S.T.E.P.S.

Style Tracking Expressive Pad System- Arcade Dance Rhythm Game Leveraging CV Pose Detection

Group 8 Authors

Christopher Solanilla	Jani Jon Lumibao	Kaila Peeples	Andres Abrams
Computer Engineering	*Computer Engineering*	*Photonics and Science Engineering*	*Computer Engineering*

Reviewer Committee:

Dr. Stephen Eikenberry	*CREOL*	*Professor*
Dr. Saleem Sahawneh	*ECE*	*Lecturer*

Mentor:

Dr. Chung Yong Chan
Dr. Aravinda Kar

2. Project Description

2.1 Motivation and Background

In recent years, rhythm games have surged in popularity among both casual and competitive gamers. Titles like *Dance Dance Revolution (DDR)*, *Pump It Up (PIU)*, and *StepManiaX* offer not only fast-paced gameplay but also unique forms of physical interaction that make them stand out from traditional video games. Among these, *StepManiaX* has played a particularly influential role in the inception of our project, as it is actively available at a UCF campus restaurant and regularly enjoyed by students.

StepManiax on UCF at Knightros

For readers unfamiliar with rhythm games: players typically choose a song, and as the music plays, visual cues (usually arrows) scroll on the screen toward a target zone. Players must step on corresponding arrows on the dance pad in time with the music.

Successful timing earns points, while misses break combos and reduce scores. In games aforementioned, players are often using their feet to step on panels that are labeled with directional arrows. In DDR it is with the up down left right arrows, StepManiaX has the same but with an additional center arrow, and Pump it Up has diagonal arrows instead of the 4 up down left right arrows with the center note as well.

As a lifelong fan of rhythm games, I’ve always aspired to create a game that blends the fast-paced footwork of DDR and PIU with innovative mechanics that reward not only timing accuracy but also expressive performance. This passion became the foundation for our senior design project: a custom-built rhythm arcade machine that reimagines the traditional dance pad format. Our system features a unique 9-panel layout, combining the four cardinal directions, four diagonals, and a center panel. This effectively merges the core mechanics of DDR and PIU into a new hybrid experience.

To take it a *step* further, we are integrating a computer vision system capable of analyzing player movement during gameplay. This system detects whether the player completes the charts with minimal effort or performs dynamic, stylish movements such as spins or arm gestures. Players who demonstrate expressive flair are rewarded through a secondary metric we call the Style Score, adding a new dimension to gameplay that celebrates both precision and creativity.

Our team brings a diverse range of skills and experiences to the project. I contribute both a strong passion for rhythm games and hands-on experience from developing a basic dance pad prototype in the past. One of our teammates has a background in dance, offering valuable insight into expressive movement and physical design. Another teammate serves as the president of the UCF Esports Club, providing a competitive gaming perspective that helps shape our gameplay mechanics and balance. All three of us are Computer Engineering majors with solid experience in hardware integration and software development. Additionally, the teammate with a dance background is well-versed in PCB design, making them an asset for developing the input hardware. Our fourth member, a Photonics Science and Engineering student, brings specialized expertise in optics. This makes her an asset for enhancing the performance of our computer vision system through lens design or optimization.

We believe this blend of technical, creative, and performance-oriented backgrounds makes our team uniquely positioned to create a rhythm game that is both entertaining and technically ambitious. By combining game development, embedded hardware, and real-time computer vision, we aim to push the boundaries of traditional rhythm games. Ultimately, we envision this system not just as a school project, but as a potential commercial product suitable for both arcade and home use. Overall, our project is a tribute to the genre we love and an innovative leap forward in how rhythm games are played.

2.2 Project Goals

2.2.1 Hardware

Basic Goals:
The main goal we try to achieve with the PCB is to not only be able to provide the correct functionalities throughout the pad, but to also have enough ports to ensure that all arrow pads and LEDs respond as expected. Under each arrow pad, we will be using Force-Sensing Resistors (FSRs) and LEDs connected to an Arduino and Raspberry Pi based PCB that would be on one edge of the pad. It's very important that our PCB has enough ports for wires to connect to each pad so we will most likely add external GPIO peripheral ports. In terms of connections, in the mechanical part of the project, we hope to minimize wire traffic so that maintenance within the hardware of the pad is easy to navigate around.On the optical side, the primary objective is to enhance system efficiency by integrating a tailored optical component. This will involve designing a custom single-lens system that delivers sharper, more accurate images to the AI-based motion analysis tool. By improving image clarity and feature contrast at the optical level, the system can reduce the computational load required for pose estimation and movement assessment, resulting in faster and more efficient performance.

Advanced Goals:
To minimize the cost, one advanced goal we can achieve through extra time and effort is having a smaller but efficient and maintainable PCB board. More specifically we want to minimize the computational time of the PCB board. This could in turn help minimize costs, as well as help keep the design of the pad from looking bulky. In addition to the PCB board, we aim to have it protected by some cover - if possible, we aim to use 3D printing to ensure the board itself is protected from external risks like water or debris. Adding on to minimizing bulkiness of the design, because we would need different layers of the FSRs, LEDs, and the arrow pads, as much as possible we want to aim for a design that's sleek and light. Because we do aim to make the pad with hard materials instead of making it a rollable plastic or rubber mat, it's very important that we don't make the product way too heavy or bulky.To advance the lens system another lens would be added to correct specific issues. Including another lens would allow for sharper focus across the entirety of the dance pad system, enabling the system to track the moves of the player with greater precision. Some optical issues that could be addressed are chromatic aberration, depth of field limitations and field curvature. These issues are hard to address with a single lens.

Stretch Goals:
A potential stretch goal that we could implement, if not by the end of the deadline, then in the future, is a bluetooth feature where the pad doesn't need to be connected through wiring so the player can be as far from the screen and still enjoy the game. Of course, this would introduce the potential risk of inputs not registering on time so this would take some more effort to ensure peak response time for the player's inputs. Aside from functionalities, creating a hand bar to allow the user to hold on to while playing the game is a nice-to-have part to add, but only if we have extra time and money to add it to the pad. If so, most likely it would be foldable or removable so that it's more portable or easy to assemble. On the lens side, to advance the system even further, adaptive lens technology would be added. This would require either electronically tunable lenses or lens arrays that can switch focus in real time. This goal would require significant amounts of optical design and would raise the cost of the system by a substantial amount. However, it would provide the system to dynamically adjust to a multitude of factors such as lighting conditions, gameplay scenarios, and different players. This system could also bring in the possibility of 3D motion capture by using a stereo setup. Being able to capture the scene from two different perspectives would support both multiplayer additions and gain the ability to assess highly expressive players at an even faster rate.[1]

2.2.2 Software

Basic Goals
The core software functionality centers around developing a custom rhythm game engine tailored to the 9-panel pad input system. The game will read chart files that dictate when each directional pad must be pressed and evaluate player input using real-time data from FSR sensors. Additionally, the software will handle visual display output on a touchscreen interface, play synchronized audio, and compute accuracy-based scores. The system will also interface with a camera to collect real-time video data for the Style Scoring module. Initially, this data may be sent to an external cloud service or AWS instance for evaluation. A basic UI and menu system will be implemented to let players select songs and view results.

Advanced Goals
As we continue to develop the rhythm game engine, an advanced goal is to transition from using external cloud services to processing computer vision locally. Although cloud

servers are powerful, relying on them introduces latency due to the time it takes to transmit video data and receive results. By integrating a lightweight pose detection framework such as MediaPipe or OpenPose directly into the game engine, we can eliminate this network delay. This results in faster, more responsive gameplay while also allowing the system to operate completely offline.Another advanced feature would be a custom chart editor built into the interface, allowing users to upload songs and design their own step charts directly in-game. This editor would include playback tools, snap-to-beat timing, and optional auto-generation of charts using audio analysis. These additions would greatly increase user engagement and flexibility while helping build a wider library of content over time.

Stretch Goals

If time and resources allow, one of our major stretch goals is to transform the game into a fully self-contained platform that can be deployed in arcade or public spaces. This includes developing a player account system where users can log in, track progress, and build profiles tied to their performance history. To support this, we plan to implement a cloud-connected leaderboard system where players can upload high scores, view global rankings, and compare results with others. To streamline access and convenience, we also aim to develop a companion mobile app that allows users to browse scores, review play history, and potentially even queue songs remotely or receive post-game performance feedback.

In addition, we would like to package the system for deployment in arcades, which means optimizing the UI for touchscreen input, streamlining the startup and song selection process, and ensuring that the game can run continuously in a commercial environment. This may also involve implementing features like operator/admin settings, maintenance diagnostics, and game reset options. Finally, additional stretch goals include dynamic difficulty adjustment, where the game adapts in real time based on player performance, and the inclusion of a tutorial mode that helps new players improve through visual prompts and performance analysis. These features would not be necessary for the core prototype but would push the game closer to being a polished, commercially viable product.

2.3 Existing Product/Past Project/Prior Related Work

2.3.1 Dance Dance Revolution

Dance Dance Revolution (DDR), developed by Konami in 1998, is one of the most iconic rhythm games in the world. Players step on a 4-panel dance pad — with up, down, left, and right directional arrows — in time with scrolling on-screen cues synchronized to music. DDR is widely recognized for its role in popularizing rhythm games globally and has been featured in both arcade and home console formats. It uses pressure-sensitive panels and a scoring system based on timing accuracy, rewarding "Perfect," "Great," or "Miss" for each input. However, the game focuses purely on foot-based precision and lacks a scoring component for stylistic or expressive movement. Players who still choose to complete charts with style are called “free stylers” and are highly respected in the community for completing easy to mid level charts with very hard expressive and complicated movement. This movement can range anywhere from spinning, to swaying the arms, to handstands and break dancing.

2.3.2 Pump It Up

Pump It Up (PIU), developed by Andamiro in 1999, is a 5-panel dance rhythm game that includes four diagonally placed panels and a center panel. PIU emphasizes freestyle movement more than DDR and is especially popular in South Korea. While PIU retains the same timing-based scoring mechanics as DDR, it introduces more physically varied and complex choreography due to its diagonal input layout. Nonetheless, PIU still lacks any integrated camera system or style-based scoring . Just like DDR, PIU has the same respected players who complete charts with freestyling despite still not being rewarded for doing so in game.

2.3.3 Dance Around

Dance Around is a rhythm game similar to Dance Dance Revolution but instead of using pressure-sensitive dance pads, it relies solely on a camera-based motion tracking system. This is done by using VisionPose to generate a 3D model of the player’s body and assessing their dance performance based on their ability to match the poses given and their own expressive movements[2]. During the game, players are prompted to mimic target poses displayed on the monitor, with visual cues that indicate the proper hand or foot placement. However, customer feedback has highlighted several limitations of this system. Customers have stated that the game has consitently misevaluated full body movements and is only capable of   capturing the hand and foot positions of the players reliably[3]. Additionally, customers have noted that the calibration process could be overly lengthy, which detracts from the ease of use.

2.3.4 Dancerush Stardom

Developed by Konami, DANCERUSH STARDOM is a freestyle rhythm dance game that eliminates the traditional dance pad structure in favor of a large pressure-sensitive surface. The game uses a camera system to provide feedback and record gameplay, but the scoring is still based on foot movement across a large flat pad with visual indicators for steps and slides. The game promotes freestyle dancing, including spins and slides, and is considered more modern and expressive than DDR. However, like Dance Around, it does not feature a true pose recognition or performance grading system. Flair and dance expression are encouraged but not quantitatively rewarded within the game mechanics.

2.3.5 StepManiaX

StepManiaX is a rhythm game inspired by DDR and PIU and developed by the creators of StepMania. It uses a 5-panel pad layout with center, up, down, left, and right panels, and is designed for high durability and fitness applications. StepManiaX includes a touchscreen interface and modernized music selection UI, with content designed to be accessible for both casual and serious players. While it modernizes the user experience and supports a wide range of difficulty levels, it retains the traditional scoring focus on timing accuracy. There is no implementation of camera-based tracking or any scoring system that rewards visual expression or dance style beyond note timing.

Illustration prototype of StepManiaX, a 5 panel Dance Rhythm Game Arcade Cabinet[6]

2.4 Objectives

The main objective of this project is to design and build a self-contained arcade-style rhythm game system that includes both the physical dance pad hardware and the video game software. The system will feature a 9-panel layout to allow for expanded gameplay mechanics. In addition to foot-based input, the game will include a computer vision system that evaluates the player's expressive movement using a live camera feed. Our goal is to create an experience that rewards both precision and performance, allowing players to interact with the game through both steps and body movements. The final product will include a working rhythm game engine, a functioning pad-to-PC interface, custom chart creation tools, and vision-based style scoring.

We aim to complete the following specific tasks over the course of SD1 and early SD2. These objectives are listed in the rough order we plan to execute them, starting from component acquisition and hardware assembly to basic software integration and vision feature prototyping.

1. **Acquire all essential components**: order FSR sensors, RGB LEDs, microcontroller (MCU), Raspberry Pi, USB connectors, power regulators, and camera module based on BOM list.

2. **Design and fabricate the dance pad platform**: cut and assemble the frame using plywood and aluminum, and mount transparent top layers and non-slip base.

3. **Design a custom PCB schematic**: using KiCad or similar, create and route a PCB that connects FSR sensors, LEDs, and MCU. Ensure enough GPIOs and power regulation are included.

4. **Send PCB design for fabrication** and assemble the physical board with soldered components and headers.

5. **Write microcontroller firmware** to read analog signals from FSRs and convert them into digital keypress events via USB HID protocol.

6. **Test PCB + FSR response time** using a debug script to confirm low-latency response (<10ms) when pressing panels.

7. **Integrate pad input with PC**: confirm that pressing physical panels triggers keyboard inputs correctly on a connected computer (e.g., using a diagnostic test

page).

8. **Initialize the GitHub repository** and set up version control for both hardware schematics and game software.

9. **Begin a Godot project for the rhythm game engine**: set up a basic Godot 4.3 scene with a UI, note spawning, and music synchronization framework.

10. **Map keyboard inputs to note triggers** in Godot and test gameplay using manual inputs (before full integration).

11. **Design and implement a basic UI** with song selection, results screen, and audio-visual feedback for note accuracy (e.g., Perfect/Good/Miss text).

12. **Write a chart parser and loader** to read external step charts from a JSON or custom format and spawn notes accordingly.

13. **Connect the pad to the game** and test real gameplay — stepping on the pad triggers notes and feedback in the game engine.

14. **Integrate camera with Raspberry Pi** and verify real-time feed is available to the game via local network or USB interface.

15. **Use a pre-trained pose detection library** such as OpenCV, Media Pipe, or OpenPose to extract body keypoints from live video.

16. **Detect key Poses and Gestures:** For our project, we will aim at detecting specific poses from our list of poses including the Y pose, the Tiger pose, the Mantis pose, and the Flamingo pose.

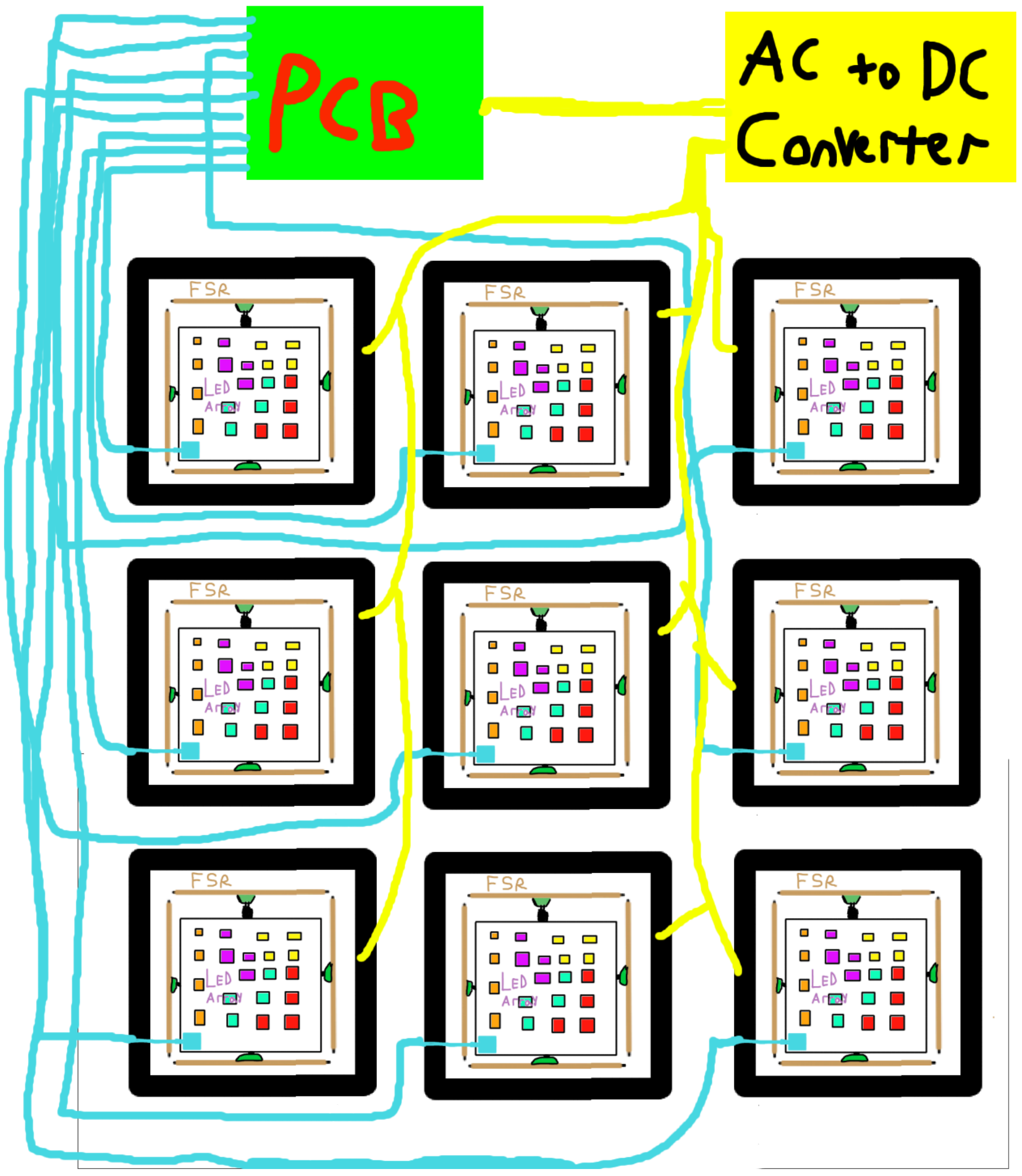
17. **Develop a basic Style Score system** that assigns bonus points based on detected expressive movements.

18. **Build a simple chart editor UI** to manually align step notes to music and export them for testing.

19. **Tune optical setup**: test different lenses or lighting setups to improve contrast and reduce computational load on the pose detection module.

20. **Document each milestone** with photos, commit history, and GitHub issues, and ensure each part is demo-ready for SD1 and SD2 deliverables.

2.4.2 Prototype illustration/Blueprint



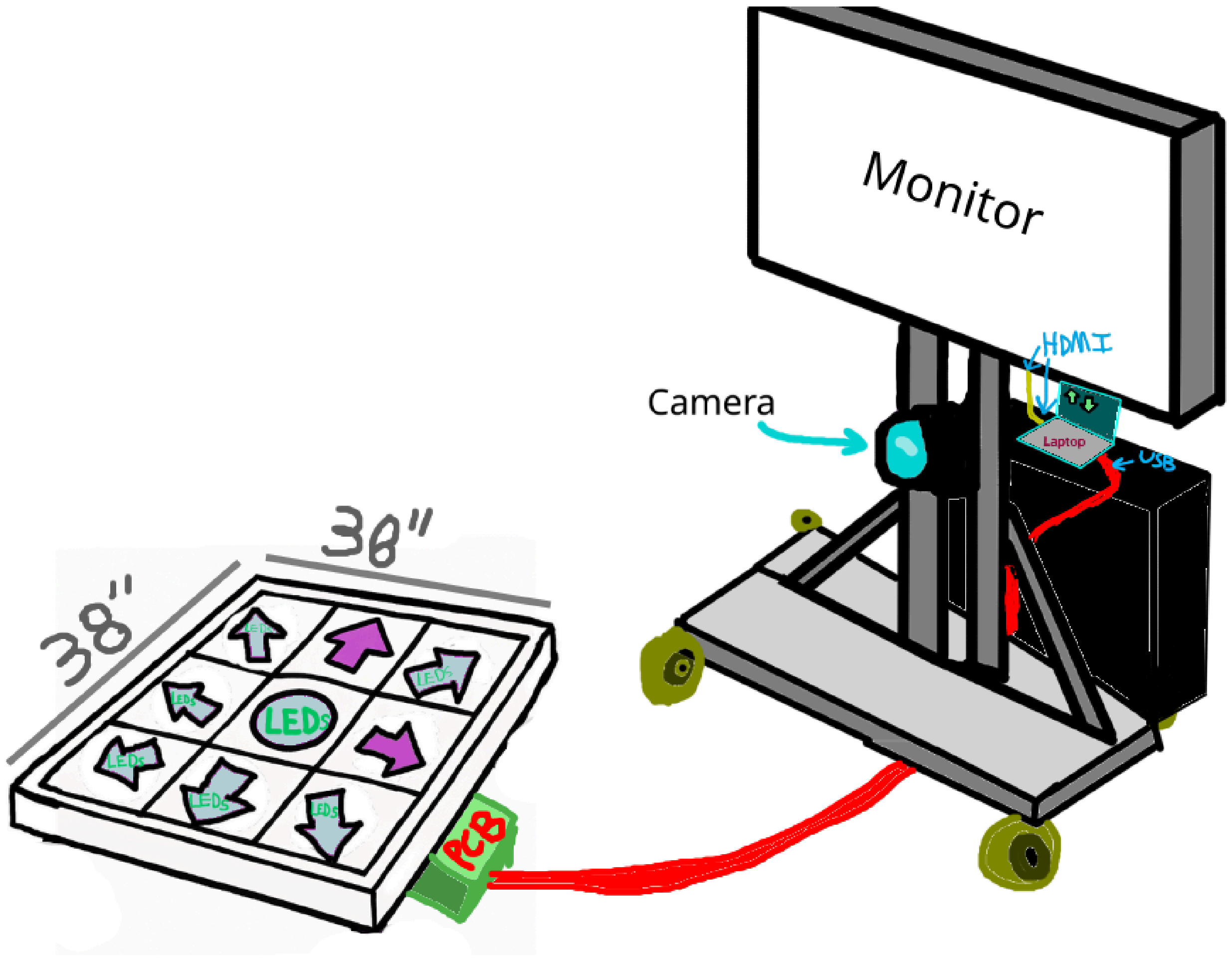


Figure 2.4.2- 2.4.3 Prototype Illustration

2.5 Project Features and Functionalities

The primary goal of this project is to develop a working proof-of-concept for a new kind of dance rhythm game that integrates physical input with computer vision-based expression tracking. The focus is on delivering a functional and demonstrable system that includes three core components: a responsive 9-panel dance pad, a custom rhythm video game engine, and a vision system that scores player movement based on style and expressiveness.

At the hardware level, the dance pad will use Force-Sensing Resistors (FSRs) beneath each panel to detect foot pressure and translate those inputs into digital signals via a microcontroller. The signals are interpreted as button presses in-game, allowing for responsive gameplay. Each panel will also include LED lighting to provide immediate visual feedback based on the game's state and player interaction. While commercial-grade materials like metal panels or acrylic overlays would be ideal, the immediate priority is to build a stable, functional pad using accessible prototyping materials to prove the input system works reliably.

The software side features a rhythm game engine tailored to the 9-direction input system. Players will step on directional pads in sync with music, guided by scrolling notes on screen. The game will evaluate the player's timing accuracy and display scores at the end of each round. Alongside traditional gameplay scoring, a connected camera system will assess the player's full-body movement using a pose detection library. Based on the amplitude, variation, and expressiveness of the player's dance, the game will generate a secondary "style score." This adds a creative and engaging layer of performance evaluation beyond pure timing.

A basic user interface will allow for song selection, score displays, and navigation through the system. A chart editor tool will also be included, enabling users to import music and design their own charts either manually or with the help of automatic generation tools.

Although not essential to the proof-of-concept, we also envision the long-term possibility of turning the system into a self-contained arcade-style unit. Features like a cabinet enclosure, co-op integration, player logins, and online leaderboards are considered stretch goals that could be implemented later with more resources. The project is designed with scalability in mind: the core technology should work independently, while leaving room for future upgrades to polish and deploy the system as a full commercial or open-source product.

2.5.1 Style Score and Pose-Based Evaluation System

In addition to the traditional timing-based score system, our game introduces a novel secondary metric known as the Style Score, which rewards players for striking expressive and clearly defined poses at designated times during gameplay. Unlike freeform dance scoring, our system focuses on detecting static full-body poses that can be consistently recognized by a pose estimation algorithm.

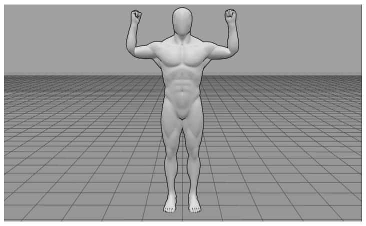
We intentionally limit the scope of detection to distinct, predefined poses that can be reliably tracked in real-time using a single camera. This approach improves detection accuracy and reduces computational load while still encouraging expressive movement.

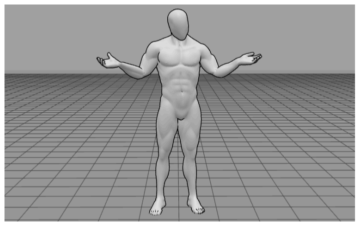
During gameplay, specific pose prompts will appear, similar to freestyle moments or bonus sections. These pose prompts will be displayed in a small icon area on screen during designated freestyle sections, giving players 2–3 seconds to match the target pose. If the player strikes the correct pose at the right time, they are awarded bonus points to their Style Score. This incentivizes physical creativity and rewards players who engage more fully with the visual performance aspect of rhythm games.

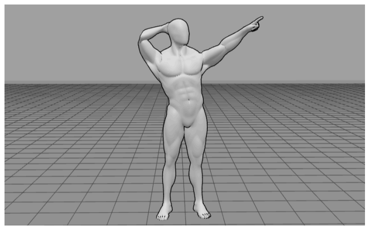
Tentative Pose List:

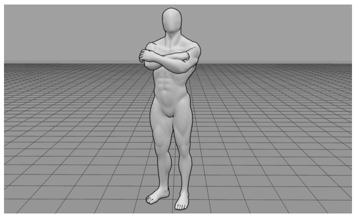
Each pose is chosen based on ease of detection, body separation, and iconic visual silhouette (all poses are mirrored when displayed to the player):

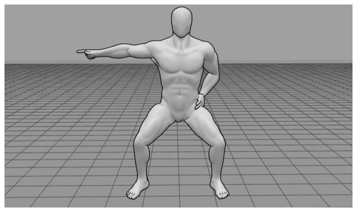
- **Muscle Man Pose**: Both arms raised as if you're flexing biceps'
- **What? Pose:** Both arms out as if you don't know about something
- **Point Up Pose:** Left arm behind the head, pointing out to a plane
- **Tough Guy Pose**: Crossed arms
- **Samurai Pose** : Legs wide apart, one hand near waist as if gripping a katana, other arm pointing forward or out.
- **Mantis Pose**: Right arm raised in front of the chest, left bent above the head, and right leg up with knee up
- **T Pose**: Both arms stretched straight out to the side (useful base for debugging).

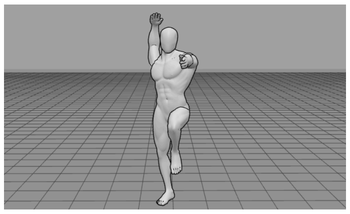

Muscle Man Pose

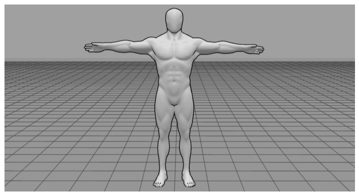

What? Pose


Point Up Pose


Tough Guy Pose


Samurai Pose


Mantis Pose


T Pose

Each pose will have keypoint thresholds that define acceptable angles and positions such as a certain limb being within a range of 20 degrees from the desired state. This will be validated using a pre-trained MediaPipe/OpenPose[5] model.

2.6 Requirements and Specifications

2.6.1 Parts Specification Requirements Table

Table 2.6.1 List of all of the project's engineering requirements and specifications. Highlighted in yellow are specifications that will be demonstrated

~~Specifications/Requirements Table~~	
Printed Circuit Board (PCB)	
Parameter	Value
~~Maximum size of the board~~	≤ 10 cm^2
Personal Computer (PC) [System Requirements to Run Game]	
Parameter	Value
Frames per second	≥ 60 fps
Game resolution	≥ 1280 * 720 pixels
Refresh rate	≥ 120 Hz
Force-Sensing Resistors (FSRs)	
Parameter	Value
Input response time	< 10 ms (almost instantaneous)
Force range input	0-100 N
Size range	~12mm^2
Microcontroller (MCU)	
Parameter	Value
Minimum GPIOs	~15 pins
Clock frequency	≥ 16 MHz
Power Regulator	

Parameter	Value
Input voltage from wall power	≥ 12 V
~~Output voltage and current~~	~~≥ 3.3 V running at ≥ 500 mA~~
Camera Module	
Parameter	Value
PSF(Sharpness, FWHM)	≤ 1.5 pixels at image center, ≤ 2 pixels at the corners.
Contrast-to-background Ratio at body edges	≥8:1 Contrast between player and background at limb boundaries
Uniform illumination across field	≥ 90% brightness uniformity from center to corners of frame.
RGB LEDs	
Parameter	Value
Luminous efficacy	≥ 60 lm/W
~~Input voltage from USB~~	~~≥ 5V~~
Current per LED	≤ 60 mA
Refresh rate	≥ 400 Hz
Operating temperature range	-20℃ to 60℃

2.6.2 Notes on Specifications

Printed Circuit Board (PCB)

- As mentioned before in goals, we want to minimize the size of the PCB as much as we can while also making it easy enough to check for errors on the design.

Personal Computer (PC) [System Requirements to Run the Game]

- Whether it be a PC or game system, the system requirements are the same. In order to run our custom made game, the system must be able to run at least 60 frames per second, have a resolution of at least 1280 * 720 pixels, and have a refresh rate of at least 120 Hz.

Force-Sensing Resistors (FSRs)

- Response time must be almost instantaneous and this is very important because rhythm games are dependent on timing of when commands are hit. There should be little to no latency when pressing the pads and triggering the FSRs
- The amount of force applied needed to trigger the FSRs must be 0-100 N so that even the smallest amount of pressure should be enough to signal

<u>Microcontroller (MCU)</u>

- The MCU we decide to use must support the USB device we will be using to connect the pad to a computer to play games. Without this support, the pad will be unable to turn on since our pad will be USB powered instead of battery powered
- Minimum number of GPIOs must be about 15 pins, if needed, we can add in GPIO peripherals to ensure we have enough pins available for our I/O
- Clock frequency must be at least 16 MHz because it would ensure sufficient processing speed for responsive input handling and LED control. At least 16 MHz would also meet the minimum timing requirements for USB communication in many MCUs such as ATmega32U4. If needed for better performance, we can aim for faster MCUs that can run higher than 16 MHz.

<u>Power Regulator</u>

- The pad is powered through a wall power cable which should be more than sufficient enough to power the pad. The input voltage of at least 12 V is required to efficiently power the whole pad. Most of the components we would use for the project such as MCUs, sensors, and RGB LEDs will operate at around 3.3V. Regulating from 5 V input to 3.3 V output would be stable enough for power regulation throughout most of our components. At least 500 mA is needed to provide enough current for all of the active components within the pad like the MCU, FSRs, and RGB LEDs.

<u>Camera Module</u>

- Having a PSF of ≤1.5 pixels at the center and ≤2 pixels at the corners ensures that image details remain well-resolved and sharp , which reduces the need for MediaPipe/OpenPose to perform any deblurring or super-resolution. This will allow for faster and more accurate real-time motion analysis.
- A contrast ratio of ≥8:1 around the player’s silhouette enhances foreground-background seperation. This contrast level simplifies keypoint detection and segmentation for the AI model. Having the player’s silhouette stand out more clearly reduces inconsistencies with pose estimation due to varied background conditions and lighting.

- Achieving ≥90% uniformity across the image frame ensures that there is consistent brightness from the center to the corners of the image taken. Minimizing vignetting effects ensures consistent illumination from the center to the corners. This will reduce the need for post-processing correction due to uneven lighting which is especially important in dynamic or uncontrolled game environments such as arcades. Having even illumination further enhances efficiency and accuracy of image extraction.

RGB LEDs

- Each of the RGB LEDs need an input voltage of at least 5V and current of about 60 mA to ensure that the LEDs can each reach full brightness in the pad.
- Refresh rate is also needed as the RGB LEDs are not just turning on, but also pulsing on beat or whenever a player steps on a step panel. A refresh rate of at least 400 Hz should be sufficient enough to generate smooth animations for the LEDs.
- Because we will be using multiple LEDs that will be placed almost all over the dance pad, it’s very important that we avoid them from being too hot as it would introduce the risk of fire. The pad will most likely face generated heat from the player and will definitely trigger the LEDs for the majority of the time the pad is active, so a temperature max of around 60℃ should be bounded for the LEDs. In addition to that, luminous efficacy is important as it directly impacts energy consumption, but we’d also want efficacy to be high enough as we use multiple LEDs in the system.

2.7 Hardware Block Diagram

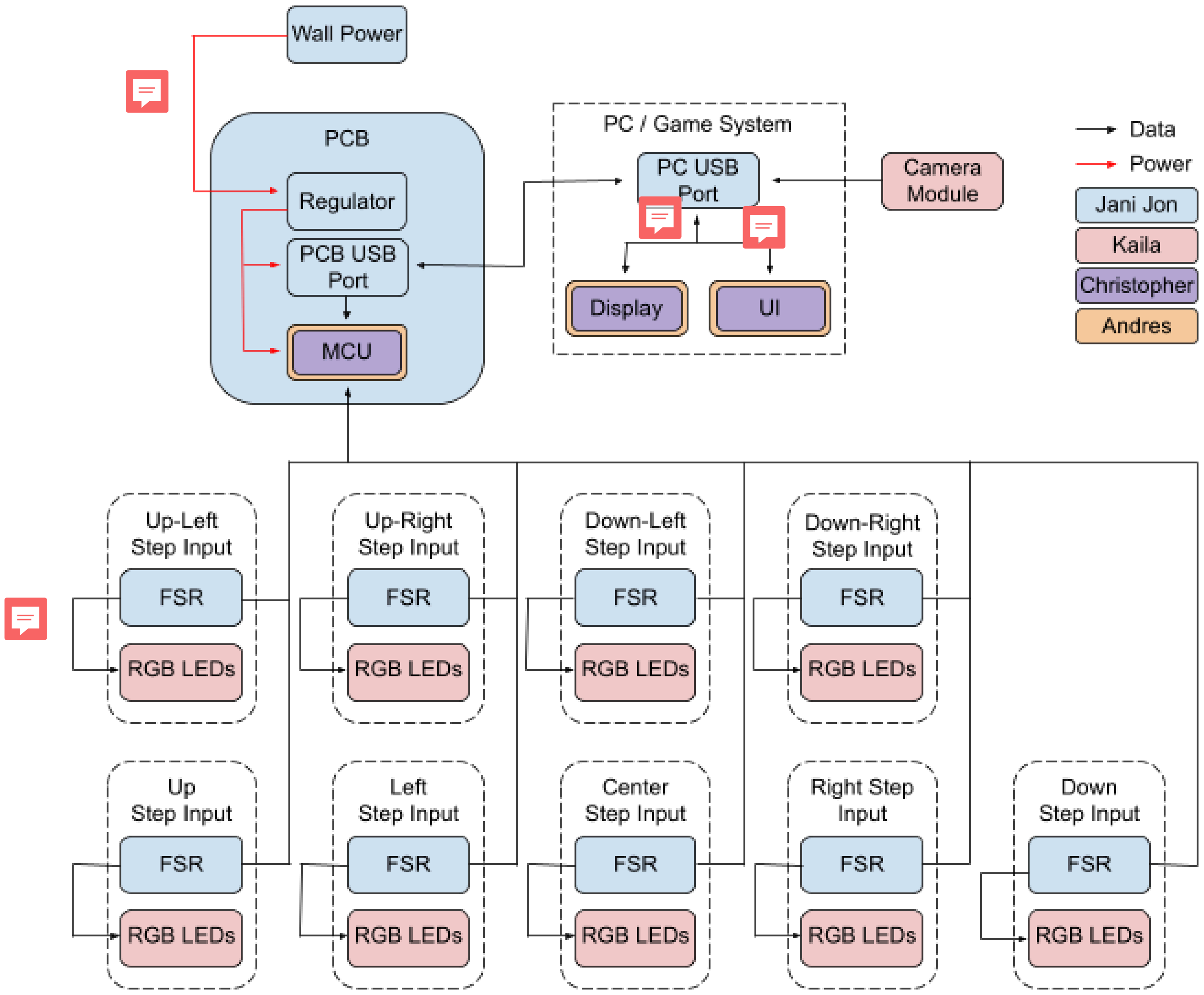


Figure 2.7 Hardware Block Diagram showcasing work distribution and major components of the design

2.8 Software Block Diagram

🟧Andres
🟪Christopher

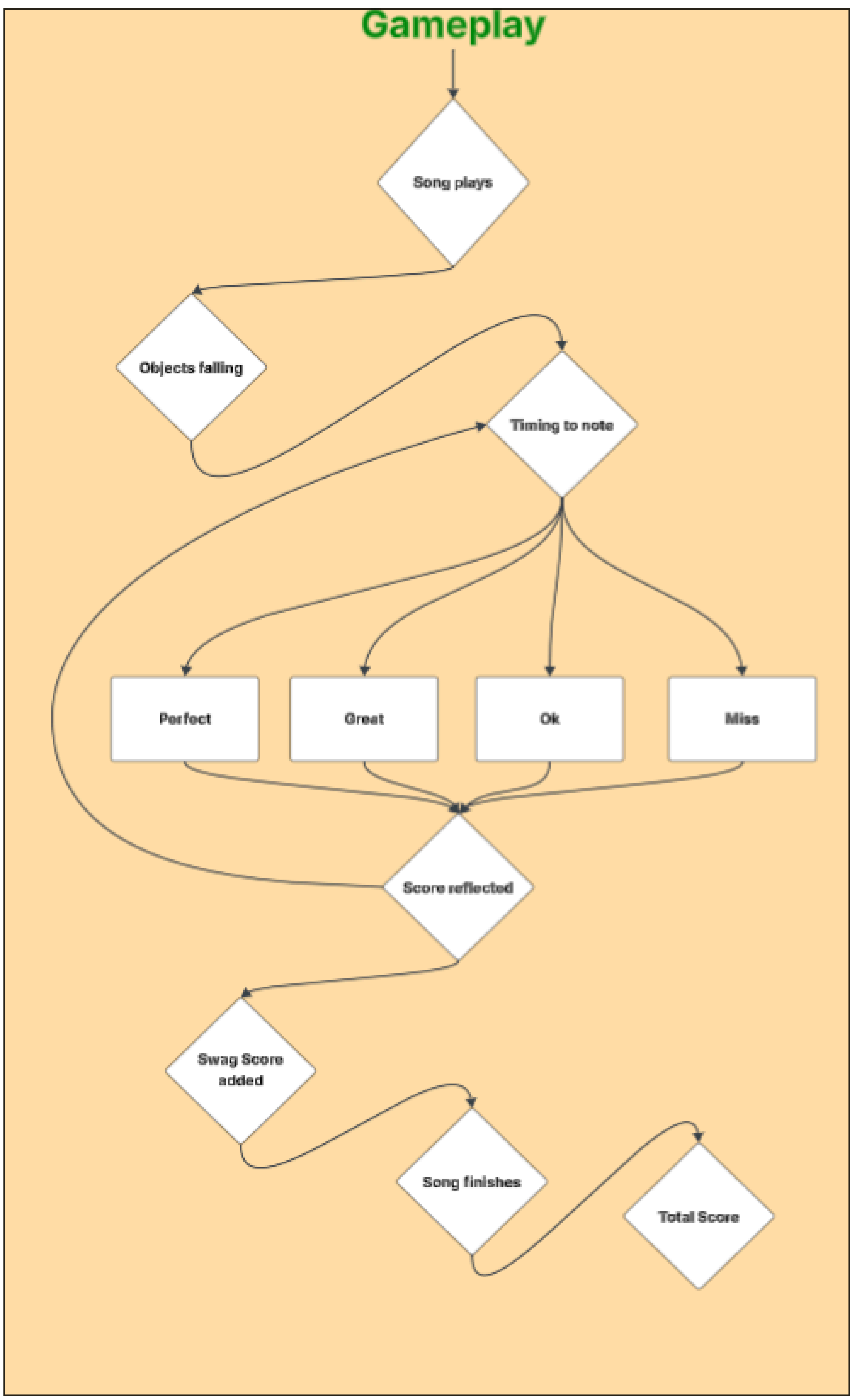

Gameplay
Song plays
Objects falling
Timing to note
Perfect
Great
Ok
Miss
Score reflected
Swag Score added
Song finishes
Total Score

Figure 2.8a Software Block Diagram gameplay

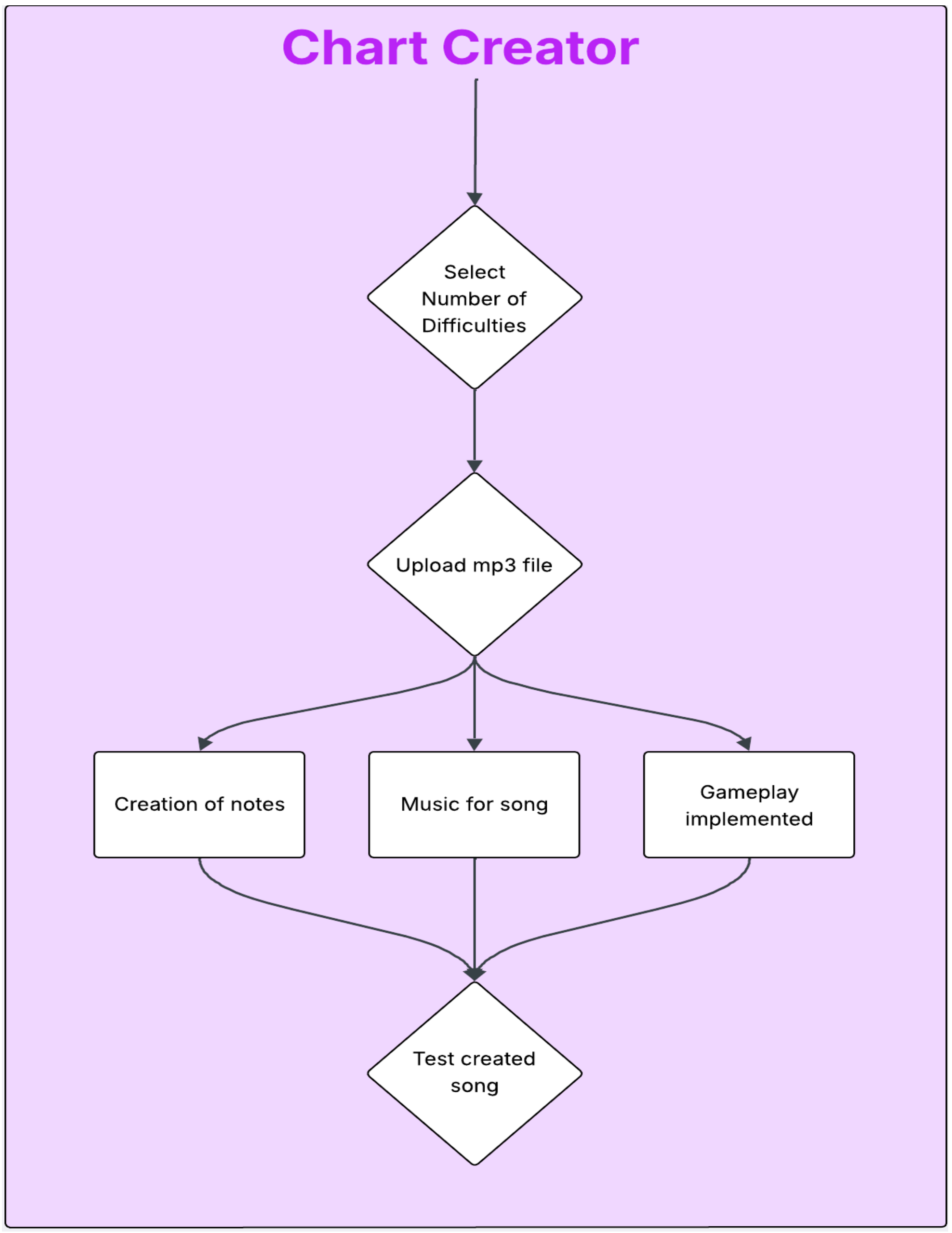


Figure 2.8a Software Block Diagram gameplay

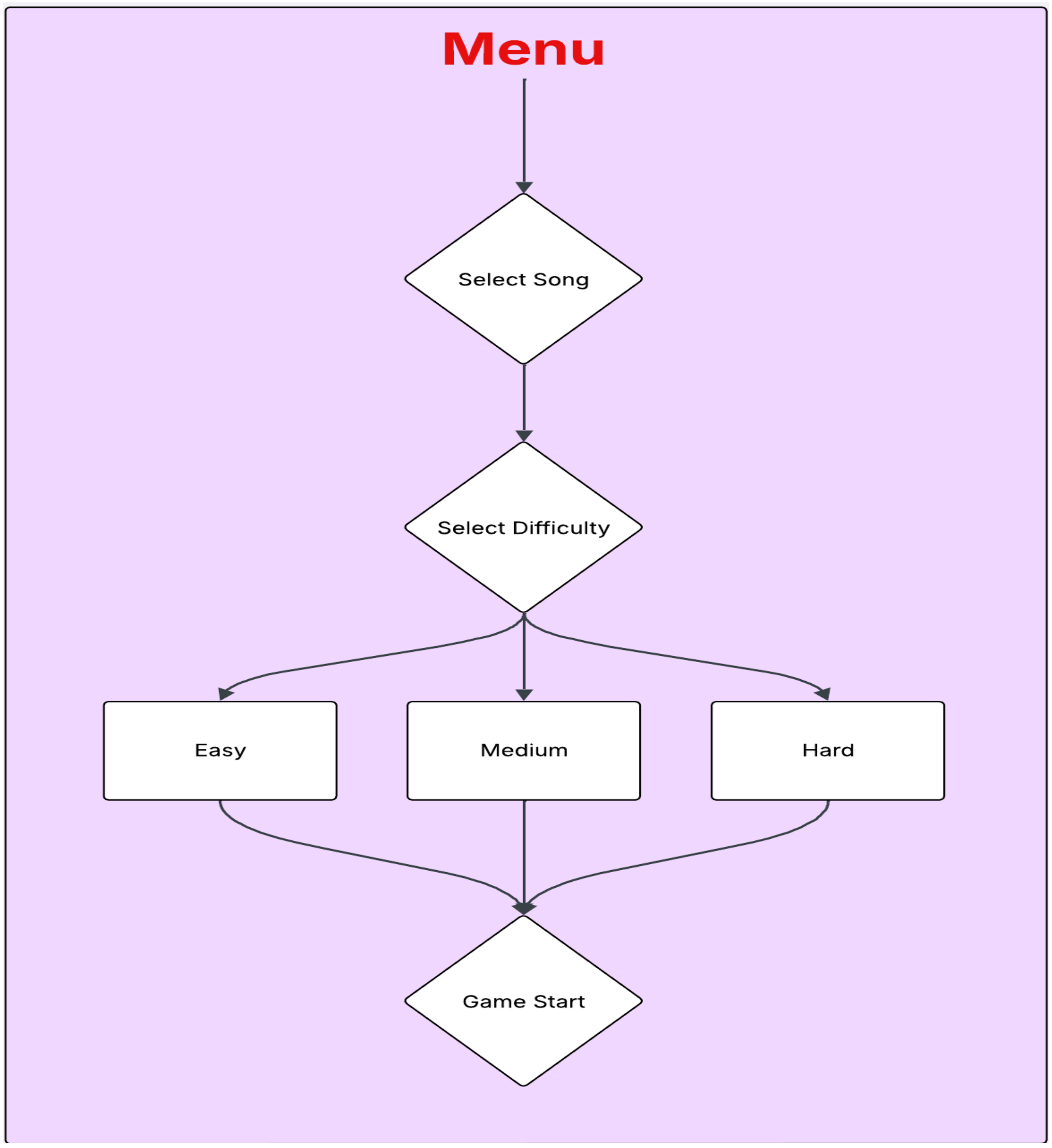
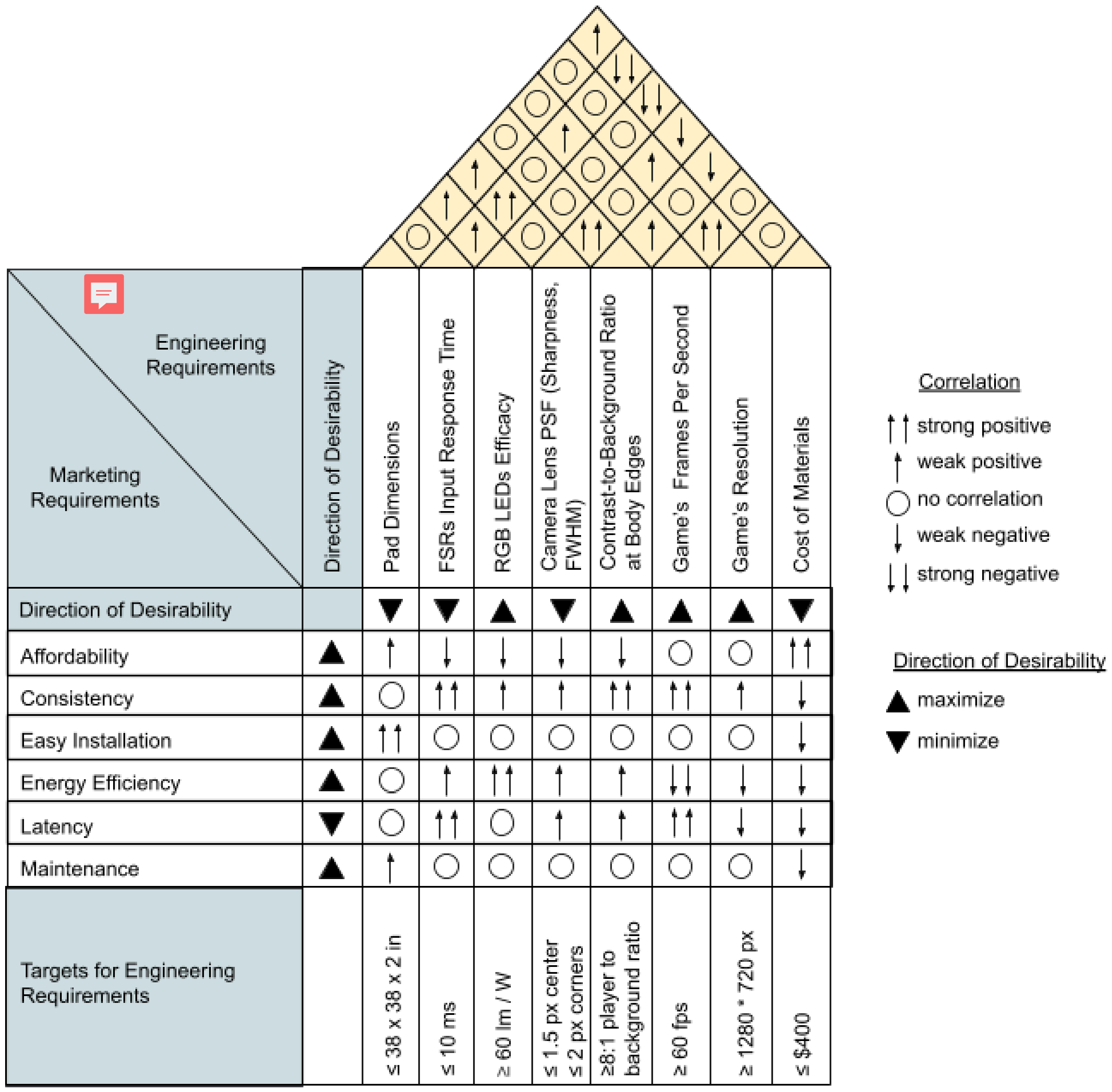


Figure 2.8a Software Block Diagram gameplay

2.10 House of Quality



Marketing Requirements \ Engineering Requirements	Direction of Desirability	Pad Dimensions	FSRs Input Response Time	RGB LEDs Efficacy	Camera Lens PSF (Sharpness, FWHM)	Contrast-to-Background Ratio at Body Edges	Game's Frames Per Second	Game's Resolution	Cost of Materials
Direction of Desirability		▼	▼	▲	▼	▲	▲	▲	▼
Affordability	▲	↑	↓	↓	↓	↓	○	○	↑↑
Consistency	▲	○	↑↑	↑	↑	↑↑	↑↑	↑	↓
Easy Installation	▲	↑↑	○	○	○	○	○	○	↓
Energy Efficiency	▲	○	↑	↑↑	↑	↑	↓↓	↓	↓
Latency	▼	○	↑↑	○	↑	↑	↑↑	↓	↓
Maintenance	▲	↑	○	○	○	○	○	○	↓
Targets for Engineering Requirements		≤ 38 x 38 x 2 in	≤ 10 ms	≥ 60 lm / W	≤ 1.5 px center ≤ 2 px corners	≥8:1 player to background ratio	≥ 60 fps	≥ 1280 * 720 px	≤ $400

Figure 2.10 Shows the House of Quality table of this project's engineering and marketing requirements

10. Administrative Content

10.1 Budget

We are aiming to limit the budget of this project to $400. *Table 10.2* as shown below lists the bill of materials. Although we want to minimize costs, we also want to ensure we have materials that have a good enough quality so that they are reliable and efficient. This includes the framework (hard materials), PCB, RGB LEDs, camera module, and force-sensing resistors. If any of these were bought very cheap without taking into account better affordable options, we would be prone to multiple potential risks once we begin testing.

10.2 Bill of Materials

Item	Dimensions	Estimated Unit Cost	Quantity	Estimated Total Cost
Raspberry Pi HQ Camera	38mm x 38mm x 12.4mm	$50.00	1	$50.00
Custom glass lens	undetermined	50.00	1	$50.00
RGB LEDs	5mm	$0.12	1	$5.99
PCB	undetermined	$50	1	$50
Force-Sensing Resistors	12.7mm x 57mm	$5	18	$90
Plywood	36 `` x 38 x ¼ ``	$50	1	$50
Aluminum Square tubing	38`` x 38`` x 1/8``	$10	6	$60
Polycarbonate Sheets	11.75 ``x 11.75`` x 0.25 ``	$0	9	$0(already have)
Non-Slip Rubber Mat	2ft x 4ft x 3mm	$20	1	$20

Table 10.2 Itemized Bill of Materials

10.3 Project Milestones for SD1 and SD2

10.3.1 Project Milestones for SD1

Table 10.3.1 Project Milestones SD1

Due Week	Advancement
1	Group Creation and have base idea
2	Researching and Innovating based off idea
3	Divide and Conquer Document completed, Have at least 1 committee member
4	Meet for revisions (if any) of Divide and Conquer Document with committee members, upload revised document into group website
4-5	Individual Research
6	30 pages finished
7	Meet for revisions (if any) with committee members
7-8	Testing of components
9	60 pages finished turn in Midterm Report
10	Meet for revisions (if any) with committee members
10-11	Start on video, finishing touches to 120 page document
12	120 page document finished, mini video finalized

10.3.2 Project Milestones for SD2

Table 10.3.2 Project Milestones for SD2

Anticipated Start Week	**Advancement**
1	Camera Completion
1	PAD completion
1	Game Completion
4	Pad and Game integration
4	Camera detection
10	Camera integration
16	Final Day and Live Demo

Appendices

[1]Guerra-Filho, G. B. (2005). Optical Motion Capture: Theory and Implementation. *Revista de Informática Teórica e Aplicada (RITA)*, 12(2), 1–18.

https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=07147486b65d12c4326ccb3ad54ca612b52e1ac3

[2] *Dance around information*. RemyWiki. (n.d.). https://remywiki.com/DANCE_aROUND_Information

[3] *Dance around (AC) - bemani games - music game forums*. ZIv. (n.d.). https://zenius-i-vanisher.com/v5.2/thread?threadid=11041&page=2#:~:text=The%20games%20body%20tracking%20is,games%20that%20cater%20to%20everyone.

[3] *Z. Cao, G. Hidalgo, T. Simon, S. Wei, and Y. Sheikh. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields IEEE TPAMI, 2019. https://doi.org/10.1109/TPAMI.2019.2929257*

[5] *Google MediaPipe. (2024). Pose Estimation. Retrieved from = https://developers.google.com/mediapipe/solutions/vision/pose*

[6] *Step Revolution. (2024). StepManiaX. Kyle Ward Retrieved from https://stepmaniax.com*