**4**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**System Analysis**

**4.1 Study of Current System**

**4.2 Problem and Weaknesses of Current System**

**4.3 Requirements of New System**

**4.4 System Feasibility**

**4.4.1 Does the system contribute to the overall**

**Objectives of the organization?**

**4.4.2 Can the system be implemented using the**

**current technology and within the given cost**

**and schedule constraints**

**4.4.3 Can the system be integrated with other**

**systems which are already in place?**

**4.5 Activity / Process in New System / Proposed System**

**4.6 Features of New System / Proposed System**

**4.7 List Main Modules / Components / Processes**

**/ Techniques of New System / Proposed System**

**4.8 Selection of Hardware / Software / Algorithms**

**/ Methodology / Techniques / Approaches and Justification**

**4.1 Study of Current System**

The existing system is a subfield of computer vision and pattern recognition, which aims to recognize human gestures or actions through the use of cameras or other sensing devices. Python is a popular programming language for building computer vision and machine learning systems, and there are many libraries and tools available for implementing gesture recognition systems using Python.

One popular Python library for gesture recognition is OpenCV, which provides a wide range of computer vision algorithms and functions for detecting and tracking objects, as well as recognizing and classifying gestures. OpenCV can be used to process video or image data from cameras or other sensing devices, and can be combined with machine learning algorithms to improve recognition accuracy.

Another popular Python library for gesture recognition is TensorFlow, an open-source machine learning framework developed by Google. TensorFlow provides tools for building and training deep neural networks, which can be used for recognizing complex patterns and features in gesture data. TensorFlow also provides pre-trained models and APIs for recognizing common gestures, such as hand gestures.

In addition to OpenCV and TensorFlow, there are many other Python libraries and tools available for gesture recognition, including scikit-learn for machine learning, Mediapipe for hand Detection, and PyAutoGUI for mouse cursor.

**4.2 Problem and Weaknesses of Current System**

**Current system has some weakness**

* Performance is slow
* Some field is required to add in this system
* UI is not proper as per the system require
* User experience is not proper
* System performance is slow
* Some functionality missing.

**4.3 Requirements of New System**

* **Gameplay mechanics**:

The new system should include fun and engaging fruit-slicing gameplay mechanics that are easy to learn and hard to master. This could involve introducing new fruits with unique properties or adding power-ups that enhance the slicing experience.

* **Graphics** **and** **animations**:

The new system should have high-quality graphics and animations that bring the fruit-slicing experience to life. This could include realistic fruit textures and particle effects that create a satisfying slicing sensation.

* **User** **interface**:

The new system should have a user-friendly interface that makes it easy for players to start playing the game and navigate through the menus. This could involve designing an intuitive and visually appealing main menu and in-game HUD.

* **Multiplayer** **functionality**:

The new system could include multiplayer functionality that allows players to compete against each other online or locally. This could involve creating different multiplayer modes, such as co-op or versus, and implementing matchmaking and leaderboards.

* **Cross**-**platform** **compatibility**:

The new system should be compatible with different platforms, including mobile devices, PCs, and game consoles. This could involve optimizing the game for each platform's hardware and input mechanisms.

* **Localization**:

The new system should be localized to different languages and regions to reach a wider audience. This could involve translating the game's text and audio into different languages and adapting the game's mechanics to suit different cultural norms.

* **Performance** **optimization**:

The new system should be optimized for performance, ensuring that the game runs smoothly on a wide range of devices without lag or other performance issues. This could involve optimizing the game's code, reducing the game's file size, and implementing efficient resource management techniques.

**4.4 System Feasibility**

The feasibility of a gesture recognition project depends on a combination of hardware and software requirements, accuracy and reliability, user acceptance, and cost-effectiveness. If these factors can be adequately addressed, then the project is feasible.

* **Hardware Requirements:** Gesture recognition projects require specific hardware components such as cameras or sensors that can capture the movements of the user's body. The feasibility of the project depends on the availability and cost of such hardware.
* **Software Requirements:** Gesture recognition projects also require software components that can process the data captured by the hardware and recognize the gestures made by the user. The feasibility of the project depends on the availability of suitable software and the expertise required to develop or customize it.
* **Accuracy and Reliability:** The accuracy and reliability of gesture recognition technology are critical to its success. The feasibility of the project depends on the ability of the technology to correctly recognize a wide range of gestures under different lighting and environmental conditions.
* **User Acceptance:** The success of a gesture recognition project also depends on the acceptance of users. The feasibility of the project depends on whether the technology is intuitive, easy to use, and adds value to the user experience.
* **Cost:** The feasibility of the project also depends on its cost-effectiveness. The cost of developing and implementing gesture recognition technology must be justified by the potential benefits it offers.

**4.4.1 Does the system contribute to the overall objectives of the organization?**

The answer to whether a gesture recognition system contributes to the overall objective of an organization depends on the specific objectives of the organization and how the system is implemented. Here are some ways that a gesture recognition system could potentially contribute to an organization's overall objectives:

* Improving User Experience
* Increasing Efficiency
* Supporting Innovation
* Enhancing Safety

**4.4.2 Can the system be implemented using the current technology and within**

**the given cost and schedule constraints**

Yes, this system can be implemented within given cost and schedule constraints requires a thorough analysis of the specific requirements and constraints of the project. If the required technology is available and affordable, the budget is sufficient, and the schedule is realistic, then it may be feasible to implement the system.

**4.4.3 Can the system be integrated with other systems which are already in**

**place?**

Integrating this system with other systems that are already in place requires careful planning, design, and implementation. If the systems are compatible and interoperable, and appropriate security measures are in place, then it may be feasible to integrate the systems.

**4.5 Activity**

Activity Diagram displays a special state diagram where most of the states are action states and most of the transitions are triggered by completion of the actions in the source states. This diagram focuses on flows driven by internal processing. Activity diagrams show the flow of activities through the system. Diagrams are read from top to bottom and have branches and forks to describe conditions and parallel activities. A fork is used when multiple activities are occurring at the same time.

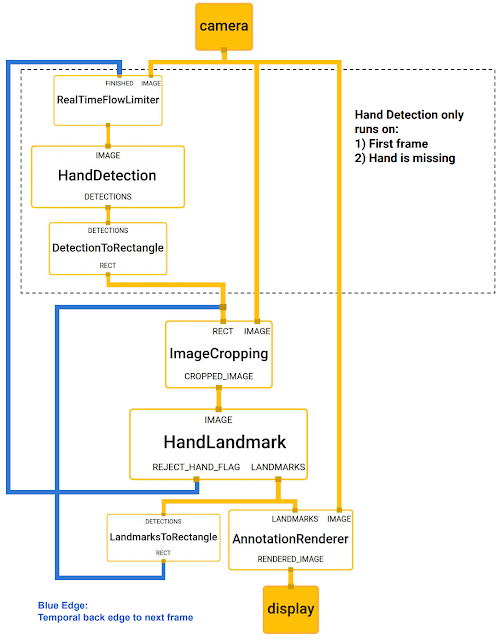
****

Fig 4.5.1 Activity Diagram

**4.6 Features of New System**

* Multi-modal Input
* Multi-Gesture Recognition
* High Accuracy
* Robustness
* Customizable Gestures
* Continuous Improvement
* Integration
* Proper Detection

**4.7 List Main Modules**

* OpenCV
* NumPy
* TensorFlow or PyTorch
* Unity ML-Agents
* Pygame
* PyAutoGUI
* Mediapipe

**4.8 Selection of Hardware**

* **Processing power**: Gesture recognition algorithms can be computationally intensive, especially if you are processing high-resolution video feeds in real-time. Therefore, you need to select hardware with enough processing power to handle the load. For example, a high-end desktop computer or a dedicated graphics processing unit (GPU) can provide the necessary processing power for real-time gesture recognition.
* **Camera quality**: The quality of the camera used for capturing video feeds is also important. A high-quality camera can capture clear and detailed images, which can improve the accuracy of gesture recognition.