Mini Project Report On

Gesture-Based Interactive Game

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science & Engineering

By

Rohan Raghavan (U2103181)

Sneha R. (U2103199)

Sreya P. (U2103202)

Vismaya Balakrishnan (U2103215)

Under the guidance of

Ms. Amitha Mathew

Department of Computer Science & Engineering
Rajagiri School of Engineering & Technology (Autonomous)
(Affiliated to APJ Abdul Kalam Technological University)
Rajagiri Valley, Kakkanad, Kochi, 682039
May 2024

CERTIFICATE

This is to certify that the mini project report entitled ***"GESTURE-BASED INTERACTIVE GAME"*** *is a bonafide record of the work done by* ***Rohan Raghavan (U2103181)****,* ***Sneha R. (U2103199)****,* ***Sreya P. (U2103202)****,* ***Vismaya Balakrishnan (U2103215)****, submitted to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B. Tech.) in Computer Science and Engineering during the academic year 2023-2024.*

Ms. Amitha mathew
Asst. Professor
Dept. of CSE
RSET

Dr. Uma Narayanan
Asst. Professor
Dept. of CSE
RSET

Dr. Preetha K.G.
Head of the Department
Dept. of CSE
RSET

ACKNOWLEDGEMENTS

We wish to express our sincere gratitude towards **Dr P. S. Sreejith**, Principal of RSET, and **Dr. Preetha K.G.**, Head of the Department of Computer Science and Engineering for providing us with the opportunity to undertake our mini project, "GESTURE-BASED INTERACTIVE GAME".

We are highly indebted to our project coordinator, **Dr.Uma Narayanan**, Designation, Department of Computer Science and Engineering for their valuable support.

It is indeed our pleasure and a moment of satisfaction for us to express our sincere gratitude to our project guide **Ms. Amitha Mathew** for her patience and all the priceless advice and wisdom she has shared with us.

Last but not the least, we would like to express our sincere gratitude towards all other teachers and friends for their continuous support and constructive ideas.

Rohan Raghavan

Sreya P.

Sneha R.

Vismaya Balakrishnan

Abstract

The Interactive Sorting Game reimagines learning through a dynamic fusion of entertainment and education, accommodating individuals with varying levels of computer system familiarity and certain disorders. This game introduces players to the fundamental concept of sorting through an immersive gameplay experience. Players engage with the game using intuitive hand gestures captured by OpenCV and refined with MediaPipe. Through these gestures, players manipulate a diverse array of items, each characterised by distinct attributes such as numbers, colours, and letters, sorting them according to predefined criteria. Real-time feedback and dynamic challenges ensure an interactive and rewarding learning journey. Moreover, the integration of TensorFlow enables precise gesture recognition, facilitating accessibility for individuals with motor skill impairments and other disorders. By leveraging alternative methods of interaction, such as gesture-based controls, the game promotes inclusivity, ensuring that all players can actively participate and learn effectively. In summary, the Interactive Sorting Game offers an inclusive and engaging platform for fostering cognitive development and deeper understanding of sorting concepts, all through the innovative use of hand gestures.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

In today's fast-paced digital world, traditional educational methods often struggle to captivate and engage learners accustomed to interactive and immersive experiences. Simultaneously, accessibility remains a pressing concern, with individuals facing physical challenges encountering barriers to accessing educational content.

Recognizing this gap, the Gesture-Based Interactive Sorting Game emerges as a solution that bridges entertainment and education. Leveraging gesture recognition technology, the game offers an intuitive interface, allowing users to interact with educational content through natural hand gestures. This approach not only enhances accessibility but also provides an engaging learning experience for all users.

Moreover, research suggests that immersive learning experiences lead to deeper engagement and better retention of concepts. By integrating educational content with interactive gameplay mechanics, the Gesture-Based Interactive Sorting Game aims to foster cognitive development while keeping users entertained.

In summary, amidst evolving educational needs and technological advancements, the Gesture-Based Interactive Sorting Game represents an innovative approach to learning. By harnessing immersive experiences and cutting-edge technology, the game addresses the challenges of engagement and accessibility, paving the way for more inclusive and effective education.

1.2 Problem Definition

The aim of the Gesture-Based Interactive Sorting Game is to revolutionize learning by combining entertainment and education through immersive gameplay experiences. The project seeks to address the challenge of engaging modern learners and enhancing accessibility by leveraging gesture recognition technology to create a dynamic and inclusive learning platform.

1.3 Scope and Motivation

1.3.1 Scope

The Gesture-Based Interactive Sorting Game project aims to develop a software system integrating hand gesture recognition technology and interactive gameplay mechanics. This includes real-time processing of hand gestures, creation of intuitive graphical user interfaces, and design of sorting and grouping challenges. Educational elements will be incorporated to promote cognitive development and enhance learning outcomes. The project aims to deliver a user-friendly application that caters to diverse user preferences and abilities. Scalability for potential future enhancements will also be considered within the project scope.

1.3.2 Motivation

The project is motivated by the need to address the evolving challenges of engaging modern learners in the digital era. Traditional educational methods often struggle to captivate learners accustomed to interactive experiences. By merging entertainment with education, the project seeks to make learning more engaging and accessible to a wider audience. Additionally, there's a commitment to inclusivity, with a focus on accommodating individuals with physical challenges through intuitive gesture-based interactions. Ultimately, the project aims to empower users to actively participate in their learning journey and achieve meaningful cognitive development through immersive experiences.

1.4 Objectives

1. Develop a robust software system that integrates hand gesture recognition technology with interactive gameplay mechanics.

2. Implement real-time processing of hand gestures to ensure responsive and intuitive user interaction.

3. Design and create intuitive graphical user interfaces that facilitate easy navigation and engagement for users of all abilities.

4. Create diverse sorting and grouping challenges that offer engaging gameplay experiences while reinforcing fundamental concepts.

5. Incorporate educational elements into the game to promote cognitive development and enhance learning outcomes.

6. Ensure scalability of the application for potential future enhancements and expansions to accommodate evolving educational needs.

1.5 Challenges

The Gesture-Based Interactive Sorting Game project faces several challenges, including the need for precise hand gesture recognition to ensure accurate user interaction. Additionally, designing intuitive graphical user interfaces that cater to diverse user preferences and abilities presents a significant challenge. Furthermore, integrating educational elements seamlessly into the gameplay while maintaining engagement poses another hurdle for the project.

1.6 Assumptions

Assumption 1: The hand gesture recognition technology will reliably detect and interpret user gestures with precision during real-time interaction.
Assumption 2: Users will possess access to compatible hardware, such as webcams meeting a minimum resolution requirement of 480p, to facilitate accurate gesture recognition.
Assumption 3: The intended audience will find the interactive sorting and grouping challenges sufficiently engaging and conducive to learning.
Assumption 4: The seamless integration of educational elements within the gameplay mechanics will effectively foster cognitive development and improve learning outcomes.

1.7 Societal / Industrial Relevance

The Gesture-Based Interactive Sorting Game holds relevance for both educational institutions and the broader society. In educational settings, the game can serve as a valuable tool for teachers and educators to enhance classroom learning experiences. By providing interactive and immersive sorting challenges, the game facilitates hands-on exploration of fundamental concepts such as mathematics, language, and problem-solving skills. It offers a dynamic platform for engaging students of all ages and learning abilities, making learning more enjoyable and effective.

Furthermore, beyond the classroom, the Gesture-Based Interactive Sorting Game has applications in community centers, libraries, and other informal learning environments. It can be used to supplement traditional educational approaches and provide an accessible avenue for self-directed learning. Moreover, the inclusivity features of the game, such as gesture-based controls, make it suitable for individuals with physical disabilities, ensuring that everyone can participate in and benefit from the learning experience.

In the industry, the game's immersive and interactive nature can be leveraged for training purposes. For example, companies can use similar technologies to develop simulation-based training programs for employees, enhancing skill acquisition and retention in various fields such as manufacturing, logistics, and healthcare. Additionally, the Gesture-Based Interactive Sorting Game can be adapted for entertainment purposes, offering engaging experiences at interactive exhibits, theme parks, and other recreational venues.

Overall, the project's applicability in educational, community, and industrial settings underscores its potential to make a positive impact on society by fostering learning, skill development, and inclusivity through innovative technology-driven experiences.

1.8 Organization of the Report

Chapter 1 of the report provides a comprehensive introduction to the Gesture-Based Interactive Game project, outlining the background, problem definition, scope, and motivation behind the initiative. It highlights the project's objectives, challenges, and assumptions, emphasizing the integration of gesture recognition technology to create an engaging learning platform. Additionally, the chapter discusses the societal and industrial relevance of the project, noting its potential impact on educational settings, community learning environments, and industry training programs. This foundational information is crucial for understanding the project's aims and guiding its development and implementation.

Chapter 2 outlines the critical technical requirements and system features for the Gesture-Based Interactive Sorting Game. It details hardware and software needs, including memory, graphics, and compatible operating systems, as well as communication interfaces for hand gesture data transmission. Key system features such as hand gesture recognition, GUI design, sorting and grouping objects, and data persistence are described with their functional requirements. The chapter also covers nonfunctional requirements like performance, safety, security, and software quality attributes. These elements are essential for ensuring the software's responsiveness, reliability, and user satisfaction, forming a comprehensive guide for the project's development.

Chapter 3 outlines the system architecture and design for the Gesture-Based Interactive Sorting Game, detailing the input, processing, and output stages. It highlights the use of MediaPipe for gesture recognition and the translation of gestures into game actions. The chapter includes architectural diagrams and identifies key datasets for hand landmark detection. It also presents the proposed algorithms for the game's waste management feature, describing the steps for capturing gestures, detecting movements, and implementing game logic. This concise overview is essential for guiding the project's development and implementation.

Chapter 4 delves into the testing and discussion of GestureQuest, an integration of gesture-based controls into interactive gaming. It highlights the testing stages of three game modules: Aim Trainer, Waste Sort, and NumSort, showcasing their progression from initial to final stages. The Waste Sorting Game notably achieved a high accuracy rate of around 90 percent in gesture recognition, ensuring responsive gameplay. Despite minor issues like performance fluctuations in varying lighting conditions and a limited range of recognized gestures, GestureQuest maintained high accuracy and low latency. These insights inform potential future enhancements, such as improving gesture recognition algorithms and expanding gesture repertoire, to further enrich interactive applications.

Chapter 5 concludes the report by highlighting the transformative impact of the Gesture-Based Interactive Game on both entertainment and education. It emphasizes the game's ability to engage users through intuitive hand gesture recognition and interactive gameplay, bridging the gap between physical and virtual interactions. The chapter also outlines future scope, including introducing progressively challenging tasks, enhancing motivation through rewards and leaderboards, and fostering social interaction. These insights underscore the game's potential for further development and enhancement, ensuring a comprehensive platform for cognitive growth and enjoyment.

Chapter 2

Software Requirements Specification

2.1 Introduction

2.1.1 Purpose

This Software Requirements Specification (SRS) outlines the requirements for the development of the Gesture-Based Interactive Sorting Game. The game aims to combine entertainment and education, fostering cognitive development and a deeper understanding of sorting concepts through an immersive experience using hand gestures.

Gesture recognition systems are said to be the "Finger motion recognition and tracking systems". A finger motion recognition system is rendered to be the most effective means of typing that helps bring in the ideology of "touch-less typing". Finger tracking systems majorly focus on user-data interaction, and aim at having the gestures and hand movements to be more intuitive and creative.

Human society has been using hand gestures for communication since generations. Different hand gestures such as shaking of hands, Thumbs up and Thumbs down have been ever existing in the environment. It is believed that gestures are the easiest way of interacting with anyone. So then why not utilize it to the machines that we are using. In this work, we are demonstrating real-gestures. The initial setup includes a low-cost USB web camera/laptop camera that can be used for providing input to the system. The gesture recognition process is divided into 4 steps which are video-capturing, frame processing, region extraction, and feature matching. Human – Computer interaction became a very huge concern as technology seemed to change all over the years. On the other hand, amateur or aged people find computers hard to use. Overcoming this difficulty stands as the ultimate goal for our proposed system.

2.1.2 Product Scope

The project accommodates individuals with varying levels of computer familiarity promoting an engaging learning platform. The hand gesture recognition technology brought a new era to the artificial intelligence branch of human-computer interaction. This initiative will nevertheless provide a more natural and dynamic form of communication by use of gestures. It will be of significant help in the field of education, also a user-friendly system for physically challenged people. Hand gesture detection systems are one of the hottest fields of research since it is of great significance in designing artificially intelligent human computer interfaces (HCI) for vehicular applications. Emerging technologies now make it possible to create gestural - interfaces for vehicles that make it easier for the driver to carry the tasks effectively. System works extremely well to overcome complications in the real-world including conditions where there can be no space to use a physical mouse button as well as for the particular people who may have difficulties in their palms and are not necessarily able to manage a physical computer mouse.

2.2 Overall Description

2.2.1 Product Perspective

This project merges innovative hand gesture recognition technology with the timeless concept of sorting and grouping games. By leveraging cutting-edge technology, the project aims to enhance user engagement and immersion, offering a unique and intuitive gaming experience. The integration of hand gestures for sorting operations introduces a novel interaction paradigm, catering to both casual gamers and enthusiasts alike. With a focus on accuracy, responsiveness, and accessibility, the project seeks to deliver a seamless and inclusive gaming experience for users of all backgrounds and abilities. Furthermore, the project's scalability and adaptability pave the way for future enhancements and expansions, ensuring its longevity and relevance in the ever-evolving gaming landscape.

2.2.2 Product Functions

- Enable users to sort or group objects based on criteria like color, size, shape, or custom attributes through hand gestures.

- Provide real-time feedback and guidance to users during the sorting process.
- Allow users to group objects together based on shared characteristics or relationships.
- Offer various sorting challenges and game modes to cater to different skill levels and preferences.
- Incorporate accessibility features to ensure usability for users with diverse abilities.
- Focus on accuracy, responsiveness, and user experience to enhance engagement and cognitive development.

2.2.3 Operating Environment

The software operates within an interactive environment conducive to hand gesture recognition and gaming, seamlessly integrating with the Pygame GUI framework for graphical user interface development. It is designed to run on hardware platforms capable of supporting real-time image processing and gesture detection, including desktop computers and laptops equipped with webcams or depth-sensing cameras. The software is compatible with major operating systems including Windows, macOS (versions 10.13 and above), and Linux distributions. It relies on libraries and frameworks for hand gesture recognition, such as MediaPipe, OpenCV, or TensorFlow, which work in tandem with the Pygame GUI components to provide an immersive user experience. Additionally, the software may interface with gaming peripherals or input devices, including keyboards and mice to enhance user interaction and gameplay.

2.2.4 Design and Implementation Constraints

The Hardware limitations, including timing constraints and memory requirements, impact performance and scalability, necessitating optimization efforts. Interfacing with other applications or services requires compatibility with specific APIs and protocols, ensuring seamless integration.

Additionally, developers must adhere to technology mandates, utilizing designated tools, frameworks, and databases as per project standards. Considerations such as parallel operations and language constraints further shape development decisions, affecting scalability and maintainability.

Finally, maintaining consistency with design conventions and programming standards ensures readability, maintainability, and long-term sustainability. Despite these constraints, developers navigate them skillfully to deliver a robust, reliable, and compliant software solution that meets stakeholder expectations.

2.2.5 Assumptions and Dependencies

Assumptions and dependencies impacting the SRS requirements include: potential third-party components for hand gesture recognition, assumed compatibility with existing hardware and operating systems, and reliance on stable communication protocols. The project's success hinges on accurate assumptions and shared understanding of these factors, as deviations could affect implementation and functionality. Additionally, dependencies on external factors like software components from other projects may influence development timelines and resource allocation. Regular validation of assumptions and proactive management of dependencies are critical to mitigate risks and ensure project success.

2.3 External Interface Requirements

2.3.1 User Interfaces

The software product interfaces with users through a graphical user interface (GUI), adhering to standard GUI principles and design conventions. This includes intuitive screen layouts, consistent button placement, and standard functions such as help and settings accessible across all screens. Keyboard shortcuts and error message display standards are implemented for efficient user interaction and error handling. The user interface is designed to accommodate the software components requiring user interaction, ensuring a seamless and user-friendly experience. Detailed specifications of the user interface design are documented separately in the user interface specification.

2.3.2 Hardware Interfaces

The minimum hardware requirements for this project are:
Input Devices: The software requires a webcam or depth-sensing camera with a minimum resolution of 480p for capturing hand gestures.
Processor: A single-core processor running at a clock speed of 1.5 GHz or higher is sufficient for basic operation of the software.

Memory (RAM): A minimum of 2 GB of RAM is required to run the software smoothly, ensuring adequate memory allocation for processing hand gesture data.
Graphics Card: The software can run on integrated graphics or older dedicated graphics cards, with support for OpenGL 2.0 or higher.
Storage: A minimum of 200 MB of available storage space is required for installing the software.
Operating System: The software is compatible with Windows 7, macOS 10.12, or Linux distributions (Ubuntu 18.04 LTS or equivalent) for basic functionality.

2.3.3 Software Interfaces

The software requirements for this project are:
Operating system : Windows 7/8/10
Coding Language : Python
Tools : PyCharm , VS Code
Libraries : OpenCV, Media pipe,TensorFlow,PyGame
Front end : PyGame
Backend : Python 3.8-3.9

2.3.4 Communication Interfaces

1. Hand Gesture Data Transmission: Utilize standard protocols (USB, Bluetooth) for capturing and transmitting hand gesture data from input devices to the software.
2. Data Synchronization: To ensure data consistency across multiple instances of the software.
3. Data Transfer Rates: Support sufficient data transfer rates to ensure smooth operation, especially for real-time applications like hand gesture recognition or multiplayer gaming.
4. Message Formatting: Adhere to standardized message formats (JSON, XML) for structured data exchange to ensure compatibility and interoperability.

2.4 System Features

2.4.1 Hand Gesture Recognition

This feature involves detecting and interpreting hand gestures captured from input devices like webcams or depth-sensing cameras. It is of high priority as it forms the core interaction mechanism of the software. Stimulus includes hand gestures captured by input devices, while the response involves interpreting these gestures and translating them into corresponding actions within the software interface.

Functional Requirements

REQ-1: The system shall process incoming video frames from input devices in real-time.
REQ-2: It shall employ algorithms for hand detection and tracking to identify gestures accurately.
REQ-3: The software shall map recognized gestures to predefined actions within the interface, such as selecting or moving objects.

2.4.2 Graphical User Interface (GUI)

This feature provides an intuitive and interactive GUI for user interaction. It is of high priority as it directly impacts user experience and usability.

Functional Requirements

REQ-1: The GUI shall include elements such as buttons, menus, and visual feedback components.
REQ-2: It shall adapt to different screen resolutions and aspect ratios for seamless display across devices.
REQ-3: The software shall respond to user interactions by updating GUI elements and providing appropriate feedback.

2.4.3 Sorting and Grouping Objects:

This feature enables users to sort and group objects based on various criteria. It is of medium priority as it enhances user engagement and gameplay depth.Stimulus involves user commands to sort or group objects, while the response includes rearranging the objects on the interface based on the specified criteria.

Functional Requirements

REQ-1: Users shall be able to interact with objects using hand gestures initiate sorting or grouping actions.
REQ-2: The system shall update the display to reflect the rearranged order or grouping of objects in real-time.
REQ-3: It shall provide visual cues or indicators to assist users in understanding the sorting or grouping criteria being applied.

2.4.4 Data Persistence:

This feature ensures that user preferences, settings, and game progress are saved across sessions. It is of high priority as it enhances user retention and provides a seamless experience. Stimulus includes user actions that trigger changes in data (e.g., adjusting settings, completing game levels), while the response involves storing the updated data for future retrieval.

Functional Requirements

REQ-1: User preferences and settings shall be saved automatically upon modification and loaded upon software startup.
REQ-2: Game progress, including completed levels and high scores, shall be saved periodically during gameplay and restored upon subsequent sessions.
REQ-3: The system shall provide options for data backup and restoration to prevent loss of user data in case of system failures or device changes.

2.5 Other Nonfunctional Requirements

2.5.1 Performance Requirements

Performance requirements for the product are essential to ensure smooth operation and responsiveness, particularly in real-time systems such as hand gesture recognition. Here are the performance requirements specified for individual functional requirements:

1. Hand Gesture Recognition:
The system shall process hand gesture data with a latency of no more than 100 milliseconds. Low latency is crucial for real-time responsiveness, ensuring that users experience minimal delay between performing a hand gesture and seeing the corresponding action executed in the software interface.

2. Graphical User Interface (GUI):
GUI elements shall render with a frame rate of at least 30 frames per second (FPS).Smooth GUI animation and transitions enhance user experience and perception of responsiveness. Achieving a frame rate of 30 FPS or higher ensures visually fluid interaction with the interface.

3. Sorting and Grouping Objects:
Sorting and grouping operations shall complete within 500 milliseconds for datasets of up to 100 objects. Efficient processing of sorting and grouping actions ensures that users can interact with the software interface without experiencing delays or sluggishness, even when handling large datasets.

4. Game Modes and Challenges:
Game mode transitions and level loading shall occur within 2 seconds.Swift transitions between game modes and levels maintain momentum and engagement for the user. Delays exceeding 2 seconds may disrupt immersion and lead to user frustration.

5. Data Persistence:
Saving and loading user data shall occur within 1 second. Quick data persistence oper-

ations ensure a seamless user experience during startup and shutdown of the software. Users expect minimal delay when accessing their preferences, settings, and game progress. These performance requirements aim to provide clear benchmarks for developers and guide design choices to meet user expectations for responsiveness and efficiency. Adhering to these requirements ensures that the software delivers a smooth and enjoyable user experience across various usage scenarios.

2.5.2 Security Requirements

1. Data Protection: Ensure the security and integrity of user data through encryption and access controls.
2. User Privacy: Respect user privacy by minimizing data collection and obtaining explicit consent.
3. Authentication and Authorization: Securely authenticate users and enforce access controls to protect against unauthorized access.
4. Security Testing and Monitoring: Regularly assess and monitor the product's security posture to detect and mitigate vulnerabilities.
5. Compliance with Regulations: Adhere to relevant privacy regulations and industry standards to protect user rights and ensure legal compliance.

2.5.3 Software Quality Attributes

1. Usability: Achieve a high usability score on standardized tests to ensure ease of use and user satisfaction.
2. Reliability: Maintain a low mean time between failures (MTBF) to minimize system downtime and ensure consistent performance.
3. Maintainability: Keep code complexity low to facilitate ease of maintenance and future enhancements.
4. Portability: Ensure compatibility with major operating systems to reach a wider user base and enhance accessibility.
5. Testability: Achieve high code coverage in automated tests to detect and fix issues early in the development process.

6. Performance: Optimize the system to meet performance benchmarks, ensuring smooth operation, low latency, and fast response times, particularly in real-time applications like gesture recognition.

7. Scalability: Design the system to handle increased load and complexity without degradation in performance, supporting more users and additional features over time.

Chapter 3

System Architecture and Design

3.1 System Overview

Let's delve deeper into the architecture of the webcam gesture recognition game, focusing on each stage and potential complexities:

Input: Webcam

Resolution and Frame Rate: The game needs to specify the desired webcam resolution (e.g., 640x480) and frame rate (e.g., 30 FPS) for optimal performance. Higher resolutions and frame rates provide more detail for gesture recognition but require more processing power.

Processing: Gesture Recognition with Mediapipe

Preprocessing: The raw video frames from the webcam might require preprocessing before feeding them to MediaPipe. This could involve noise reduction, color space conversion (e.g., converting from RGB to grayscale), or background subtraction to isolate the user's hand from the background.

Landmark Detection: MediaPipe's hand landmarking model analyzes each frame to identify and localize specific keypoints on the hand. These keypoints are typically represented as 21 (or more) 3D coordinates on the hand skeleton.

Post-Processing: The raw keypoint data might require further processing. Smoothing algorithms can be applied to reduce jittery hand movements caused by slight tremors or frame rate inconsistencies.

Gesture Classification: Depending on the game's complexity, the system might need to classify specific gestures beyond just hand pose detection. This could involve analyzing sequences of hand poses over multiple frames to identify gestures like swipes, pinches, or finger curls. Here, machine learning models trained on specific gesture datasets might be employed.

Game Logic: Interpreting Hand Gestures

Mapping Hand Poses to Actions: The game logic translates the hand pose information (e.g., keypoint locations or identified gestures) into meaningful actions within the game world. For instance, a closed fist gesture might translate to a punch attack in a fighting game, while an open palm facing forward might trigger a character to jump.

Output: Game Loop and User Feedback

Rendering: The game engine renders the game world based on the updated game state and user input from gestures. This involves updating the positions and animations of characters, objects, and the user's avatar (if applicable).

Score Calculation and Display: The game logic calculates the user's score based on their performance within the defined rules. This score is then updated and displayed on the screen, providing feedback to the user.

Win/Loss Conditions and Exiting the Game: The game logic continuously evaluates the game state and user performance. When win/loss conditions are met (e.g., reaching a specific score, completing a level) or exit conditions are triggered (e.g., user chooses to quit), the game transitions to a "game over" state. This state might involve displaying a game over screen, summarizing the user's score, and offering options to restart or exit the game.

Additional Considerations:

Real-time Performance: Processing video frames, recognizing gestures, and updating the game state needs to happen in real-time for a smooth and responsive gameplay experience. Optimizing algorithms and utilizing hardware acceleration techniques might be necessary.

Lighting and Background Conditions: Variations in lighting and background clutter can affect the accuracy of gesture recognition. The game might provide recommendations for optimal lighting conditions or implement background subtraction techniques to mitigate these challenges.

3.2 Architectural Design

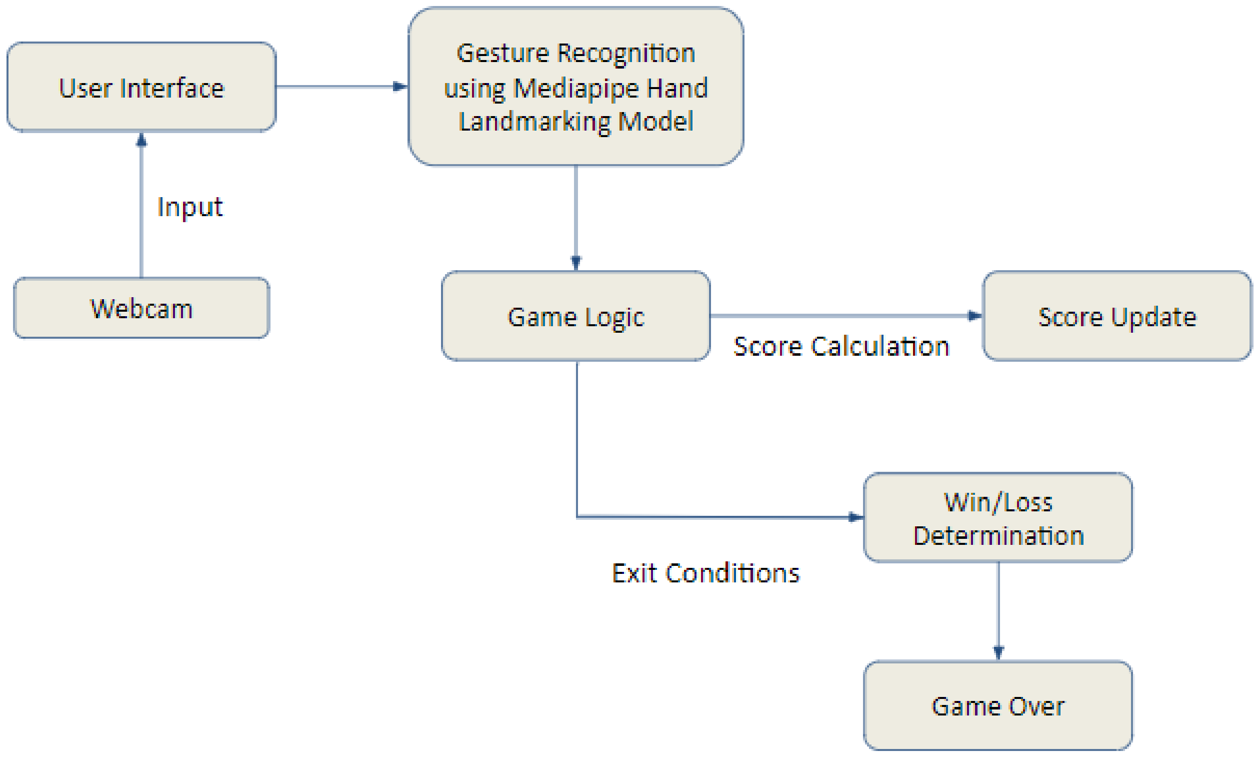


Figure 3.1: Architectural Diagram

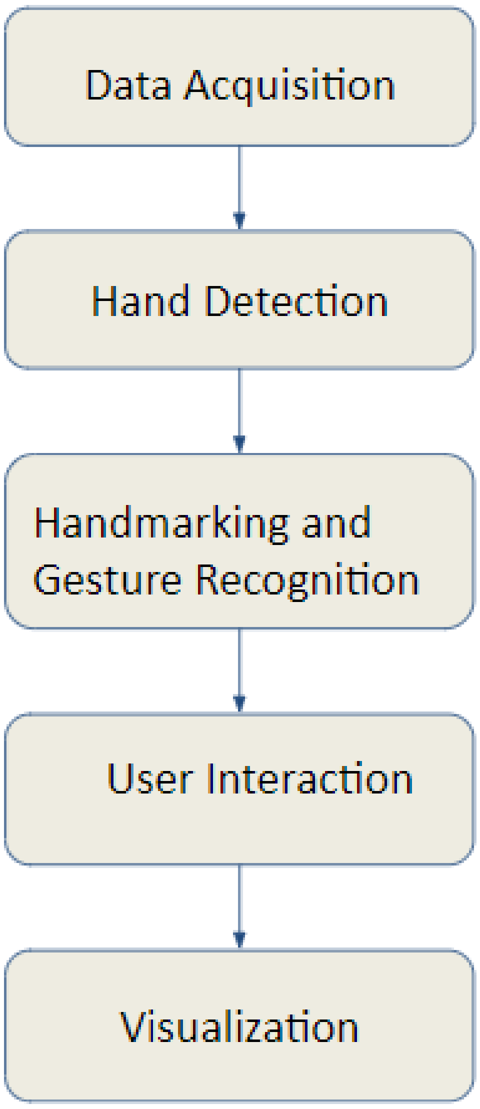


Figure 3.2: Workflow

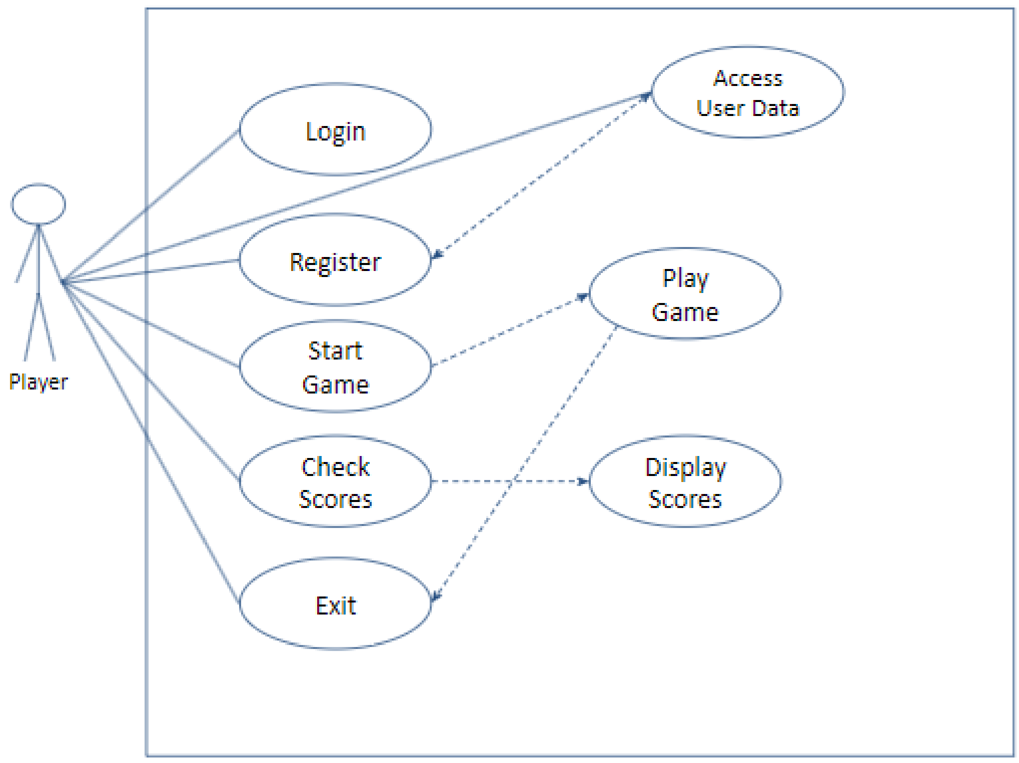


Figure 3.3: Use Case Diagram

3.3 Dataset identified

Hand landmarks detection guide: The MediaPipe Hand Landmarker task lets you detect the landmarks of the hands in an image. You can use this task to locate key points of hands and render visual effects on them. This task operates on image data with a machine learning (ML) model as static data or a continuous stream and outputs hand landmarks in image coordinates, hand landmarks in world coordinates and handedness(left/right hand) of multiple detected hands. Get Started Start using this task by following one of these implementation guides for your target platform. These platform-specific guides walk you through a basic implementation of this task, including a recommended model, and code example with recommended configuration options:

- Android - Code example - Guide
- Python - Code example - Guide
- Web - Code example - Guide

The hand landmark model bundle detects the keypoint localization of 21 hand-knuckle

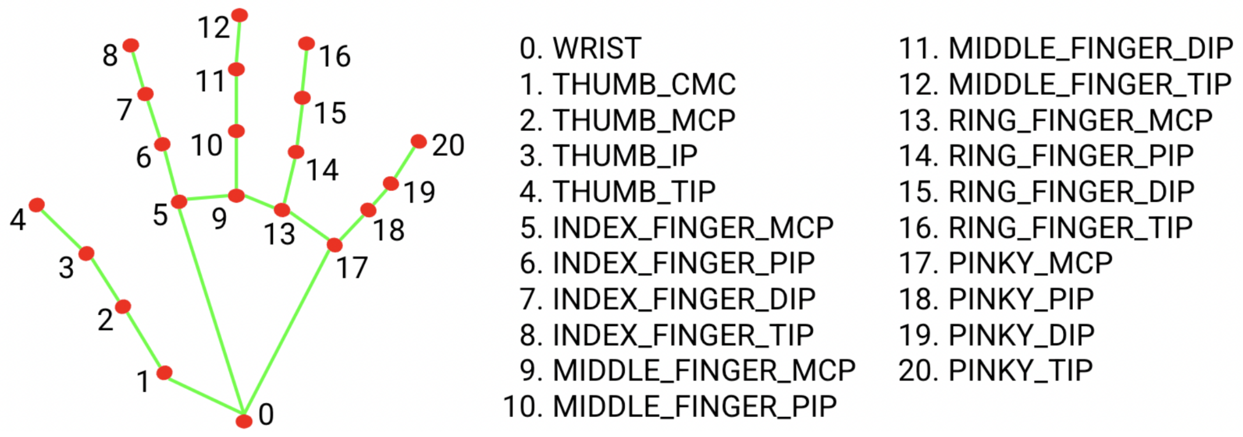


Figure 3.4: Handlandmarks

coordinates within the detected hand regions. The model was trained on approximately 30K real-world images, as well as several rendered synthetic hand models imposed over various backgrounds.

The hand landmarker model bundle contains a palm detection model and a hand landmarks detection model. The Palm detection model locates hands within the input image, and the hand landmarks detection model identifies specific hand landmarks on the cropped hand image defined by the palm detection model. Since running the palm detection model is time consuming, when in video or live stream running mode, Hand Landmarker uses the bounding box defined by the hand landmarks model in one frame to localize the region of hands for subsequent frames. Hand Landmarker only re-triggers the palm detection model if the hand landmarks model no longer identifies the presence of hands or fails to track the hands within the frame. This reduces the number of times Hand Landmarker tiggers the palm detection model.
Task benchmarks
Here's the task benchmarks for the whole pipeline based on the above pre-trained models. The latency result is the average latency on Pixel 6 using CPU / GPU. Model Name CPU Latency GPU Latency HandLandmarker (full) 17.12ms 12.27ms

3.4 Proposed Methodology/Algorithms

Algorithm for waste management

1. Create objects (waste items and bins) with positions and types.
2. Initialize variables like score, pointer position, and game state.
3. Capture video frame.
4. Detect hand landmarks in the frame (using MediaPipe).
5. Track finger movement and handle object selection:
6. If hand is pinching:
7. Get fingertip location.
8. Smooth finger movement for a natural feel.
9. Update object position based on smoothed finger movement.
10. Check if object collides with any other object (mainly other waste items).
11. If object collides with designated bin:
12. Check if object type matches the bin type (organic/plastic).
13. If correct type, add points to score.
14. If wrong type, deduct points.
15. Draw on screen:
16. Fill background with a color.
17. Draw a green circle at the fingertip location.
18. Draw all objects (waste items and bins) on their assigned positions.
19. Check for submit button press:
20. If pressed, calculate final score based on remaining time and enter game over state.

21. Otherwise, continue running the game loop.
22. Display "GAME OVER" and final score.
23. Check for button presses:
24. Replay button: Reset game by recreating objects and variables.
25. Exit button: Go back to the menu state.

Algorithm for numsort

1. Load images for game assets (submit, numbers, restart, exit, grey).
2. Initialize MediaPipe hands for gesture recognition.
3. Start webcam capture.
4. Set initial positions and velocities for the pointer.
5. Initialize fonts and buttons.
6. Randomly place Things objects and create slots.
7. Set initial game state flags.
8. Reset the game state if starting.
9. Capture and process a webcam frame.
10. Detect hands and update pointer position using smoothing.
11. Check for pinching gesture to move Things objects.
12. Render the webcam frame, slots, Things objects, and pointer.
13. Check if the submit button is pressed to transition to game over state.
14. Check win condition by comparing positions of Things with slots.
15. Calculate and display score.
16. Render win/loss text and restart/exit buttons.

17. Reset the game if restart is pressed or switch to menu if exit is pressed.
18. Clear and reinitialize Things and slots.
19. Reset the score and reinitialize the Numsort class.

Algorithm for aimtrainer

1. Set screen dimensions, initialize clock, and set colors.
2. Initialize MediaPipe hands for gesture recognition.
3. Capture video from webcam and set frame dimensions.
4. Set initial pointer position, smoothing factor, and target position.
5. Load fonts for displaying score and timer.
6. Load images for restart and exit buttons and create button instances.
7. Set initial values for score, timer, pinching state, and game state flags.
8. Capture and process a frame from the webcam.
9. Detect hand landmarks and update pointer position using smoothing.
10. Check for pinching gesture to determine if the target is hit.
11. Update target position and score if hit.
12. Clear the screen and draw the target and pointer.
13. Display the score and remaining time.
14. Check if time limit is reached and handle game over state.
15. Show final score and options to restart or exit the game.
16. Update Method:
17. Clear the screen.
18. Draw the target and pointer.

19. Display the current score and remaining time.

20. Check if the game is over and display final score if time is up.

21. Show Final Score Method:

22. Clear the screen.

23. Display the final score text.

24. Restart Game Method:

25. Reset the game state to start a new game.

26. Reset score, start time, target position, and game over flag.

3.5 User Interface Design

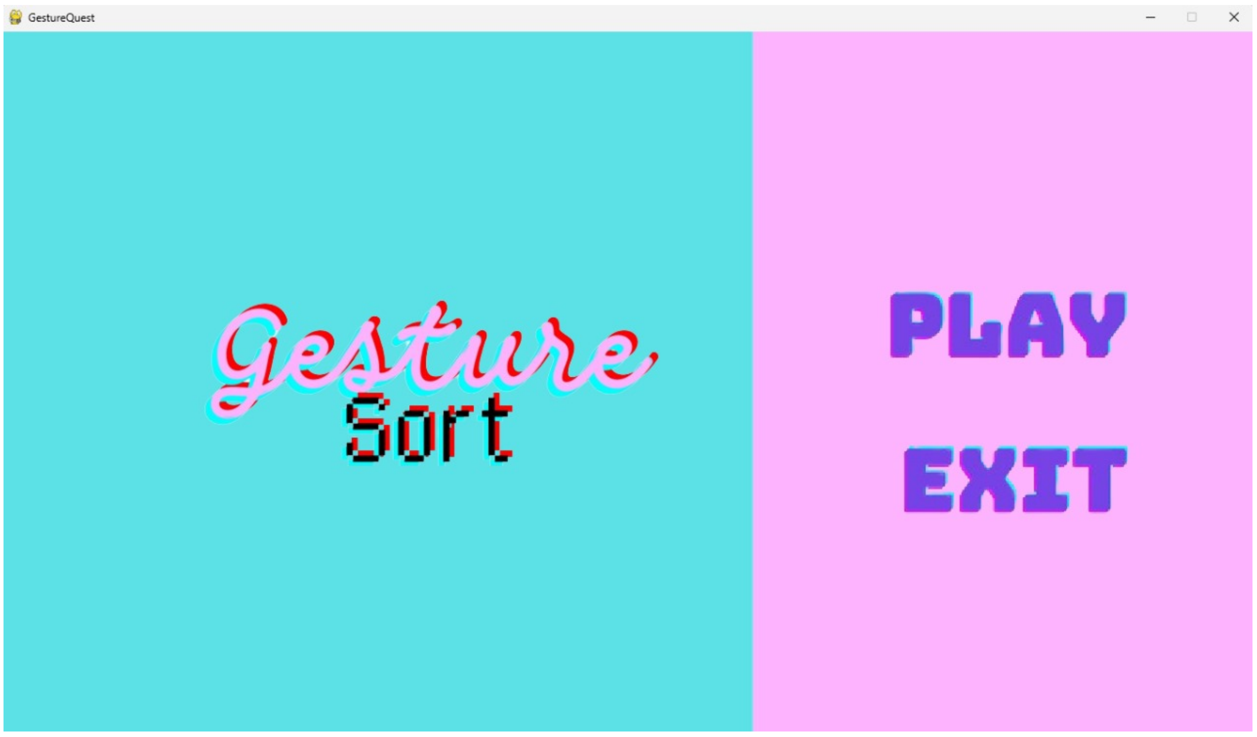


Figure 3.5: Gesture sort UI

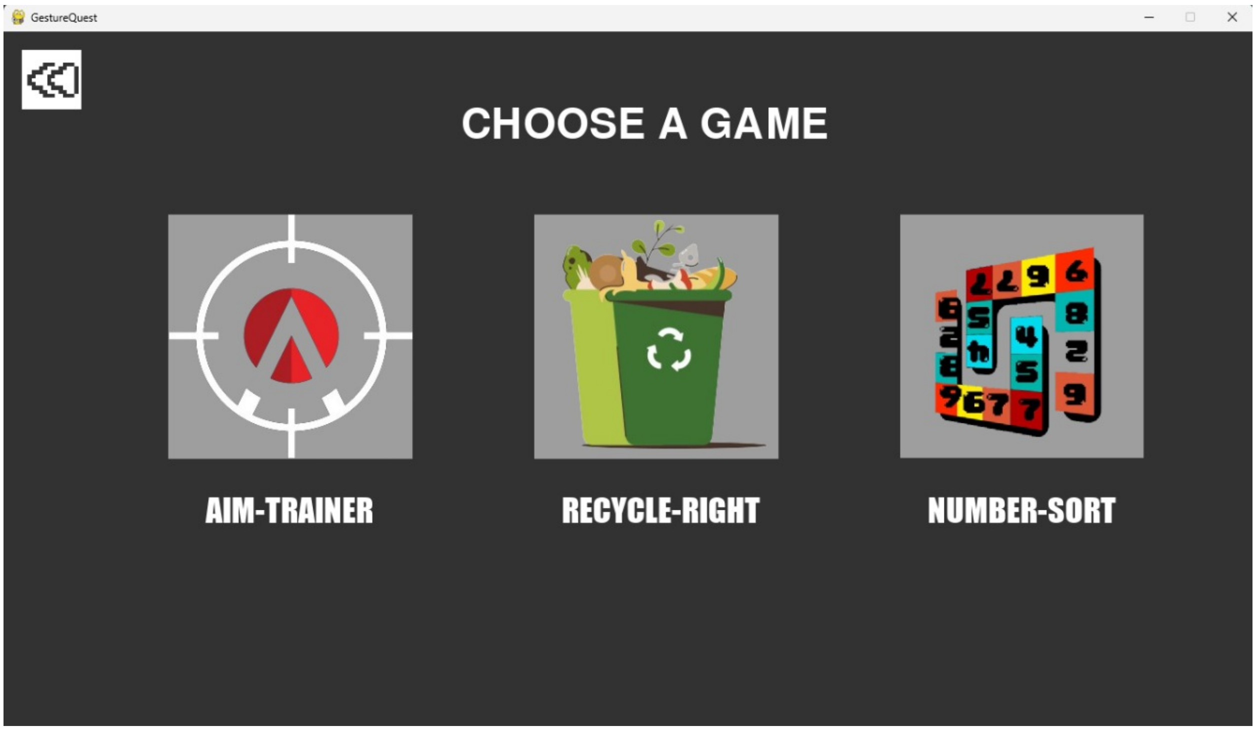


Figure 3.6: Main menu

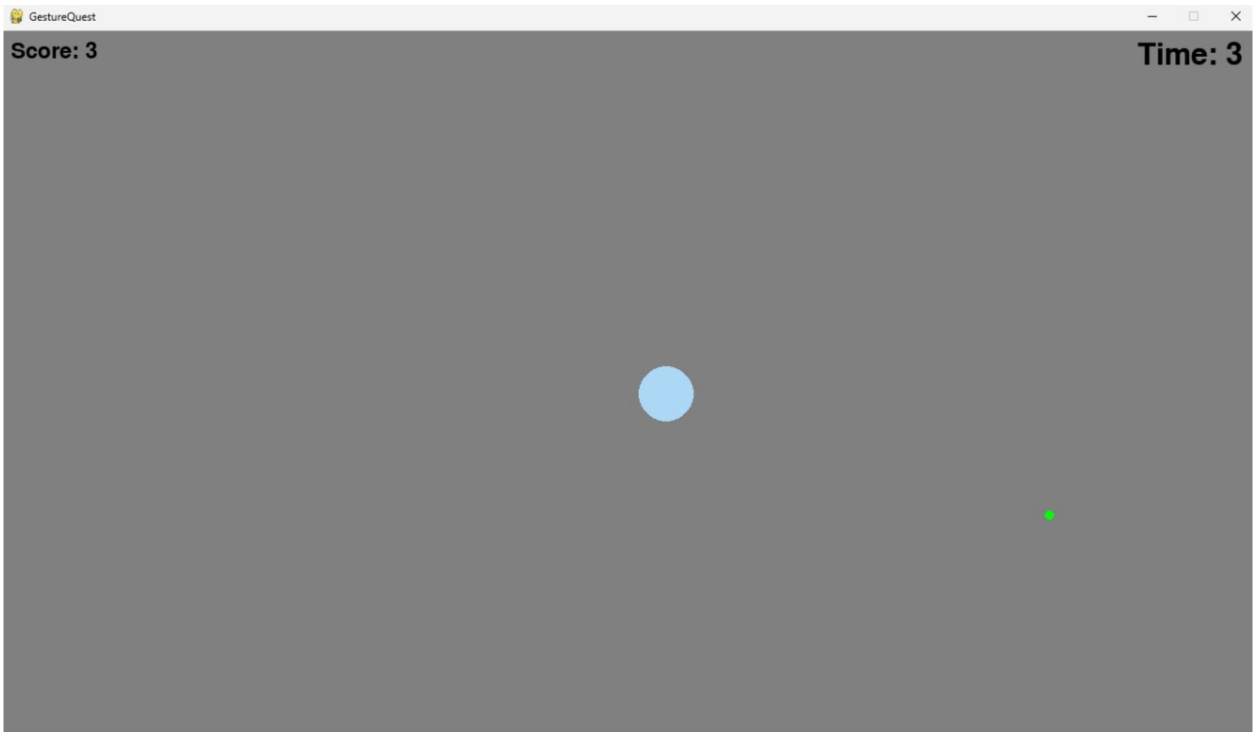


Figure 3.7: Aim Trainer

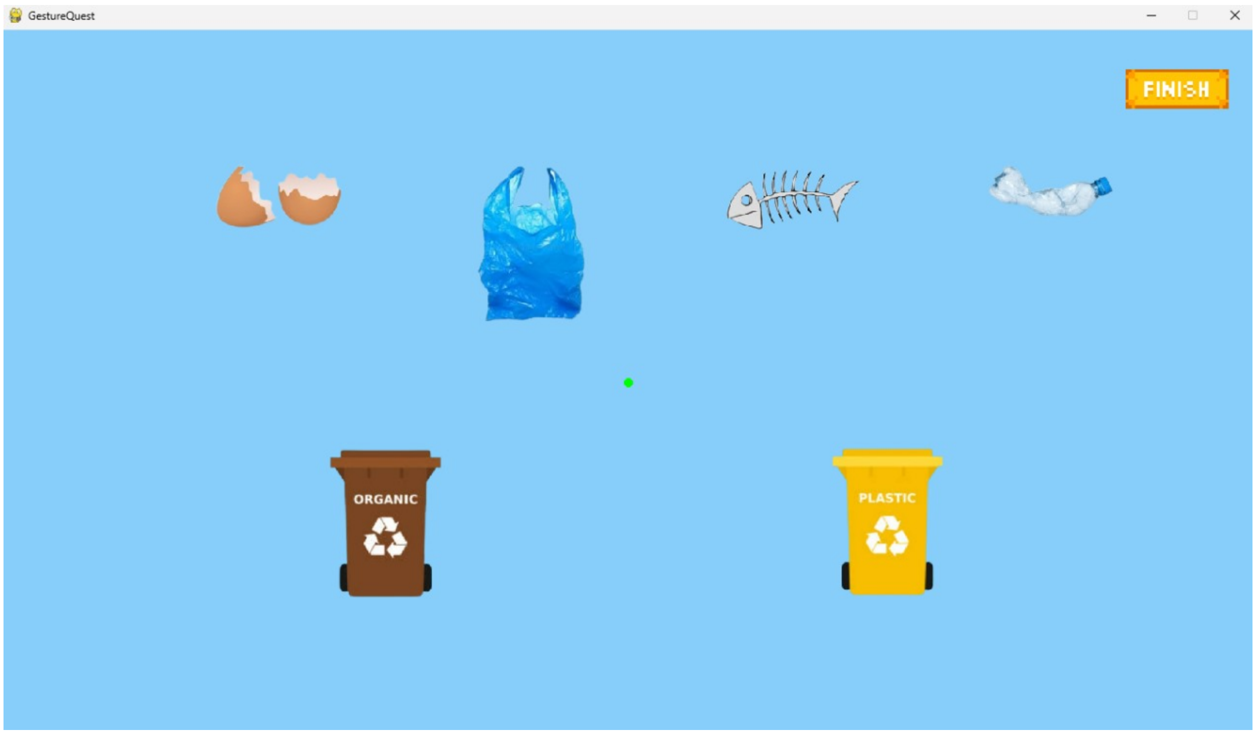


Figure 3.8: Waste sort

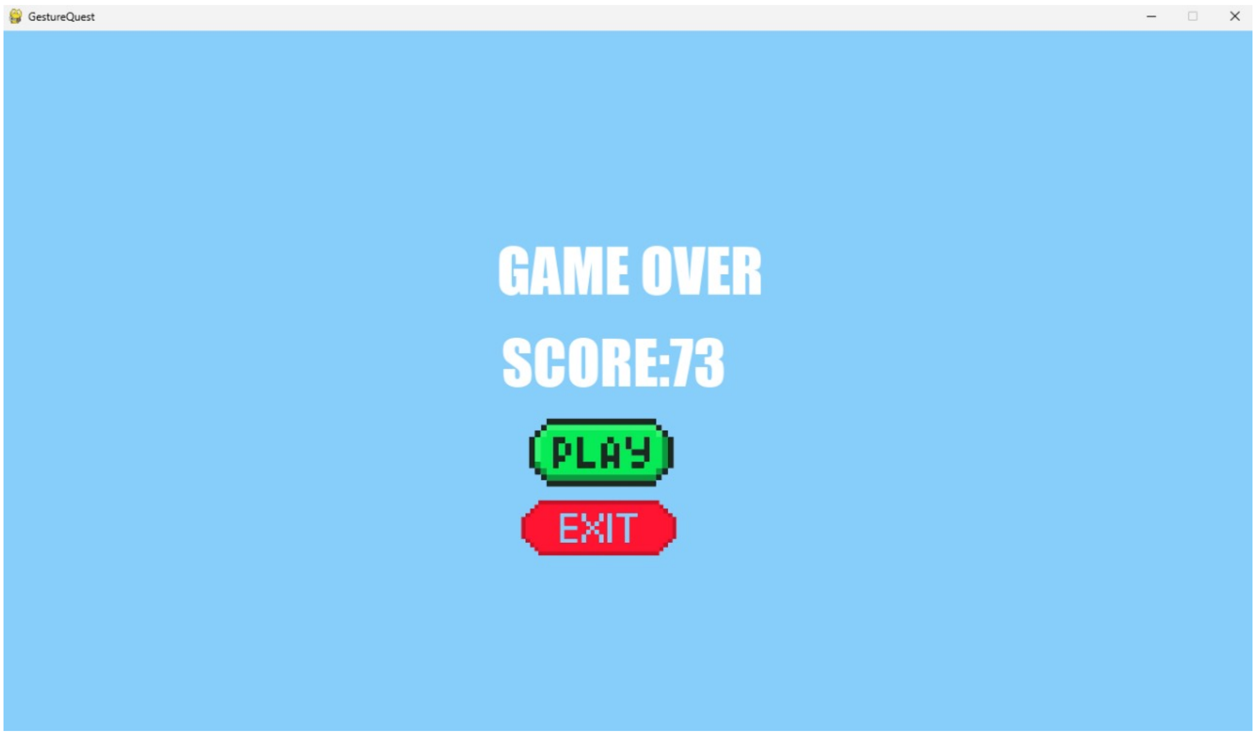


Figure 3.9: Waste sort score

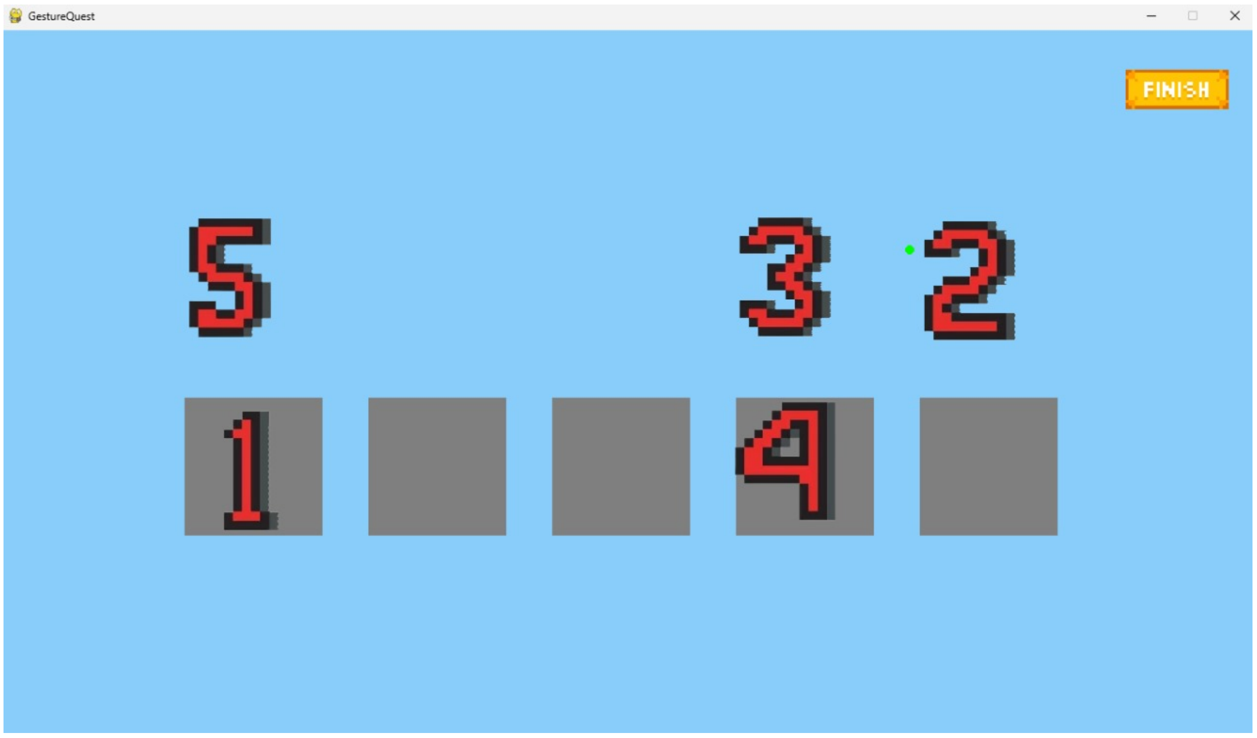


Figure 3.10: Numsort

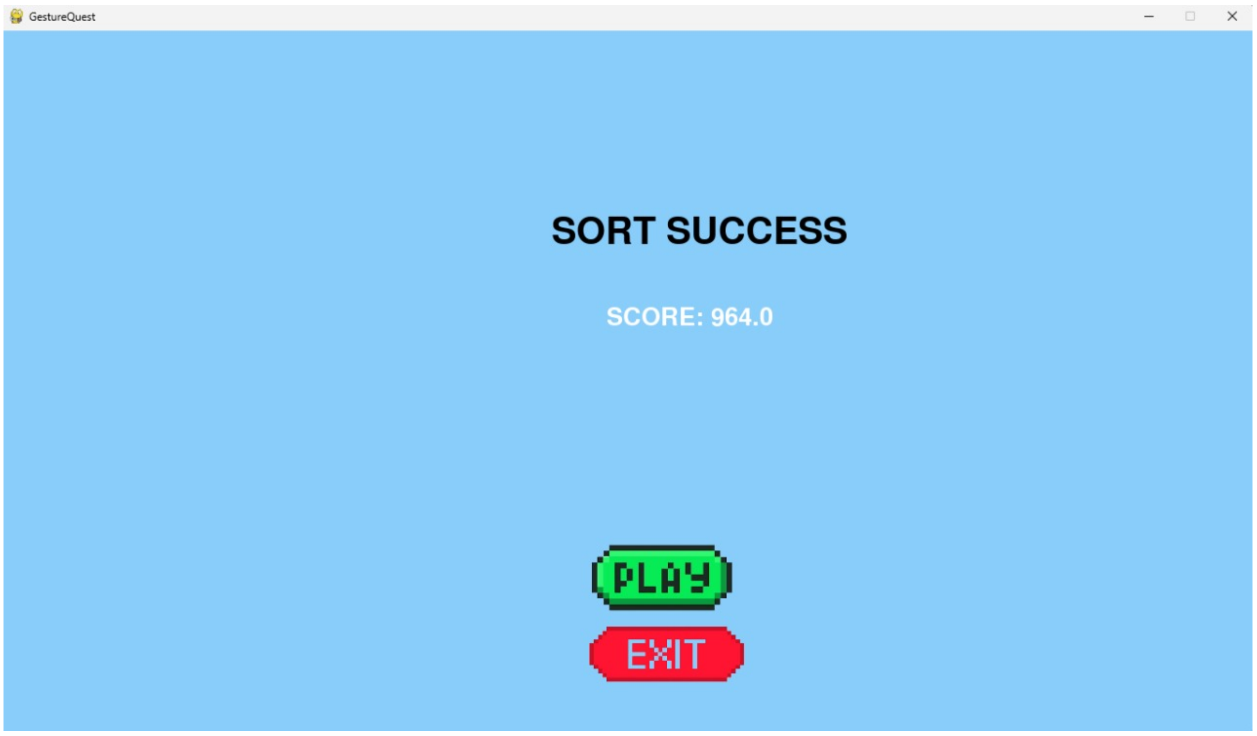


Figure 3.11: Numsort score

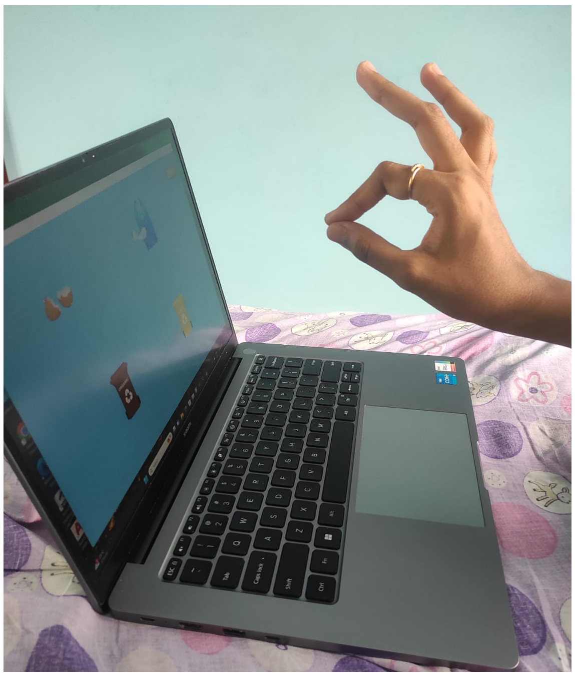


Figure 3.12: Pinching Hand Gesture

3.6 Description of Implementation Strategies

1.Video Capture using OpenCV:

- OpenCV library (‘cv2‘) is used for video capture from a webcam.
- The ‘VideoCapture‘ class is used to access the webcam feed, and you’ve set properties like frame height and width.

2.Hand Gesture Recognition using MediaPipe:

- Hand gesture recognition is implemented using the MediaPipe library (‘mediapipe‘).
- Used the ‘Hands‘ class from ‘mediapipe.solutions.hands‘ for detecting hand landmarks.
- The ‘drawing_utils‘ module is used to visualize the hand landmarks in the video feed.

3.Object Detection and Interaction:

- Implemented object detection and interaction based on hand gestures.
- Pinching gesture detection is used for object selection.
- The position of the index finger tip is tracked to interact with objects in the game.

4.Game State Management:

- You’ve implemented a game state manager (‘GameStateManager‘) to manage different states of the game (e.g., menu, game selection, individual games).
- The state manager allows for switching between different game states seamlessly.

5.Button Interaction:

- Buttons are implemented using the 'Button' class, allowing for user interaction with the game interface.
- Buttons change appearance when hovered over or clicked, providing visual feedback to the user.

6.Rendering and User Interface:

- Pygame is used for rendering graphics and creating the user interface.
- Different game elements, such as buttons, game objects, and text, are rendered on the screen using Pygame's rendering functions.

These strategies provide a solid foundation for building interactive games with hand gesture recognition and user interface interaction.

3.7 Module Division

Module 1: Data Acquisition

The Data Acquisition module is responsible for capturing a live video feed from the user's camera. This module sets up the connection to the camera hardware, initializes the video stream, and continuously captures video frames in real-time. These frames serve as the raw input data for subsequent processing. The primary objective is to ensure a steady and reliable video stream that can be analyzed for hand movements and gestures without significant lag or interruptions.

Module 2: Hand Detection Module

The Hand Detection module leverages the MediaPipe Hands framework to identify and locate hand landmarks within the captured video frames. MediaPipe Hands is a robust and efficient machine learning solution that can detect up to 21 distinct hand landmarks, such as finger joints and the palm base, in real-time. This module processes each frame, detects the presence of hands, and marks key points on the detected hands. These landmarks are crucial for understanding the hand's position, orientation, and shape, forming the basis for gesture recognition.

Module 3: Gesture Recognition Module

The Gesture Recognition module is tasked with analyzing the hand landmarks identified by the Hand Detection module to recognize predefined gestures. By examining the relative positions and movements of the landmarks, this module can distinguish between different gestures, such as swipes, pinches, and rotations. The recognition process involves comparing the detected landmark configurations against a set of predefined templates or using machine learning models trained to identify specific gestures.

Module 4: User Interface (Pygame)

The User Interface module utilizes Pygame, a popular library for creating multimedia applications, to render the graphical user interface (GUI) for the system. This module is responsible for visualizing the live video feed, overlaying detected hand landmarks, and providing feedback to the user based on recognized gestures. It also manages the overall game environment, displaying game elements, scores, and instructions to the player. The UI aims to offer a clear, engaging, and interactive experience, making it easy for users to understand their interactions and the system's responses.

3.8 Work Schedule - Gantt Chart

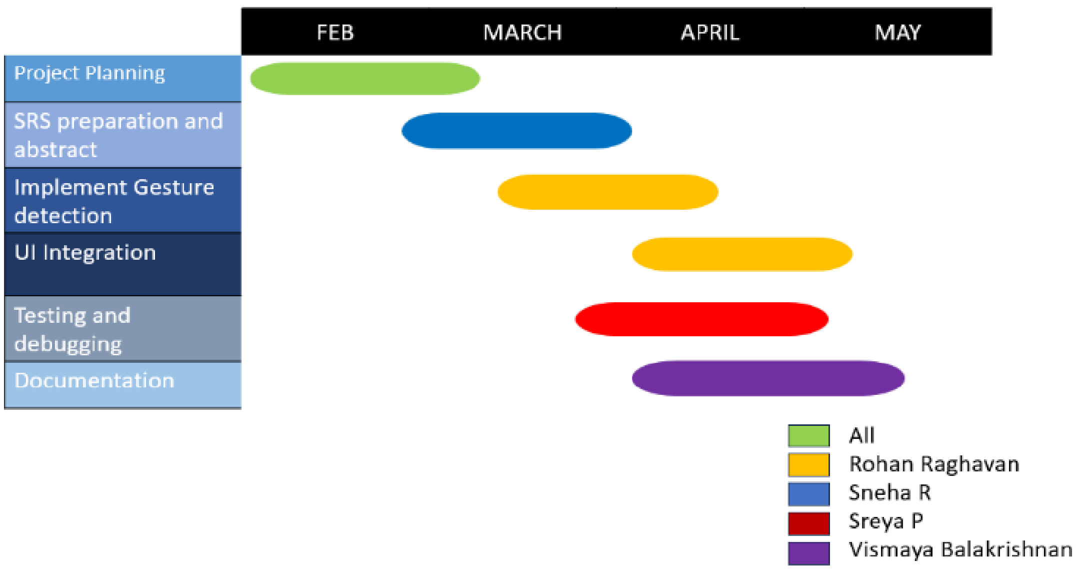


Figure 3.13: Gantt chart

Chapter 4

Results and Discussions

4.1 Overview

GestureQuest is an innovative project that created an interactive game using gesture-based controls, leveraging computer vision techniques for hand tracking and gesture recognition. Developed with Python, Pygame, and Mediapipe, the game enables players to interact through hand gestures without traditional input devices.

A key component, the Waste Sorting Game, allows players to sort waste into recycling bins using hand movements. The system accurately tracks gestures in real-time, converting them into game actions like grabbing and dropping objects.

The project achieved a high gesture recognition accuracy of approximately 90 percent, ensuring precise and responsive interactions. Players experienced seamless gameplay, with gesture inputs correctly translated into game actions. The game maintained a low latency of under 100 milliseconds and a stable frame rate of 60 frames per second, contributing to a smooth and immersive experience.

Quantitative results highlighted the system's robustness under controlled lighting, though performance slightly declined in varying lighting conditions. Future improvements could focus on enhancing this robustness and expanding the range of recognized gestures.

Overall, GestureQuest demonstrated the feasibility and excitement of gesture-based gaming, offering an engaging and intuitive user experience. The successful integration of Pygame and Mediapipe points to significant potential for future developments in interactive applications using gesture recognition.

4.2 Testing

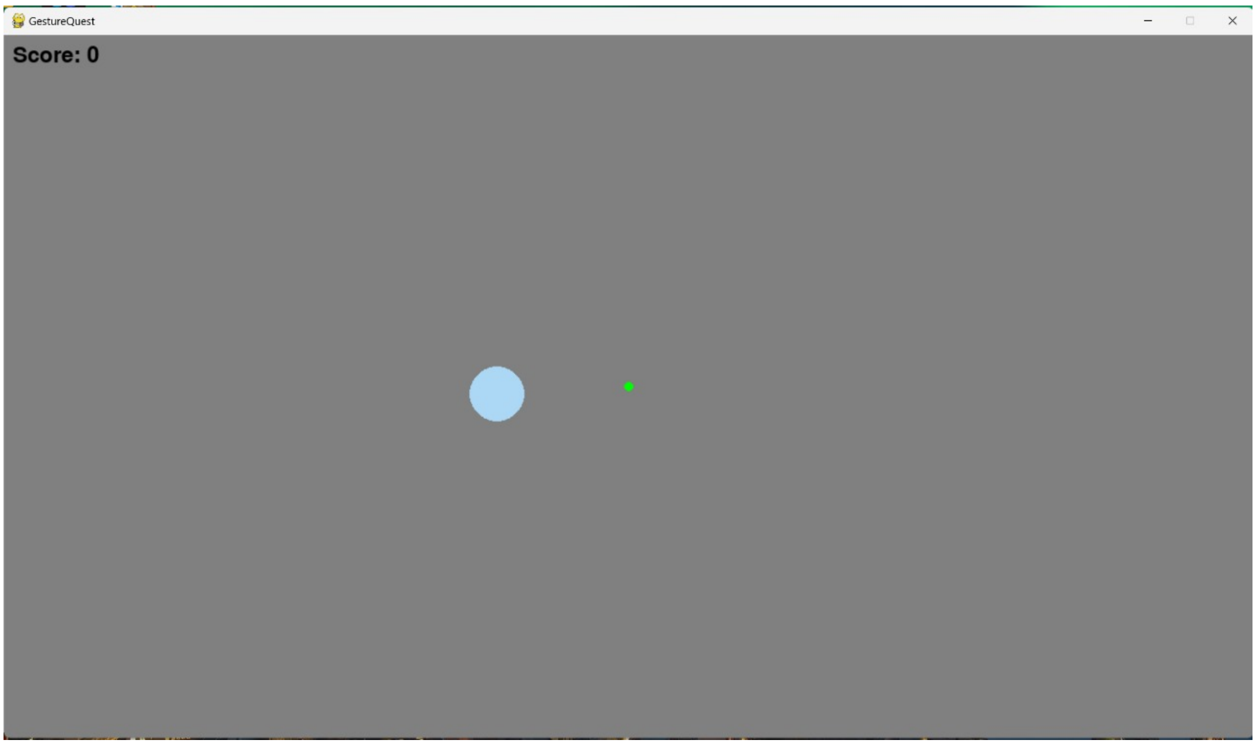


Figure 4.1: Aim Trainer Initial Stage

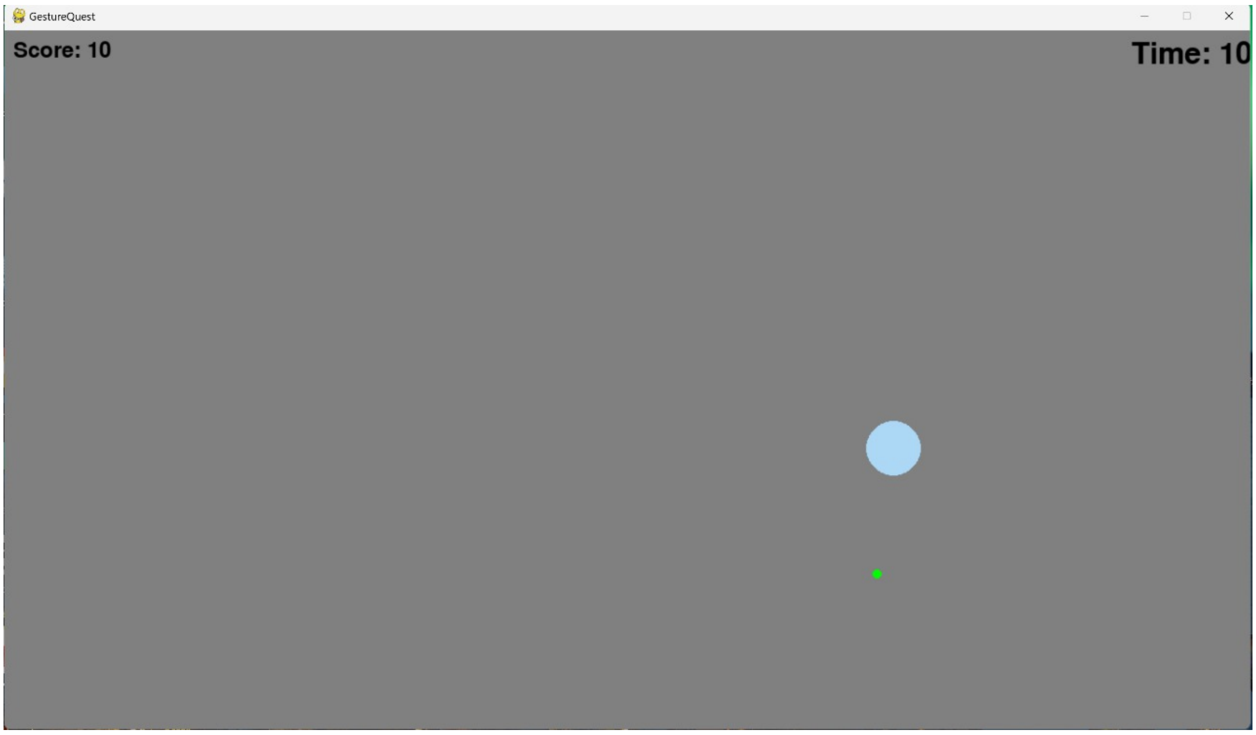


Figure 4.2: Aim Trainer Intermediate Stage

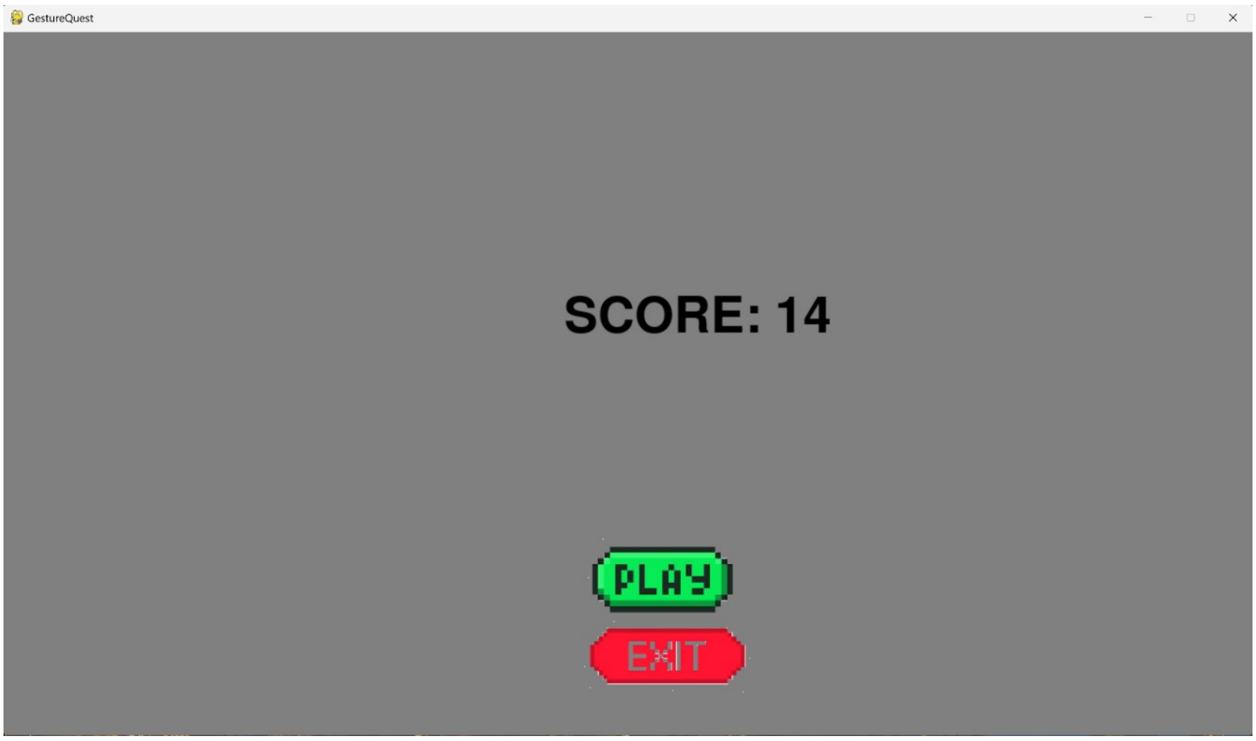


Figure 4.3: Aim Trainer Final Stage

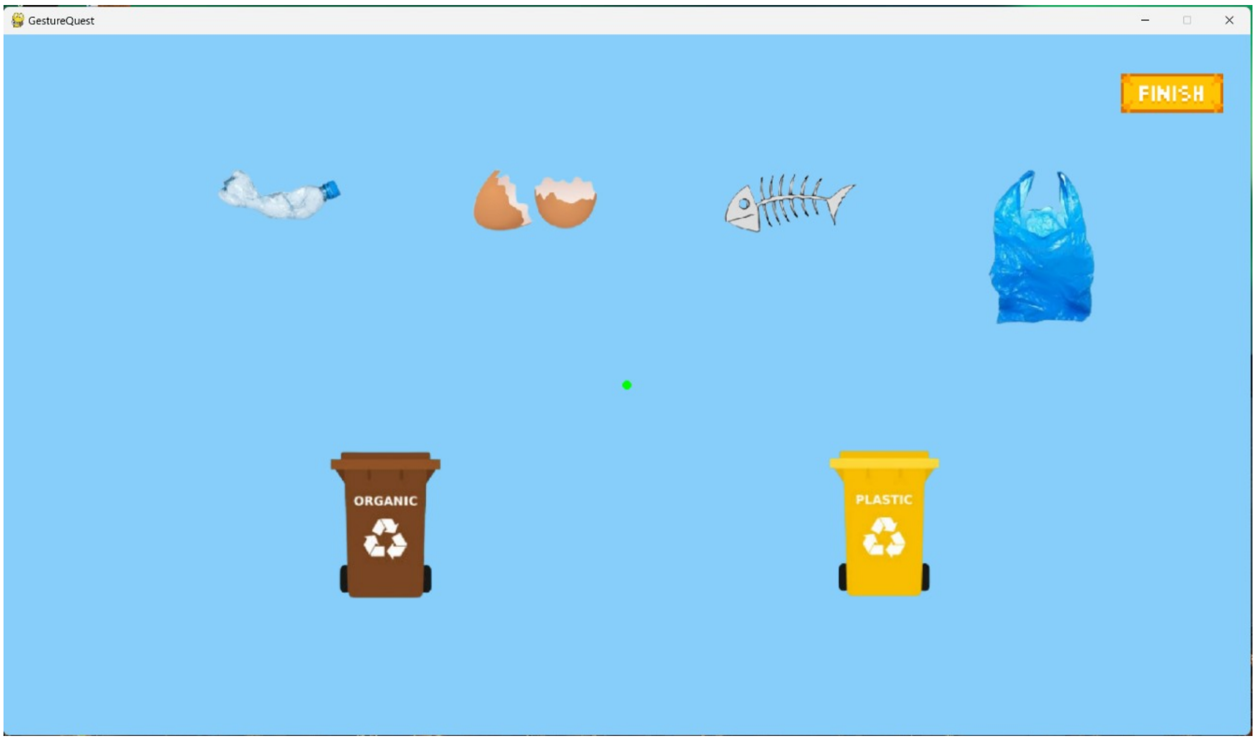


Figure 4.4: Waste Sort Initial Stage

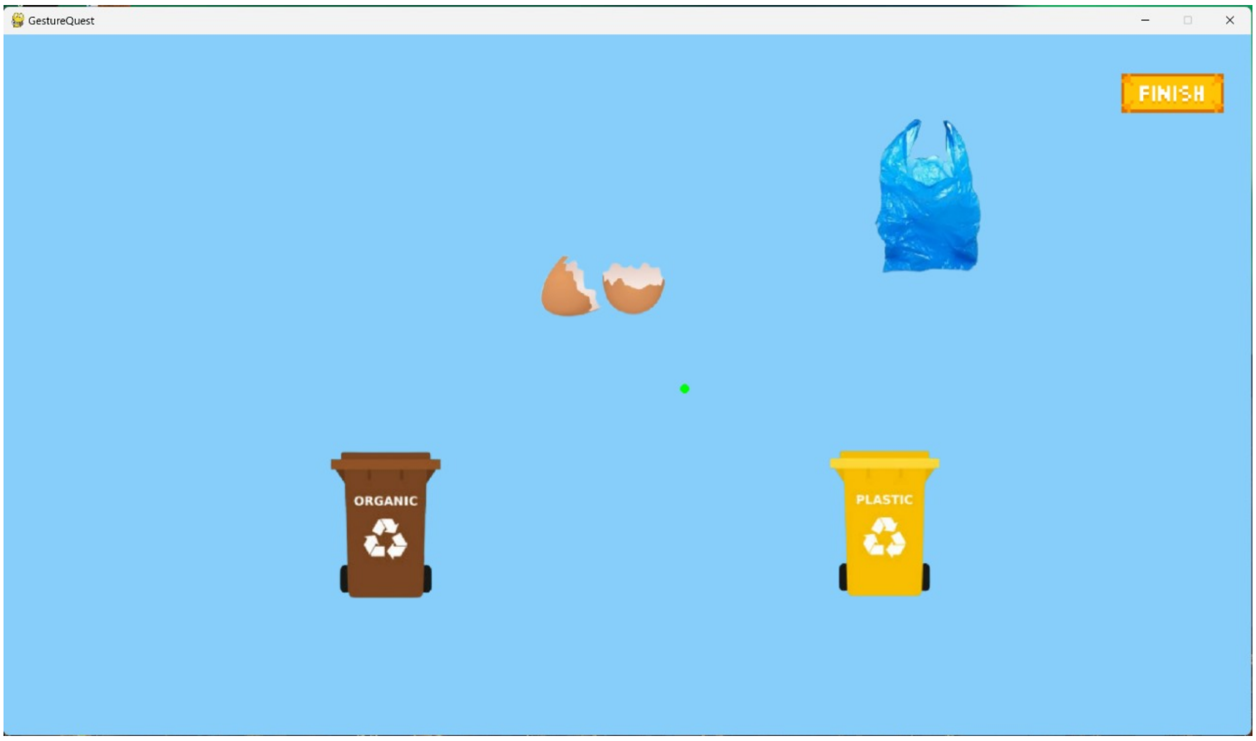


Figure 4.5: Waste Sort Intermediate Stage

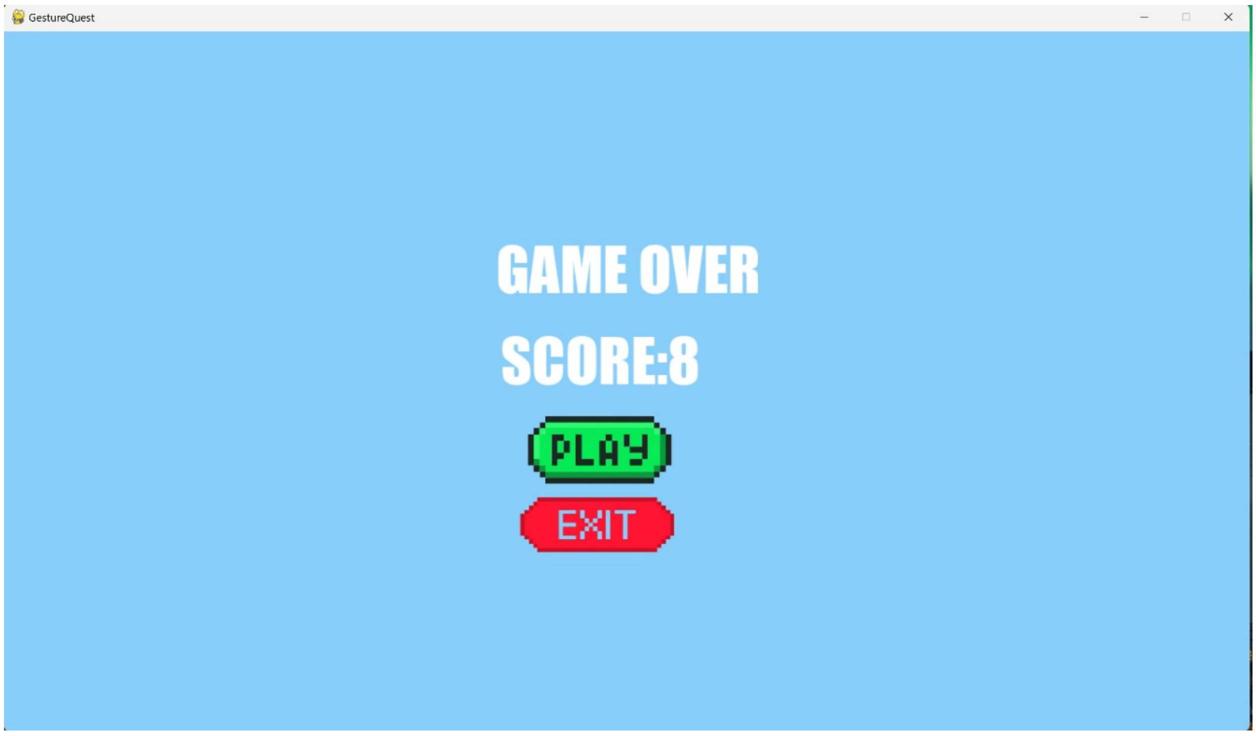


Figure 4.6: Waste Sort Final Stage

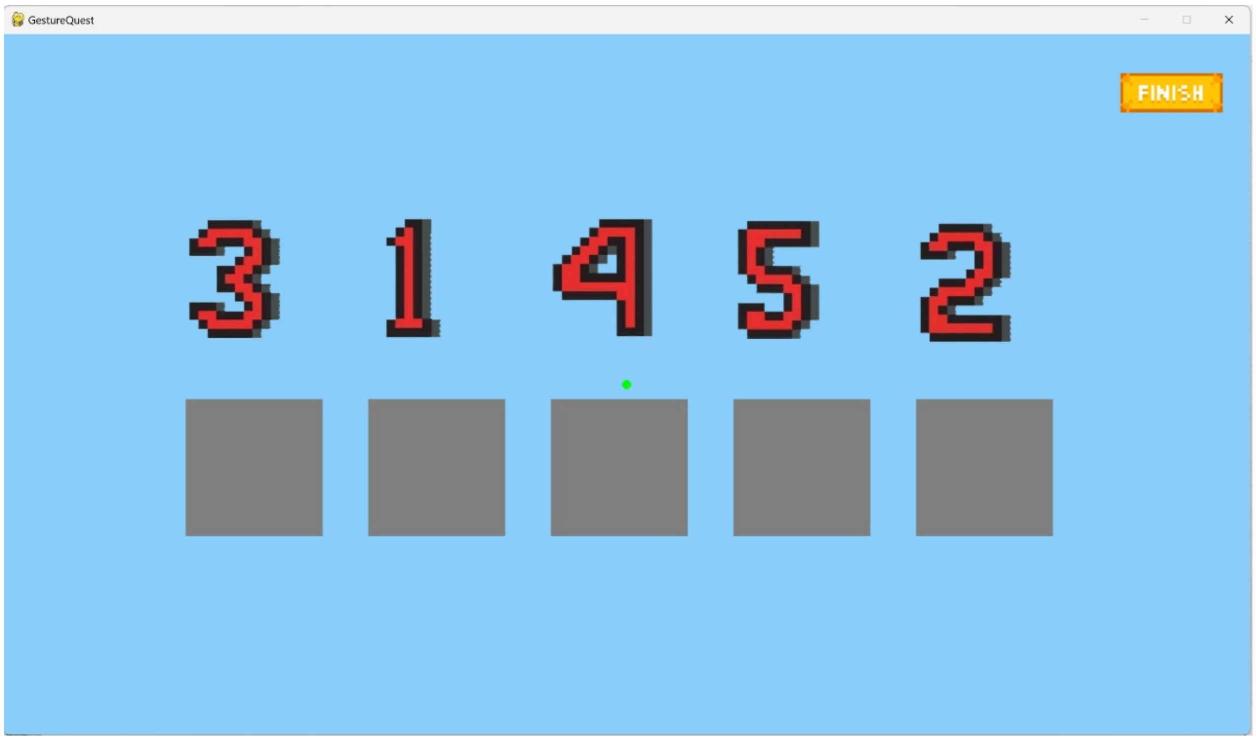


Figure 4.7: NumSort Initial Stage

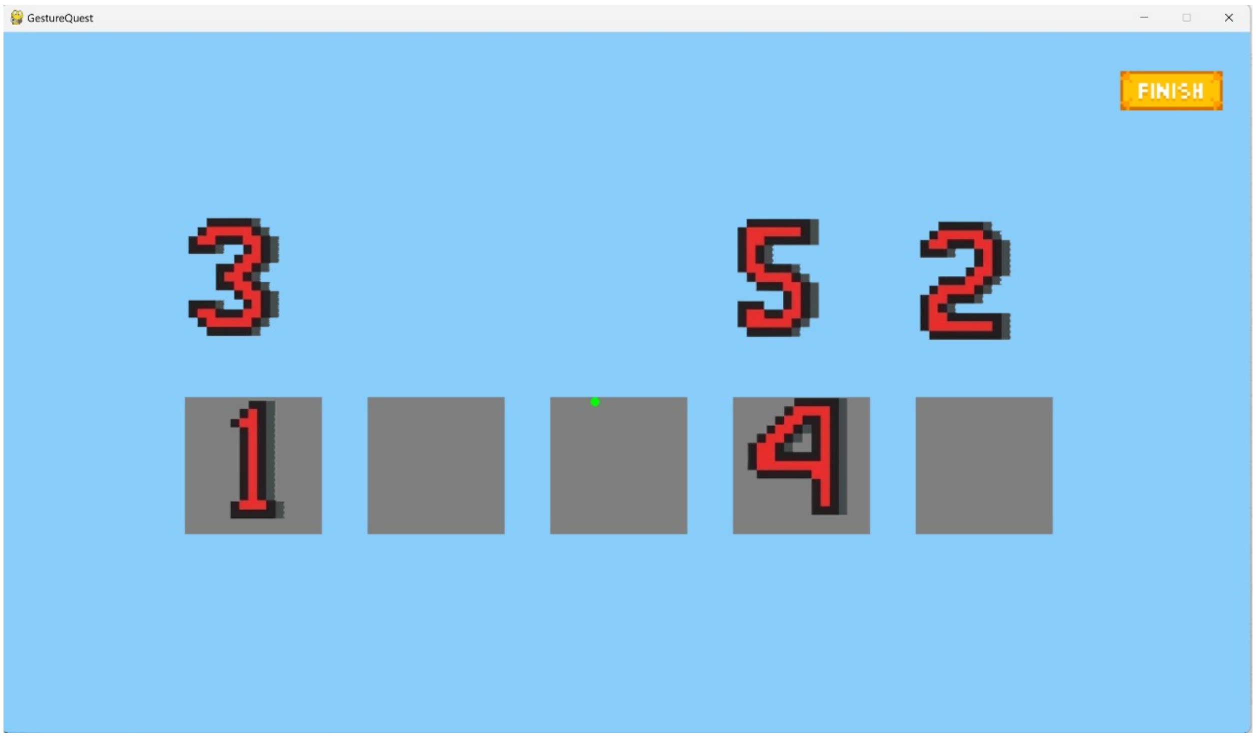


Figure 4.8: NumSort Intermediate Stage

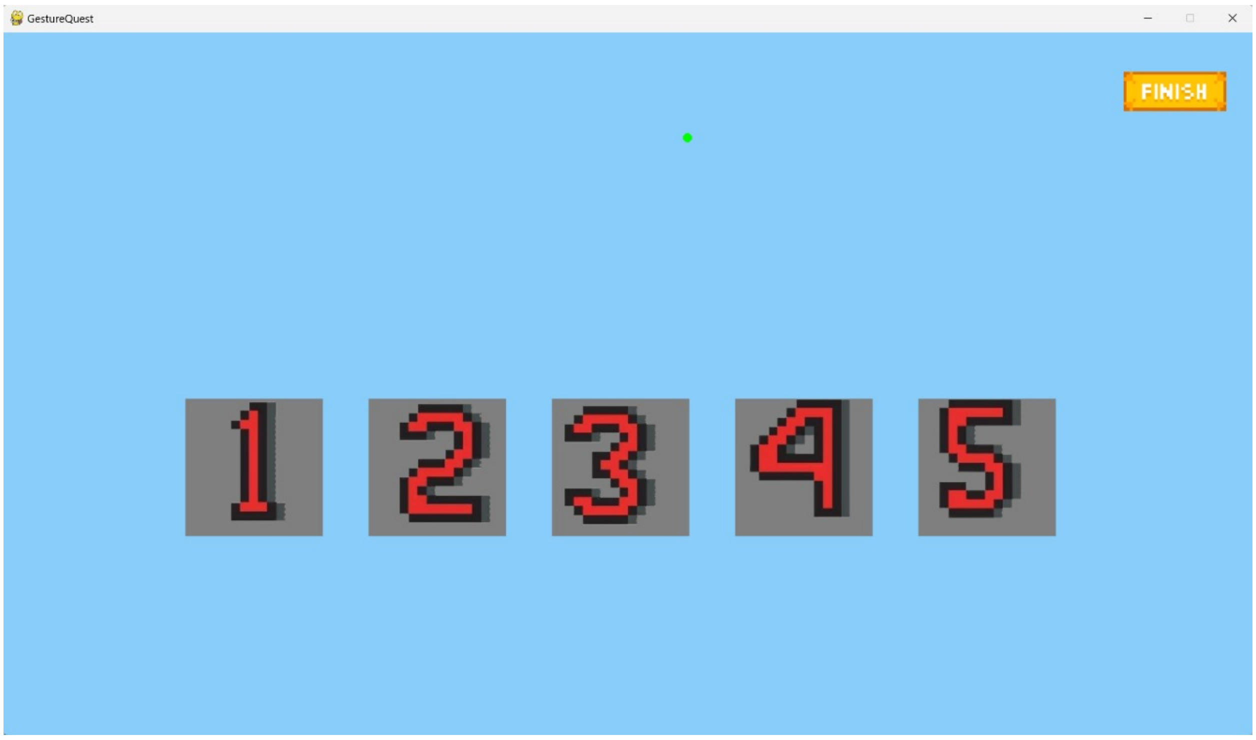


Figure 4.9: NumSort After Correct Sorting Stage

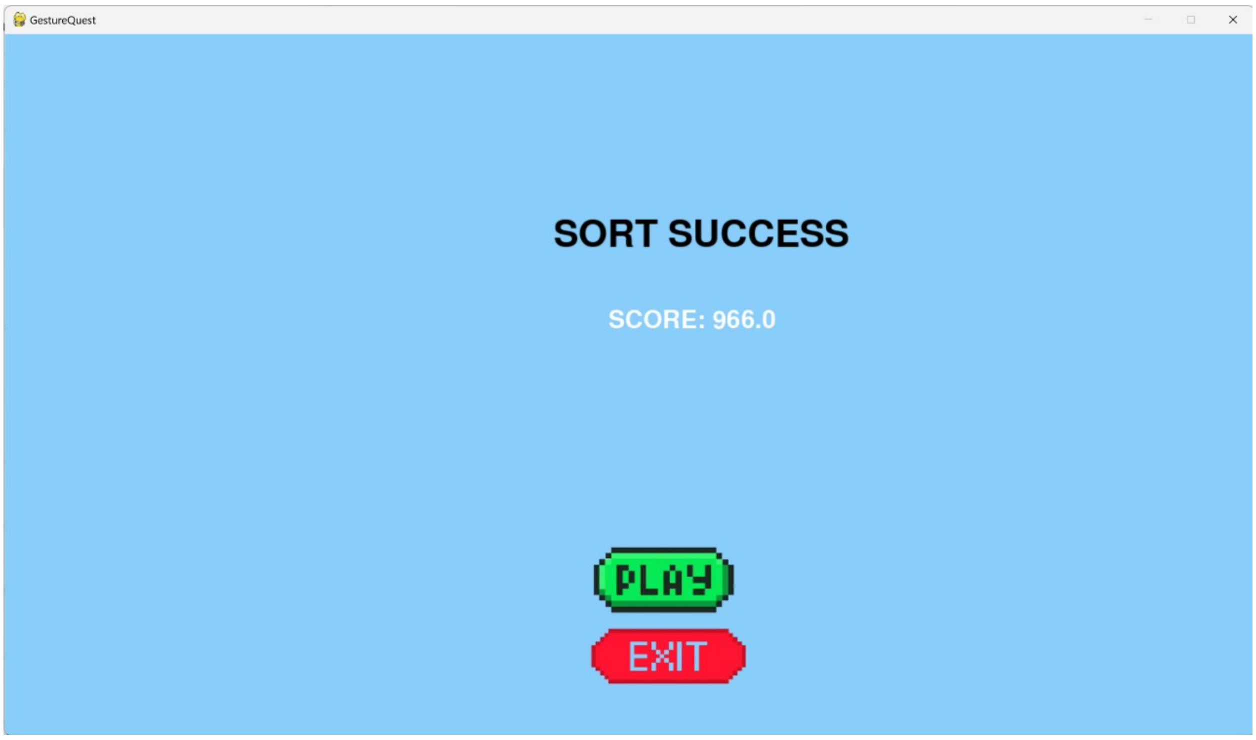


Figure 4.10: NumSort After Correct Sort Final Stage

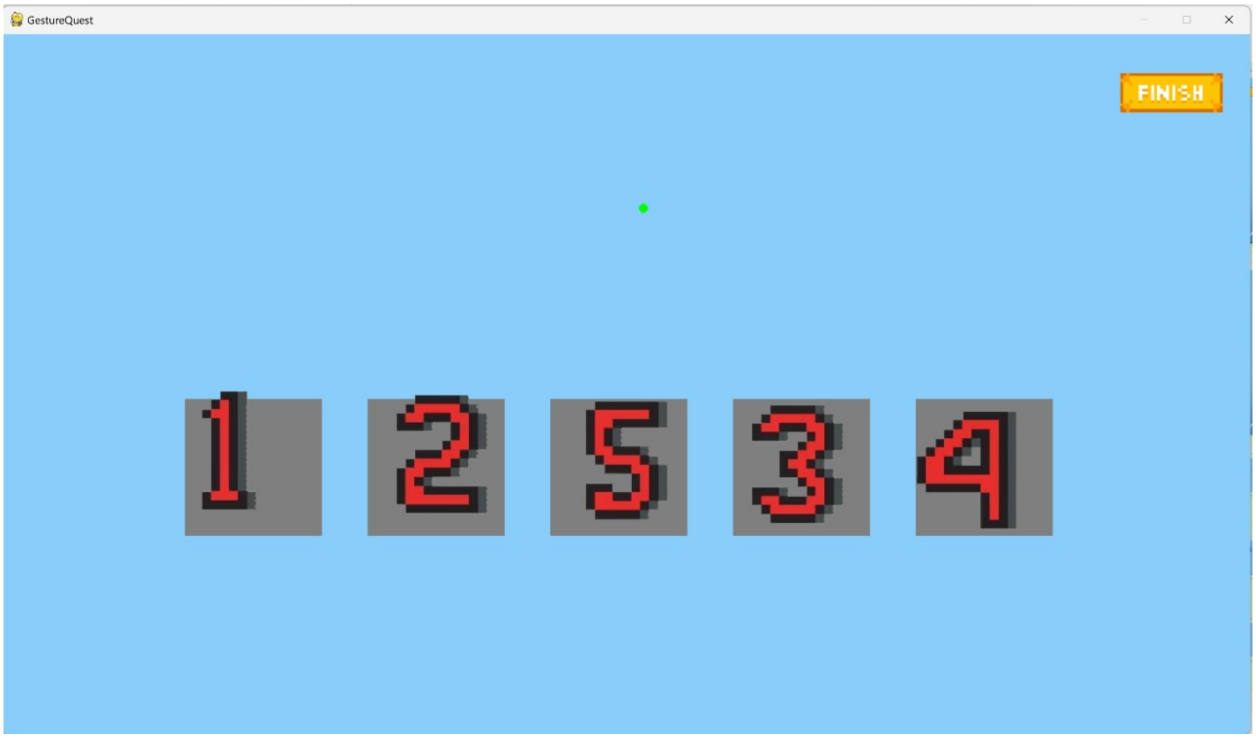


Figure 4.11: NumSort After Wrong Sorting Stage

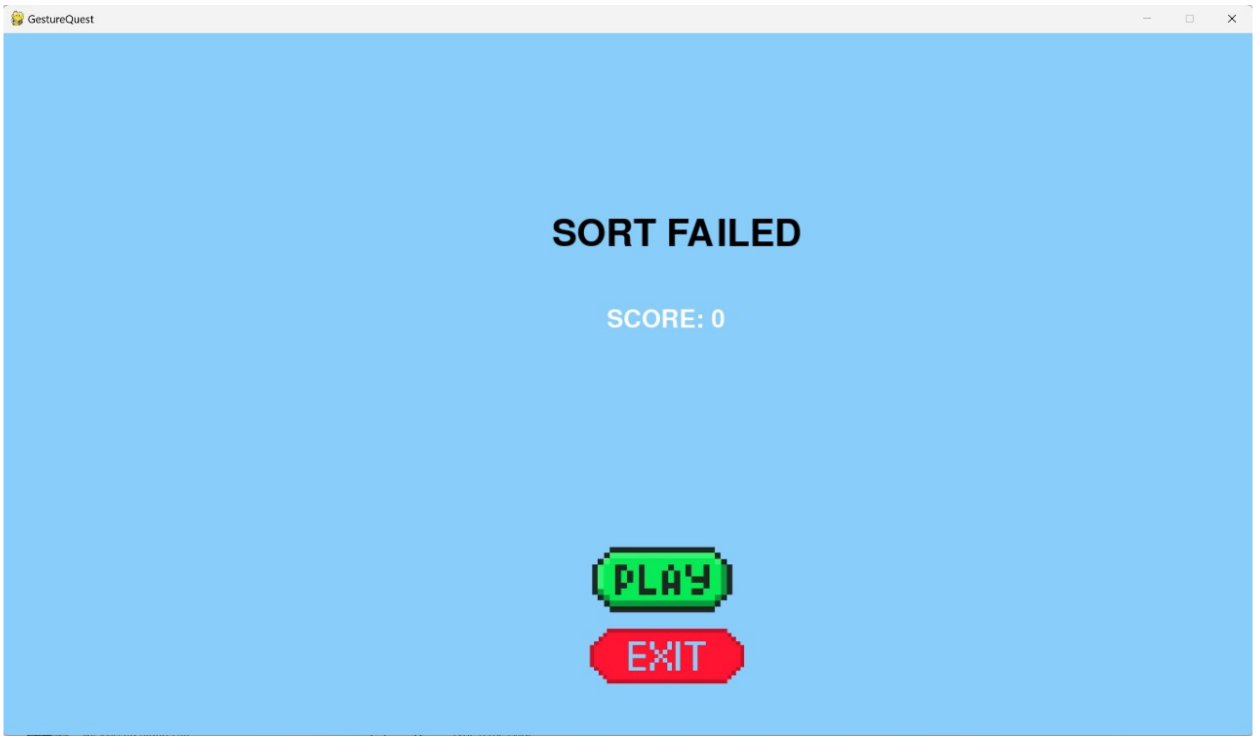


Figure 4.12: NumSort After Wrong Sort Final Stage

4.3 Discussion

GestureQuest, a project integrating gesture-based controls into interactive gaming, achieved notable results with high accuracy and user satisfaction. By employing Python, Pygame, and Mediapipe for real-time hand gesture tracking, the game ensures seamless player interaction.

In the Waste Sorting Game, a key component, players sort virtual waste using hand gestures. The gesture recognition system reached an accuracy rate of around 90 percent, ensuring responsive gameplay. This high accuracy was essential for maintaining user engagement and game fluidity. Performance decreased slightly under varying lighting conditions compared to controlled environments, indicating a need for improved gesture recognition algorithms for diverse settings.

The high accuracy and low latency confirm the project's success in delivering an engaging and intuitive experience. The identified deviations highlight opportunities for future enhancements, such as better handling of different lighting conditions and expanding the gesture repertoire. Overall, GestureQuest demonstrates the potential of gesture-based gaming, providing a solid foundation for further development in interactive applications.

Chapter 5

Conclusion

5.1 Conclusion

A revolutionary Gesture-Based Interactive Game for immersive learning and entertainment is designed to engage users with intuitive hand gesture recognition and interactive gameplay. This innovative technology transforms the traditional gaming experience, allowing players to interact naturally through hand gestures, significantly enhancing the overall enjoyment and immersion. The seamless integration of gesture-based controls makes the game more accessible and engaging for a diverse range of users, creating a unique and captivating experience that bridges the gap between physical and virtual interactions. By enabling players to use their hands to control the game, it offers a more intuitive and engaging way to play, fostering a deeper connection with the game environment.

In addition to its entertainment value, the game is designed to enhance cognitive development through educational elements and customizable settings. By incorporating a wide variety of educational content, it promotes learning in a fun and interactive way, making it an invaluable tool for both entertainment and education. The customizable settings allow the game to be tailored to individual learning needs and preferences, ensuring a personalized and effective learning experience. This adaptability makes the game suitable for users of all ages and learning stages, providing a comprehensive platform for cognitive growth. The combination of entertainment and education not only makes the game enjoyable but also supports cognitive development, providing a holistic and enriching experience that benefits users in multiple ways.

5.2 Future Scope

- Introduce progressively challenging tasks and level progression to keep players engaged and improve their sorting skills.

- Allow for using both hands simultaneously for more complex sorting tasks or introducing cooperative gameplay.

- Incorporate points, rewards, leaderboards, and different game modes to enhance motivation and create a more competitive or social gaming experience.

- Players could compete or collaborate in sorting challenges, adding a layer of social interaction and friendly competition.

Bibliography

[1] Dardas, N. H., & Petriu, E. M. (2011, September). Hand gesture detection and recognition using principal component analysis. In 2011 IEEE International conference on computational intelligence for measurement systems and applications (CIMSA) proceedings (pp. 1-6). IEEE.

[2] Li, T., Yan, Y., & Du, W. (2022, June). Sign language recognition based on computer vision. In 2022 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA) (pp. 927-931). IEEE.

[3] Davis, A., Overmyer, S., Jordan, K., Caruso, J., Dandashi, F., Dinh, A., ... & Theofanos, M. (1993, May). Identifying and measuring quality in a software requirements specification. In [1993] Proceedings First International Software Metrics Symposium (pp. 141-152). Ieee.

[4] Neethu, P. S., Suguna, R., & Sathish, D. (2020). An efficient method for human hand gesture detection and recognition using deep learning convolutional neural networks. Soft Computing, 24(20), 15239-15248.

Appendix A: Presentation

GESTURE-BASED INTERACTIVE GAME

GUIDE:
Ms. Amitha Mathew

TEAM MEMBERS:
Rohan Raghavan
Sreya P
Sneha R
Vismaya Balakrishnan

CONTENTS

INTRODUCTION

- Human-Computer Interaction (HCI) with gesture recognition technology.
- Growing importance of immersive experiences in education and entertainment.
- Combines entertainment and education through hand gesture interaction.

The Need:
- Foster cognitive development through engaging gameplay.
- Provide a natural and intuitive interaction experience.
- Address the demand for innovative educational tools.

PROBLEM DEFINITION

Development of a gesture-based interactive game leveraging hand gesture recognition technology, aiming to blend entertainment and education while fostering cognitive development through interactive gameplay.

OBJECTIVES

- Create a fun and immersive experience with intuitive hand gesture controls.
- Improve problem-solving and decision-making through sorting challenges.
- Enhance motor skills and reflexes with precise hand movements and improve Hand-Eye Coordination
- Learning & Competition: Promote learning about sorting concepts while encouraging competition.
- Performance: Optimize performance for better gameplay.

SCOPE AND RELEVANCE

Scope

- ❑ Users sort items using intuitive hand gestures, creating an engaging and accessible gameplay experience.
- ❑ Integrate MediaPipe and OpenCV to demonstrate real-time hand tracking and gesture recognition within a game environment.

Relevance

- ❑ Accessibility: Offers alternative control for gamers with limited mobility
- ❑ Education: Gamifies learning through interactive sorting and hand gesture recognition.

SYSTEM DESIGN : OVERVIEW

Modules:

Module 1:

Data Acquisition: Captures live video feed.

Module 2:

Hand Detection Module: Identifies hand landmarks using Media-pipe Hands framework.

Module 3:

Gesture Recognition Module: Analyzes hand landmarks to recognize predefined gestures.

Module 4:

User Interface (Pygame): Renders GUI for visualization and feedback.

MODULE 1: DATA ACQUISITION

- ❑ **OpenCV for Video Capture:** Utilizes OpenCV's functionalities to capture a live video stream from your webcam.
- ❑ **Webcam Access:** It establishes a connection to your default camera (usually the webcam).
- ❑ **Optional Resolution Setting:** The code allows you to optionally set the desired resolution for the captured video.
- ❑ **Continuous Frame Grabbing:** The code enters a continuous loop, grabbing individual frames (images) that make up the video stream one by one.
- ❑ **Frames as Snapshots:** Each captured frame is essentially a snapshot of the video at a specific moment in time.

- **Frames as NumPy Arrays:** These captured frames are stored as NumPy arrays, a data structure efficient for handling large image datasets.
- **Frames for Processing:** The captured frames are then available for further processing within the program, allowing for real-time interaction with the video data.
- **Potential Processing Examples:** This processing might involve tasks like flipping the frame for proper orientation or converting the color format for compatibility with other libraries.
- **Analogy: Video Recorder:** In essence, acts like a video recorder, continuously capturing frames from the live feed and preparing them for further use within the program.

MODULE 2: HAND DETECTION MODULE

- **Hand Detection Setup:** The system would first initialize itself for hand detection. This likely involves loading necessary models and preparing for image processing.
- **Frame Analysis:** Each captured frame is then analyzed to see if there's a hand present.
- **Hand Landmark Detection:** If a hand is found in the frame, the system can pinpoint important points on it, like your fingertips. These points are often called "landmarks".
- **Natural Interaction:** By understanding hand posture and fingertip locations, the system allows for more natural interaction with objects on the screen based on your actual hand movements.

MODULE 3: GESTURE RECOGNITION MODULE

- **Simulated Fingertips:** The code simulates a pinch gesture by using pre-defined locations for the thumb and index fingertip.
- **Distance Calculation:** It calculates the distance between these assumed fingertip locations using the Euclidean distance formula.
- **Threshold for Pinch:** A threshold distance (currently 0.1) is set. This represents the maximum allowed distance between the fingertips for a pinch to be registered. The code compares the calculated distance with the threshold.
- **Simplified Approach:** This is a simplified way to mimic a pinch gesture, not a real-world hand recognition method.

ALGORITHM

AIM TRAINER

1. Capture and process the frames from the video capture to detect hands using MediaPipe.
2. If hands are detected:
   2.1. Extract the index and thumb finger tip coordinates.
   2.2. Update the pointer position with smoothing based on the fingertip coordinates.
   2.3. Check if it is a pinch and potentially update score and target position.
3. Render and display the text for score and remaining time (if the timer has started).
4. Check if the timer has run out and set the game over flag if so.
5. Update the display.

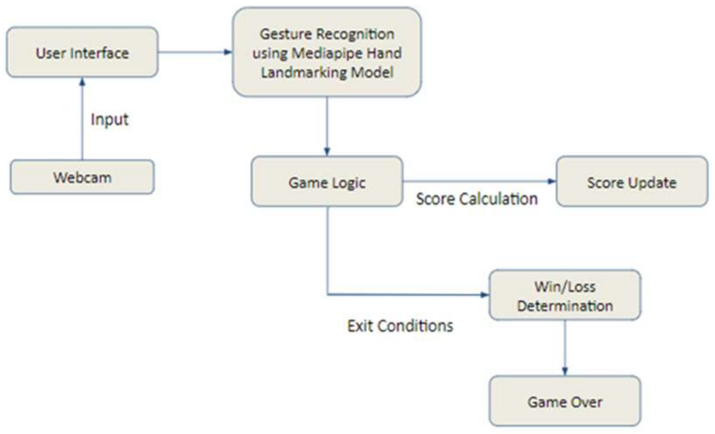
SCORE IS UPDATED BASED ON THE NUMBER OF CLICKS IN 30 SECONDS

WASTE SORT/NUMSORT

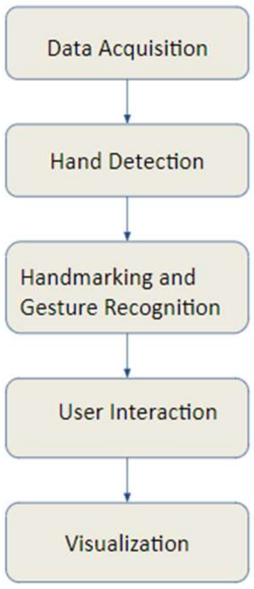
1. Capture and process the frames from the video capture to detect hands using MediaPipe.
2. Update pointer position based on smoothed fingertip position.
3. Check for pinch gesture and update object selection and scoring.
4. Calculate the final score based on time and precision.
5. Display the final score on the screen.
6. Check for button presses (replay or exit) and takes appropriate actions (reset or transition to menu).

BOTH USE SIMILAR ALGORITHMS. WASTESORT IS GROUPING BASED WHILE NUMSORT IS SORTING BASED

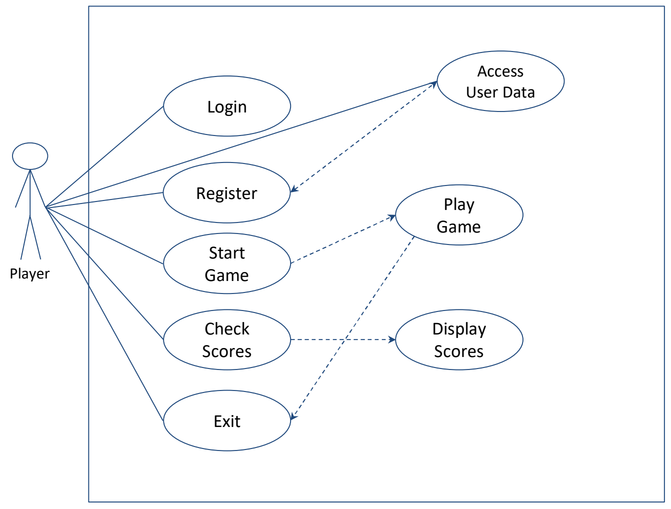
ARCHITECTURAL DESIGN



Workflow:



USE CASE DIAGRAM



DATASETS

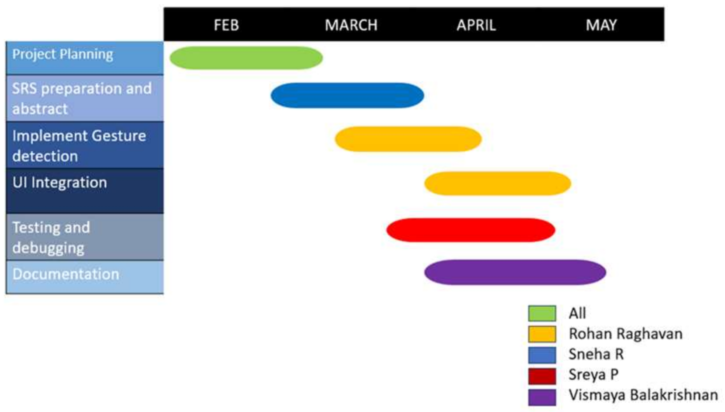
- The project utilizes a pre-trained model from the Mediapipe framework for hand landmark detection. This model has been trained on a large dataset of hand images containing various hand poses, orientations, and skin tones.

- Source: python mediapipe library

[Hand landmarks detection guide | MediaPipe | Google for Developers]

Work Division

Gantt Chart



SOFTWARE/HARDWARE REQUIREMENTS

Software:

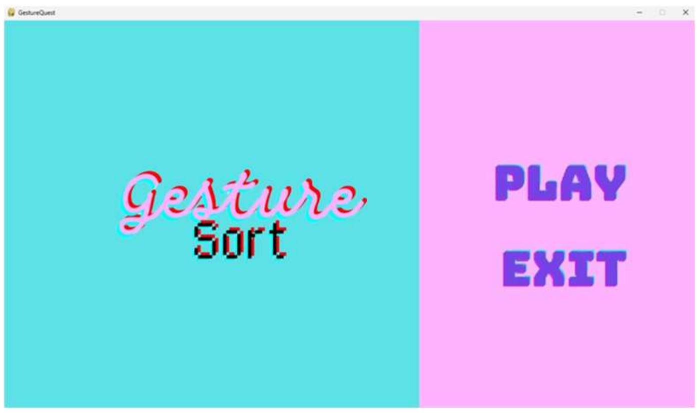
- Operating System: Windows 7
- Coding Language: Python
- Tools: Visual Studio Code
- Libraries: OpenCV, MediaPipe, PyGame
- Frontend Framework: PyGame
- Backend Framework: Python 3

Hardware:

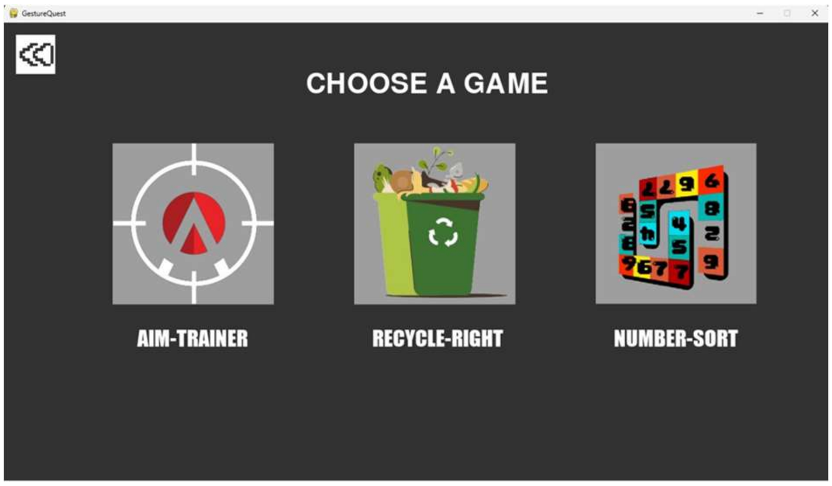
- Input Devices: Webcam(minimum resolution of 480p)
- Processor: Single-core processor with a clock speed of 1.5 GHz or higher
- Memory (RAM): Minimum 2 GB RAM
- Graphics Card: Integrated or dedicated graphics card with support for OpenGL 2.0 or higher
- Storage: Minimum 200 MB of available storage space

RESULTS

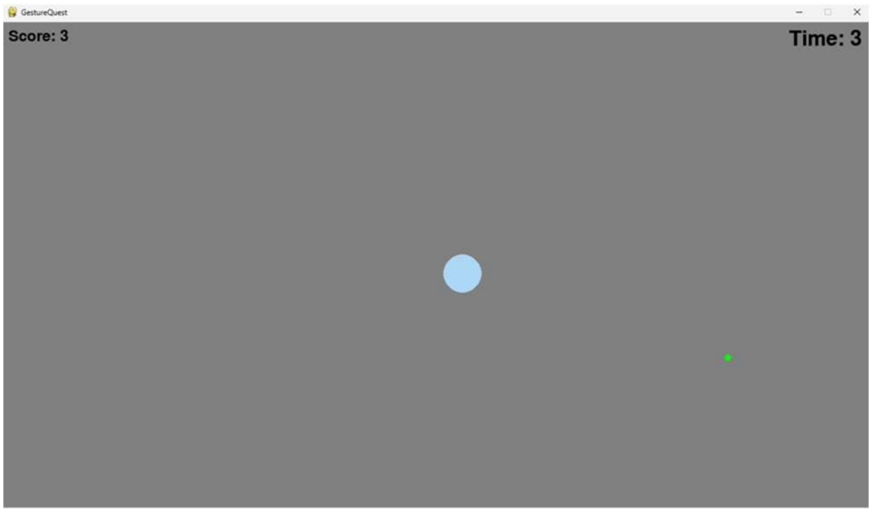
GESTURE SORT UI



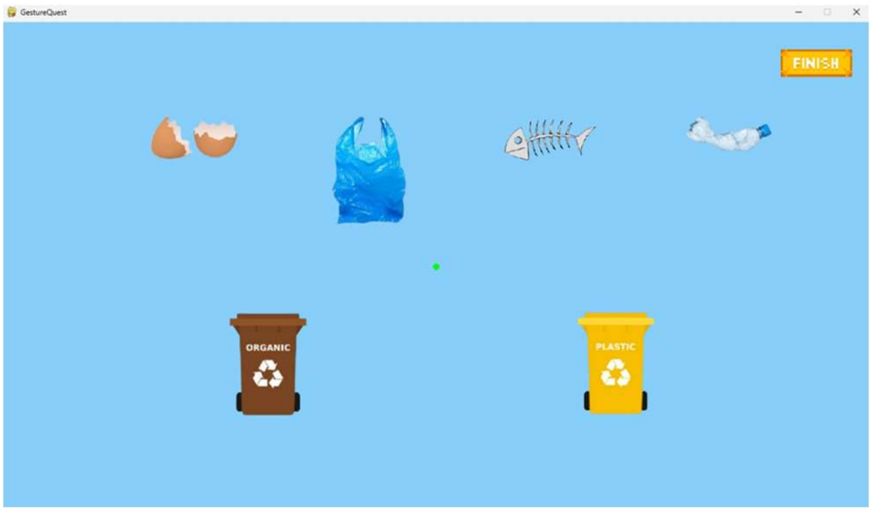
MAIN MENU

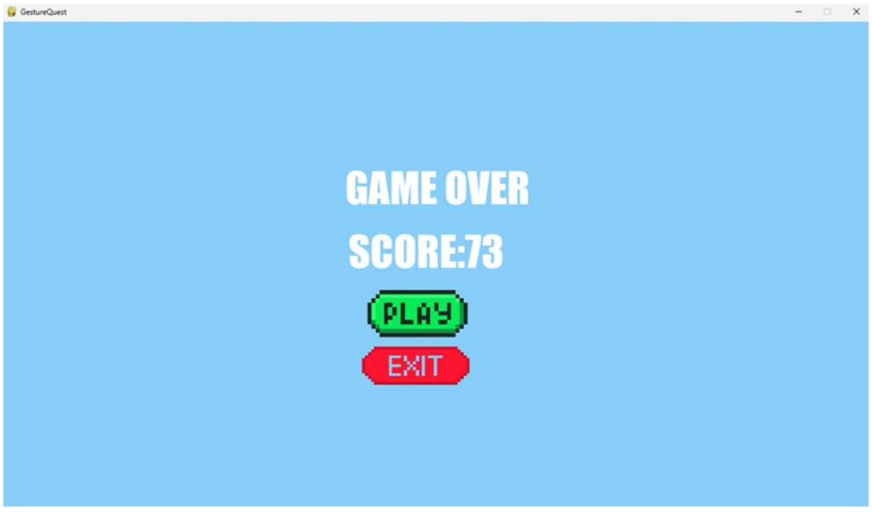


AIM TRAINER

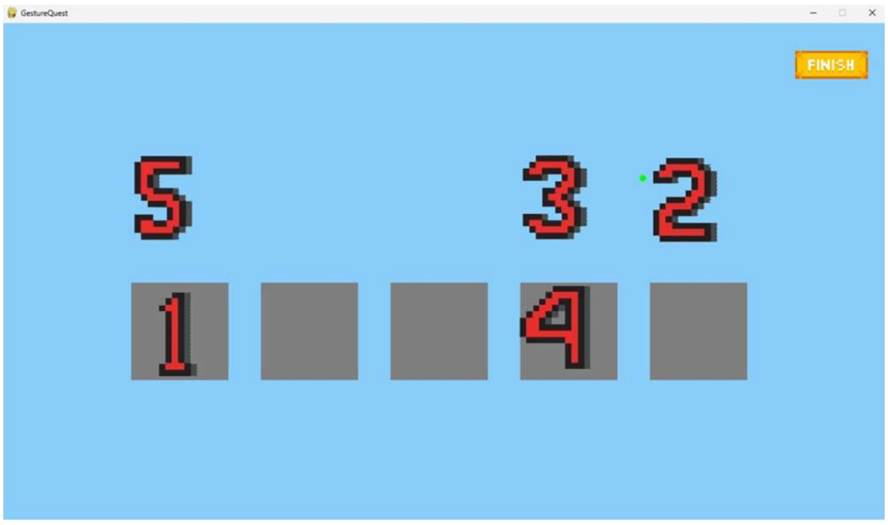


WASTE SORT


GestureQuest
FINISH
ORGANIC
PLASTIC


GestureQuest
GAME OVER
SCORE:73
PLAY
EXIT

NUMSORT



CONCLUSION

- A revolutionary Gesture-Based Interactive Game for immersive learning and entertainment.
- Engage users with intuitive hand gesture recognition and interactive gameplay.
- Enhance cognitive development through educational elements and customizable settings.

FUTURE ENHANCEMENTS

- Introduce progressively challenging tasks and level progression to keep players engaged and improve their sorting skills.
- Allow for using both hands simultaneously for more complex sorting tasks or introducing cooperative gameplay.
- Incorporate points, rewards, leaderboards, and different game modes to enhance motivation and create a more competitive or social gaming experience.

REFERENCES

1. Dardas, N. H., & Petriu, E. M. (2011, September). Hand gesture detection and recognition using principal component analysis. In *2011 IEEE International conference on computational intelligence for measurement systems and applications (CIMSA) proceedings* (pp. 1-6). IEEE.
2. Li, T., Yan, Y., & Du, W. (2022, June). Sign language recognition based on computer vision. In *2022 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)* (pp. 927-931). IEEE.
3. Davis, A., Overmyer, S., Jordan, K., Caruso, J., Dandashi, F., Dinh, A., ... & Theofanos, M. (1993, May). Identifying and measuring quality in a software requirements specification. In *[1993] Proceedings First International Software Metrics Symposium* (pp. 141-152). Ieee.
4. Neethu, P. S., Suguna, R., & Sathish, D. (2020). An efficient method for human hand gesture detection and recognition using deep learning convolutional neural networks. *Soft Computing*, *24*(20), 15239-15248.

Appendix B: Vision, Mission, Programme Outcomes and Course Outcomes

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

RAJAGIRI SCHOOL OF ENGINEERING & TECHNOLOGY (AUTONOMOUS)

RAJAGIRI VALLEY, KAKKANAD, KOCHI, 682039

(Affiliated to APJ Abdul Kalam Technological University)

Vision, Mission, Programme Outcomes and Course Outcomes

Institute Vision

To evolve into a premier technological institution, moulding eminent professionals with creative minds, innovative ideas and sound practical skill, and to shape a future where technology works for the enrichment of mankind.

Institute Mission

To impart state-of-the-art knowledge to individuals in various technological disciplines and to inculcate in them a high degree of social consciousness and human values, thereby enabling them to face the challenges of life with courage and conviction.

Department Vision

To become a centre of excellence in Computer Science and Engineering, moulding professionals catering to the research and professional needs of national and international organizations.

Department Mission

To inspire and nurture students, with up-to-date knowledge in Computer Science and Engineering, ethics, team spirit, leadership abilities, innovation and creativity to come out with solutions meeting societal needs.

Programme Outcomes (PO)

Engineering Graduates will be able to:

1. Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and Team work: Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.

10. Communication: Communicate effectively with the engineering community and with society at large. Be able to comprehend and write effective reports documentation. Make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team. Manage projects in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

Programme Specific Outcomes (PSO)

A graduate of the Computer Science and Engineering Program will demonstrate:

PSO1: Computer Science Specific Skills
The ability to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas by understanding the core principles and concepts of computer science and thereby engage in national grand challenges.

PSO2: Programming and Software Development Skills
The ability to acquire programming efficiency by designing algorithms and applying standard practices in software project development to deliver quality software products meeting the demands of the industry.

PSO3: Professional Skills
The ability to apply the fundamentals of computer science in competitive research and to develop innovative products to meet the societal needs thereby evolving as an eminent researcher and entrepreneur.

Course Outcomes

After the completion of the course the student will be able to:

CO1:
Identify technically and economically feasible problems (Cognitive Knowledge Level: Apply)

CO2:
Identify and survey the relevant literature for getting exposed to related solutions and get familiarized with software development processes (Cognitive Knowledge Level: Apply)

CO3:
Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions of minimal complexity by using modern tools & advanced programming techniques (Cognitive Knowledge Level: Apply)

CO4:
Prepare technical report and deliver presentation (Cognitive Knowledge Level: Apply)

CO5:
Apply engineering and management principles to achieve the goal of the project (Cognitive Knowledge Level: Apply)

Appendix C: CO-PO-PSO Mapping

COURSE OUTCOMES:

After completion of the course the student will be able to

SL. NO	**DESCRIPTION**	**Blooms' Taxonomy Level**
CO1	Identify technically and economically feasible problems (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO2	Identify and survey the relevant literature for getting exposed to related solutions and get familiarized with software development processes (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO3	Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions of minimal complexity by using modern tools & advanced programming techniques (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO4	Prepare technical report and deliver presentation (Cognitive Knowledge Level: Apply)	Level 3: Apply
CO5	Apply engineering and management principles to achieve the goal of the project (Cognitive Knowledge Level: Apply)	Level 3: Apply

CO-PO AND CO-PSO MAPPING

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO3
CO1	3	3	3	3		2	2	3	2	2	2	3	2	2	2
CO2	3	3	3	3	3	2		3	2	3	2	3	2	2	2
CO3	3	3	3	3	3	2	2	3	2	2	2	3			2
CO4	2	3	2	2	2			3	3	3	2	3	2	2	2
CO5	3	3	3	2	2	2	2	3	2		2	3	2	2	2

3/2/1: high/medium/low

JUSTIFICATIONS FOR CO-PO MAPPING

MAPPING	LOW/ MEDIUM/ HIGH	JUSTIFICATION
101003/CS622T.1-PO1	**HIGH**	Identify technically and economically feasible problems by applying the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
101003/CS622T.1-PO2	**HIGH**	Identify technically and economically feasible problems by analysing complex engineering problems reaching substantiated conclusions using first principles of mathematics.
101003/CS622T.1-PO3	**HIGH**	Design solutions for complex engineering problems by identifying technically and economically feasible problems.
101003/CS622T.1-PO4	**HIGH**	Identify technically and economically feasible problems by analysis and interpretation of data.
101003/CS622T.1-PO6	**MEDIUM**	Responsibilities relevant to the professional engineering practice by identifying the problem.
101003/CS622T.1-PO7	**MEDIUM**	Identify technically and economically feasible problems by understanding the impact of the professional engineering solutions.
101003/CS622T.1-PO8	**HIGH**	Apply ethical principles and commit to professional ethics to identify technically and economically feasible problems.
101003/CS622T.1-PO9	**MEDIUM**	Identify technically and economically feasible problems by working as a team.
101003/CS622T.1-PO10	**MEDIUM**	Communicate effectively with the engineering community by identifying technically and economically feasible problems.
101003/CS622T.1-P011	**MEDIUM**	Demonstrate knowledge and understanding of engineering and management principles by selecting the technically and economically feasible problems.
101003/CS622T.1-PO12	**HIGH**	Identify technically and economically feasible problems for long term learning.
101003/CS622T.1-PSO1	**MEDIUM**	Ability to identify, analyze and design solutions to identify technically and economically feasible problems.
101003/CS622T.1-PSO2	**MEDIUM**	By designing algorithms and applying standard practices in software project development and Identifying technically and economically feasible problems.
101003/CS622T.1-PSO3	**MEDIUM**	Fundamentals of computer science in competitive research can be applied to Identify technically and economically feasible problems.
101003/CS622T.2-PO1	**HIGH**	Identify and survey the relevant by applying the knowledge of mathematics, science, engineering fundamentals.

101003/CS622T.2-PO2	**HIGH**	Identify, formulate, review research literature, and analyze complex engineering problems get familiarized with software development processes.
101003/CS622T.2-PO3	**HIGH**	Design solutions for complex engineering problems and design based on the relevant literature.
101003/CS622T.2-PO4	**HIGH**	Use research-based knowledge including design of experiments based on relevant literature.
101003/CS622T.2-PO5	**HIGH**	Identify and survey the relevant literature for getting exposed to related solutions and get familiarized with software development processes by using modern tools.
101003/CS622T.2-PO6	**MEDIUM**	Create, select, and apply appropriate techniques, resources, by identifying and surveying the relevant literature.
101003/CS622T.2-PO8	**HIGH**	Apply ethical principles and commit to professional ethics based on the relevant literature.
101003/CS622T.2-PO9	**MEDIUM**	Identify and survey the relevant literature as a team.
101003/CS622T.2-PO10	**HIGH**	Identify and survey the relevant literature for a good communication to the engineering fraternity.
101003/CS622T.2-PO11	**MEDIUM**	Identify and survey the relevant literature to demonstrate knowledge and understanding of engineering and management principles.
101003/CS622T.2-PO12	**HIGH**	Identify and survey the relevant literature for independent and lifelong learning.
101003/CS622T.2-PSO1	**MEDIUM**	Design solutions for complex engineering problems by Identifying and survey the relevant literature.
101003/CS622T.2-PSO2	**MEDIUM**	Identify and survey the relevant literature for acquiring programming efficiency by designing algorithms and applying standard practices.
101003/CS622T.2-PSO3	**MEDIUM**	Identify and survey the relevant literature to apply the fundamentals of computer science in competitive research.
101003/CS622T.3-PO1	**HIGH**	Perform requirement analysis, identify design methodologies by using modern tools & advanced programming techniques and by applying the knowledge of mathematics, science, engineering fundamentals.
101003/CS622T.3-PO2	**HIGH**	Identify, formulate, review research literature for requirement analysis, identify design methodologies and develop adaptable & reusable solutions.

101003/CS622T.3-PO3	**HIGH**	Design solutions for complex engineering problems and perform requirement analysis, identify design methodologies.
101003/CS622T.3-PO4	**HIGH**	Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
101003/CS622T.3-PO5	**HIGH**	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools.
101003/CS622T.3-PO6	**MEDIUM**	Perform requirement analysis, identify design methodologies and assess societal, health, safety, legal, and cultural issues.
101003/CS622T.3-PO7	**MEDIUM**	Understand the impact of the professional engineering solutions in societal and environmental contexts and Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions.
101003/CS622T.3-PO8	**HIGH**	Perform requirement analysis, identify design methodologies and develop adaptable & reusable solutions by applying ethical principles and commit to professional ethics.
101003/CS622T.3-PO9	**MEDIUM**	Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.
101003/CS622T.3-PO10	**MEDIUM**	Communicate effectively with the engineering community and with society at large to perform requirement analysis, identify design methodologies.
101003/CS622T.3-PO11	**MEDIUM**	Demonstrate knowledge and understanding of engineering requirement analysis by identifying design methodologies.
101003/CS622T.3-PO12	**HIGH**	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change by analysis, identify design methodologies and develop adaptable & reusable solutions.
101003/CS622T.3-PSO3	**MEDIUM**	The ability to apply the fundamentals of computer science in competitive research and prior to that perform requirement analysis, identify design methodologies.
101003/CS622T.4-PO1	**MEDIUM**	Prepare technical report and deliver presentation by applying the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
101003/CS622T.4-PO2	**HIGH**	Identify, formulate, review research literature, and analyze complex engineering problems by preparing technical report and deliver presentation.

101003/CS622T.4-PO3	**MEDIUM**	Prepare Design solutions for complex engineering problems and create technical report and deliver presentation.
101003/CS622T.4-PO4	**MEDIUM**	Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions and prepare technical report and deliver presentation.
101003/CS622T.4-PO5	**MEDIUM**	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools and Prepare technical report and deliver presentation.
101003/CS622T.4-PO8	**HIGH**	Prepare technical report and deliver presentation by applying ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
101003/CS622T.4-PO9	**HIGH**	Prepare technical report and deliver presentation effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.
101003/CS622T.4-PO10	**HIGH**	Communicate effectively with the engineering community and with society at large by prepare technical report and deliver presentation.
101003/CS622T.4-PO11	**MEDIUM**	Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work by prepare technical report and deliver presentation.
101003/CS622T.4-PO12	**HIGH**	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change by prepare technical report and deliver presentation.
101003/CS622T.4-PSO1	**MEDIUM**	Prepare a technical report and deliver presentation to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas.
101003/CS622T.4-PSO2	**MEDIUM**	To acquire programming efficiency by designing algorithms and applying standard practices in software project development and to prepare technical report and deliver presentation.
101003/CS622T.4-PSO3	**MEDIUM**	To apply the fundamentals of computer science in competitive research and to develop innovative products to meet the societal needs by preparing technical report and deliver presentation.
101003/CS622T.5-PO1	**HIGH**	Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
101003/CS622T.5-PO2	**HIGH**	Identify, formulate, review research literature, and analyze complex engineering problems by applying engineering and management principles to achieve the goal of the project.

101003/CS622T.5-PO3	**HIGH**	Apply engineering and management principles to achieve the goal of the project and to design solutions for complex engineering problems and design system components or processes that meet the specified needs.
101003/CS622T.5-PO4	**MEDIUM**	Apply engineering and management principles to achieve the goal of the project and use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
101003/CS622T.5-PO5	**MEDIUM**	Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools and to apply engineering and management principles to achieve the goal of the project.
101003/CS622T.5-PO6	**MEDIUM**	Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities by applying engineering and management principles to achieve the goal of the project.
101003/CS622T.5-PO7	**MEDIUM**	Understand the impact of the professional engineering solutions in societal and environmental contexts, and apply engineering and management principles to achieve the goal of the project.
101003/CS622T.5-PO8	**HIGH**	Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice and to use the engineering and management principles to achieve the goal of the project.
101003/CS622T.5-PO9	**MEDIUM**	Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings and to apply engineering and management principles to achieve the goal of the project.
101003/CS622T.5-PO11	**MEDIUM**	Demonstrate knowledge and understanding of engineering and management principles and apply these to one’s own work, as a member and leader in a team. Manage projects in multidisciplinary environments and to apply engineering and management principles to achieve the goal of the project.
101003/CS622T.5-PO12	**HIGH**	Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change and to apply engineering and management principles to achieve the goal of the project.
101003/CS622T.5-PSO1	**MEDIUM**	The ability to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas. Apply engineering and management principles to achieve the goal of the project.

101003/CS622T.5-PSO2	**MEDIUM**	The ability to acquire programming efficiency by designing algorithms and applying standard practices in software project development to deliver quality software products meeting the demands of the industry and to apply engineering and management principles to achieve the goal of the project.
101003/CS622T.5-PSO3	**MEDIUM**	The ability to apply the fundamentals of computer science in competitive research and to develop innovative products to meet the societal needs thereby evolving as an eminent researcher and entrepreneur and apply engineering and management principles to achieve the goal of the project.