with each other.

Application for exoskeleton:

We need a way to make sure that the data from the exoskeleton is being transferred to our game in a succinct manner. Included in those needs is a way to retain the data from a user so that they can go back to their game and keep their progress. Another piece of data that we need is session frequency and playtime in order to send back to their parents and physical therapist.

Technology Analysis: Game Engine

Currently, the main challenge we face is the search for a game engine that not only offers us an adequate set of tools for project creation in line with the client's specifications but also facilitates the integration of data from an exoskeleton. The selected game engine must excel in rendering assets, provide the flexibility to manipulate actors and player behaviors, and accurately record playtime and session frequency. The choice of the game engine will significantly impact the game's scale and performance.

Desired Characteristics

The essential characteristics we seek in a game engine include a comprehensive and robust library of functions to streamline the game creation process, efficient rendering of assets in the desired format, real-time manipulation of sprites and animations, and dynamic audio rendering. We also require an uncomplicated game structure that promotes asset and actor reusability while allowing for easy implementation of variations. Furthermore, the game engine should be optimized for mobile devices, offering a small footprint and swift performance. Importantly, it should enable swift and hassle-free data integration to ensure minimal user input delay and should seamlessly integrate with our existing application.

Possible Solutions

Our first option includes Unity and Godot. These game engines provide an extensive toolkit for game development, catering to both beginners and more advanced users. Unity, a veteran in the industry since 2005, allows independent developers to create games inexpensively while maintaining user-friendliness. Godot, created in 2014, has gained popularity due to its efficiency and similarity to Unity, particularly in terms of space efficiency.

Swift is another viable alternative, albeit primarily a coding language akin to C# and other object-oriented programming languages. Apple introduced Swift in the early 2000s as a replacement for Objective-C, and it is renowned for iOS app development. Although not a dedicated game engine, Swift can be tailored for game development using packages and modifications.

The final alternative under consideration is Flame, a game engine built on top of Flutter, which is programmed using Dart. Flame, introduced in 2019, caters to mobile game developers seeking to create games with Dart and Flutter. While Flame offers a range of game development tools, it falls short of the robustness of Unity and similar engines.

Analysis of Solutions

To assess Unity, Tomas Jauregui and Caroline Fye developed 3D games, leveraging their approximately two months of Unity experience. Unity's extensive library of functions and packages significantly aids 2D game development. It enables the rendering of assets, both in runtime and editor modes, and offers an efficient audio rendering system. However, Unity games tend to consume substantial storage space due to their visual scripting language and built-in editor, as well as their reliance on object-oriented programming, which can negatively impact runtime performance.

Regarding Swift, Tomas Jauregui installed Xcode on his Mac and experimented with a mobile game example. This example involved a color-changing pointer that followed the user's movements. In a separate attempt, Tomas explored building Wordle using Swift. The evaluation revealed that Swift lacks a comprehensive library and game development packages. It supports basic asset rendering but lacks seamless animation integration without complex functions. Audio capabilities are similarly limited. Swift's advantage lies in its minimal space requirements and simplicity, making it suitable for straightforward games. However, it appears that Swift does not provide the optimizations required for comprehensive game development.

Chosen Approach

The selected strategy is to utilize Flame and Flutter because of their seamless data integration capabilities between our Flutter app and the Flame game. Flutter proves to be a suitable choice for mobile game development when paired with the Flame game engine. It offers a less intricate alternative compared to Unity or Godot, yet still provides ample functionality for crafting a resilient game and facilitating future integration with the core Biomotum app.

Table A: Comparisons between game engines and their use for our project

	Render Audio	Render assets	Integrate data	Low space	Contains tools for game dev	Total
Godot/Unity	***	***		*	***	10
Swift	*	*	***		*	6
Flame	**	**	***	***	**	12

Table A presents a scale ranging from 1 to 3 to evaluate the essential characteristics required. Based on the table, Flutter emerges as the preferred choice, albeit by a slight margin, owing to its overall versatility. In contrast, Unity and Godot surpass Flame in rendering capabilities and toolsets. However, they are disadvantaged by their substantial resource demands and lack of straightforward real-time data integration for the exoskeleton.

Consequently, Flame is the selected game engine for the forthcoming project. While it may not boast the same extensive toolset as Unity and Godot, it excels in data accessibility, minimal space requirements, and provides the necessary assets rendering and audio capabilities, making it the most practical choice for streamlined game development.

Plans to Prove Feasibility

As we progress in the project, it's crucial to verify the functionality of the tools offered by Flame and confirm its compatibility with our existing Flutter app. To achieve this, we plan to develop a simplified version of the game Pong within a separate Flutter application. This approach will enable us to assess the impact of object-oriented programming on performance, evaluate how well Flutter manages game development in a macOS development environment, and confirm that the Flame game maintains a small footprint within the Biomotum app.

Technology Analysis: Application for exoskeleton

To guarantee that the input from the exoskeleton is effectively transmitted to the game, it's imperative to establish a seamless data flow from the exoskeleton into an application. This process involves gaining a foundational comprehension of the exoskeleton's operations, data collection, and interactions with mobile devices. An approach that prioritizes the reliable reception and utilization of the exoskeleton’s data is essential for the successful development of the game.

Desired Characteristics

To create a game that interfaces with a Bluetooth exoskeleton, there are essential characteristics that are needed by a mobile application. The exoskeletons that currently exist use an Arduino Nano 33 BLE, a microcontroller that uses the Bluetooth Low Energy protocol to broadcast walking data. As such, our gamified walking app must use the same Bluetooth protocol in order to utilize the data streamed from the exoskeleton for game interaction and progression. Another characteristic of the app will be to store different user profiles in order to allow returning users to “level up” in the game and track rehabilitation. The last requirement is to upload user data to AWS to allow physical therapists to review the walking data on different devices such as a desktop or cellphone.

Possible Solutions

An option we are faced with is to create a flutter application from the ground up that can utilize Bluetooth Low Energy. This would require the use of a flutter library that can access a device's Bluetooth chip. Libraries such as flutter_blue and flutter_ble_lib are popular choices and both offer many features that would make programming an app with Bluetooth capabilities very user-friendly. Once Bluetooth functionality is working, we would then need to implement user profiles, and gameplay through the use of our game engine, and display and upload the data to AWS.

There are many challenges this approach lends itself to. The primary challenge is the amount of overhead we would place upon ourselves as there would be many problems to solve before we would be able to get into developing the actual game. This leads to our other alternative which is using Biomotum's existing mobile app and extending the functionality to include gameplay based on a user's walking data. Payton Cox has been working with this app for over one year as part of his undergraduate research and is familiar with the ins and outs of this app.

Chosen Approach

Because of the overhead required to make a brand-new app, we are choosing to expand on Biomotum's app. There will be a lot of work involved in creating a game and a training profile, so by using an app that can already connect and stream data from the exoskeletons as well as connect to AWS we can divert more of our time to delivering and polishing the game.

However, with this approach, there is a roadblock that needs to be addressed. Biomotum's app uses Flutter version 1.22.6 which is a version of Flutter that does not support Null Safety. Recent versions of Flutter use Null Safety in order to prevent errors from interacting with Objects that have not been initialized. Since Flutter has used Null Safety for two years, many libraries do not support operations without Null Safety. In order to circumvent this issue, we must be careful to pick libraries that are compatible with Flutter 1.22.6.

Plans to Prove Feasibility

To prove the feasibility of our chosen approach, we must ensure that the version of flame that we choose allows for non-null Safety implementation. As we develop our Pong game, we will verify that we are on the same Flutter version as Biomotum's app as well as a compatible flame version. Once we verify we can develop a game in this version of Flutter, we can incorporate the demo into Biomotum's application to ensure the current app can function properly with a game mode inside of it.

Technology Integration

There are many challenges that Motus Methods face to accomplish this task. There exist many programming languages, frameworks, IDEs, and game engines that can be used. As we have discussed the specific technological challenges and chosen approach, we must now discuss how our team plans to bring everything together into a cohesive project. We must incorporate the app that exists from Biomotum, create a game in the Flame game engine, and store user data into a profile that is uploaded to AWS for future analysis by a doctor or researcher. Figure 1 details what our solution will look like all put together.

Looking at the system diagram, the top layer represents Biomotum's current app and exoskeleton. The app is capable of looking for a Bluetooth connection and handling the data streams. The app is also capable of running trials and collecting data. Our project will extend the functionality of this app by creating a game mode that runs a trial and prompts the user for certain inputs via the game loop. If the user meets the target goal the game will reflect a positive outcome by earning coins. An example of how this will work is by showing a character running in a two-dimensional plane, either side to side or forward, and when we want a player to engage their leg muscles, an obstacle will appear; if the target is met the character will leap over the obstacle gaining coins and stumble if they do not meet the threshold. A user will first need to create a profile. If a player profile is already created, they can select that profile, or a user can

create a new profile. Upon creating and selecting a profile, the app will calibrate the game and exoskeleton and start a new game.

The game will incorporate the Flame game engine as seen in Figure 1. Assets for the game such as sprites, animations, and sounds will be displayed to the screen via a gameplay loop. Walking data, visualized by the green arrows in the system diagram, will be used as the user's input for the game. The metrics of how a user is performing will be based on measurements such as FSR(Foot Sensor Reading) data, a sensor in the footplate of the exoskeleton determining force applied during a gait cycle. Other measurements will include the velocity at which a user flexes their ankle via the use of an angle sensor located at the ankle of the exoskeleton.

Once the trial/game has been completed, the data will be collected into the player’s profile where past trials can be viewed and progress tracked. Data will be stored locally on the mobile device as well as uploaded to AWS where a doctor, physical therapist, researcher, or parent can view in-depth data on how a user performed and download the data to be used in further studies or treatments.

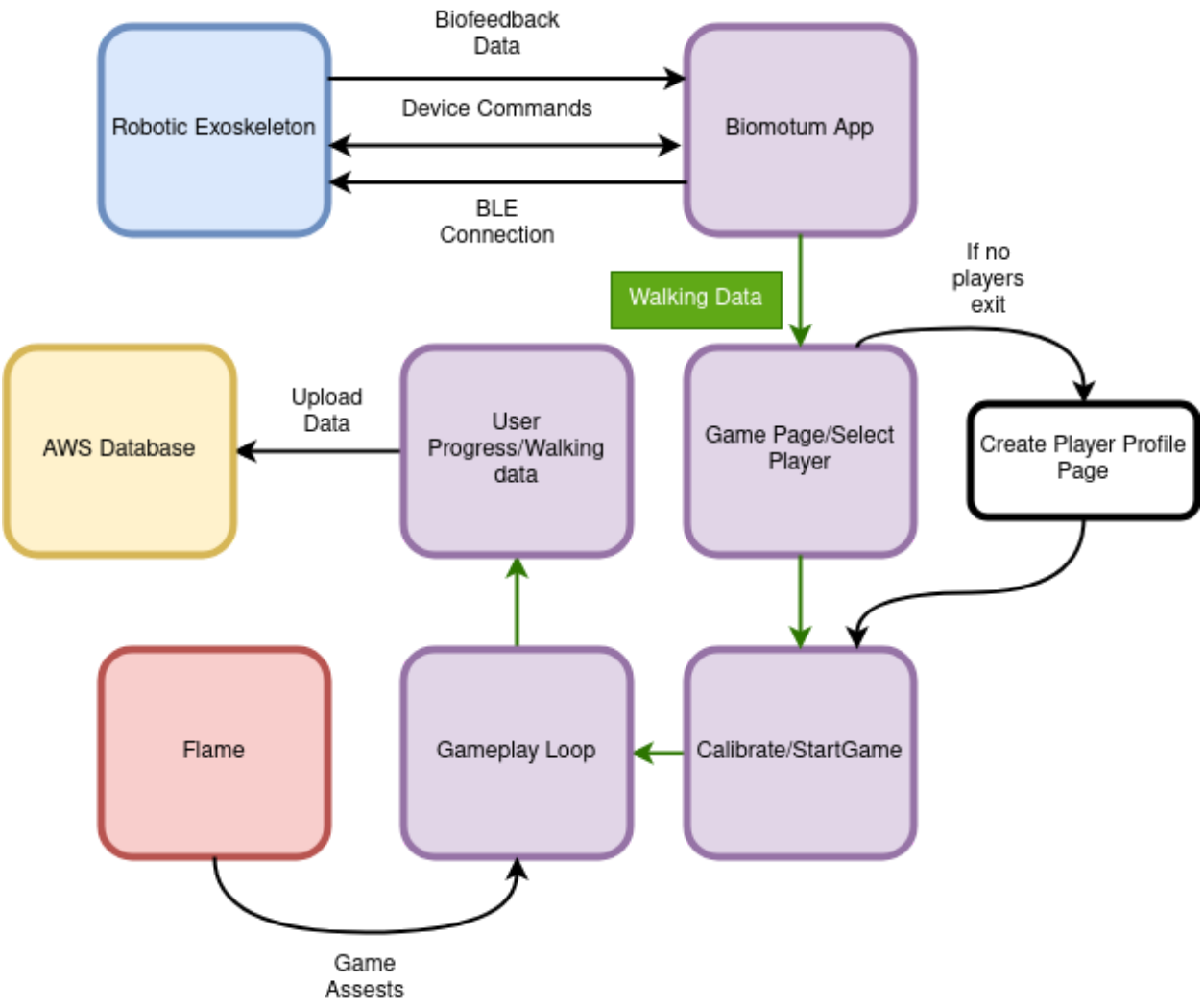

Figure 1: Motus Methods' System Diagram

Conclusion

Physical therapy is a challenging thing to go through. It requires a lot of physical effort that, without incentivizing goals, makes it hard to persevere through. Many individuals fail to see much progress. Our team aims to make such trials easier for children by providing them with a system of play and progress. We aim to create a game that can be played by walking with Biomotum's exoskeletons and provide the child with motivating, engaging, and entertaining gameplay while at the same time collecting data to utilize for aid in the child's physical development.

Flame and Flutter will help us achieve this goal by incorporating the technology developed with Biomotum's iOS app and extending the functionality of the said app with a game of our team's design by utilizing the Flame game engine. This will provide us with the smoothest development process where we will not have to worry about creating an app from the ground up with basic Bluetooth functionality and data collection, incorporating the one that already exists. With Flame, an engine designed to work with Flutter, we can easily and quickly jump into the game development process faster than creating an engine of our own or utilizing other existing engines that do not work as well with Flutter. By using the Biomotum app as a jumping-off point, we can take advantage of what was already built and add more to it to make a project that is engaging for children.

Motus Methods is confident from our planning that we will be able to develop this solution. We have considered many possible angles we can tackle this problem and feel we have chosen the best solutions for this problem. As we move forward, we will collect the requirements, create designs for the game, and collaborate with our client, Dr. Lerner, as well as researchers at NAU's Lab of Biomechatronics to build and test our project to make it the best it can be.