Project Report On

MAZE GAME USING
HAND GESTURE RECOGNITION

By

Abhishek Purandare 11209

Course: M.Sc. (Computer Science) Semester II

Paper: CS-204

P.V.G's College of Science, Pune 9

Affiliated to

Savitribai Phule Pune University

April / May 2019

Acknowledgement

I would like to express my special thanks of gratitude to my teacher **Mrs. Swati Joshi** who gave me this golden opportunity to do this wonderful project "*Maze Game using Hand Gesture Recognition*" which also helped me in doing a lot of research and I gained a Good knowledge about different technology.

Sincerely,

Abhishek Purandare

1. Problem Definition

Gestures are important aspects of Human Interactions both interpersonally and in the man-machine communication. A gesture is a form of non-verbal communication in which visible bodily actions performed particularly using Hands or Faces are used to pass messages or commands, either instead of speech or simultaneously with it. An example would be, Air Marshal using Hand Gestures to communicate with Aircraft Pilot.

Gesture Recognition is a topic in Computer Science and Language Technology with a goal of interpreting Human Gestures using Mathematical algorithms. Computer Vision is an interdisciplinary field which deals with how computers can be made to gain high-level understanding from Digital Images or Videos. Gesture Recognition is a field in Computer Vision that deals with human-computer interaction. It can be seen as a way of computers beginning to understand our language.

Hand Gesture Recognition acts as a highly adaptive interface between machine and the user. Using this technology, Machine operations can be performed using a series of fingers and hand movements, eliminating the need of physical contact between operator and machine. There are two types of Gesture Recognition, Vision-based and Glove-based.

We have been using gestures since the ancient times. So, it’s only natural and efficient that we implement applications which make our lives easier using something we are familiar with. Furthermore, disabled people have used gestures as a communication medium. These applications will help them immensely. In the earlier days, focus was on speed of the application, but now it has shifted towards user-friendly environment.

This project is a simple maze game application which can be controlled using hand gestures which are uniquely identified and provided as input for performing different actions. The project performs various techniques such as Image Pre-processing, Morphological Operations, Skin segmentation, etc. The Gesture recognition is based on simple Mathematical approach.

2. Existing System and Need for The New System

2.1 Existing System

- It is a stand-alone GUI application.
- The Game can be controlled using the devices such as Keyboard, Mouse and Joystick.
- It is difficult for some people to use the computer without any knowledge on how to use it.
- Above mentioned input devices are not convenient for some users.

2.2 Need for the New System

- Since Humans have been using gestures for a long time, it is easier for us to communicate through this medium.
- The system establishes Human-Computer Interaction (HCI) via the language we all are familiar with.
- Eliminates the need of Physical interaction, Thus Remotely Controlling Applications.
- Exploiting the devices such as Camera to do more than just take a Picture!
- Overall aim of the system is to bridge the gap between human and machine understanding through Human body language.
- Allows the user to intuitively interact with the device.

3. Scope of the work

This Project aims the Human-Computer Interaction (HCI) for real-time Hand Gesture Recognition. The hand gestures are used for many applications such as robot control, medical treatment, augmented reality, Artificial Intelligence, image processing, video processing, etc.

In this Project, Hand gestures are used for playing and controlling the Maze Game. Since, devices like Keyboard or joystick do not provide any reliability to many users. Computer-vision based applications are necessary. The project does not need any dependencies except the camera.

Objectives

- Build a Module which recognizes natural forms of hand gestures in real-time to interact with the computer System.
- Implementation of a Maze Game using Graphical User Interface (GUI).
- Controlling the above application using Hand Gesture Recognition Module.

Here, User will be able to control the Sprite (Person) inside the Maze like Stopping or Moving it in any of the four directions (Up, Down, Left, Right). This application will have Vision-based gesture recognition where Image-based techniques detect a gesture by capturing pictures of user's hand in real-time. Each controlling command will have its unique gesture with reasonable accuracy.

4. Feasibility Study and Project Requirements

4.1 Technical Feasibility

- It examines whether the system is technically feasible or not.
- Camera is required.
- No external technology will be needed.
- If any problem arises in the system people can be consulted. Majority of people use hand gestures.
- Hence, it is technically feasible.

4.2 Operational Feasibility

- It examines whether the system will fulfil the user requirements or not.
- Rather than using devices, gestures are easier to perform.
- Users can adapt to this system very quickly.
- No complex actions.
- Illumination as well as quality of the camera may cause triviality, but under right conditions, it works like a charm!
- Hence, it is operationally feasible.

4.3 Economic Feasibility

- It examines whether the system is economically feasible or not.
- Nowadays, most laptops come with a built-in webcam. A desktop user must have a camera installed.
- No extra cost is needed for hardware and software.
- It does not incur any cost for implementing the system.
- Hence, it is economically feasible.

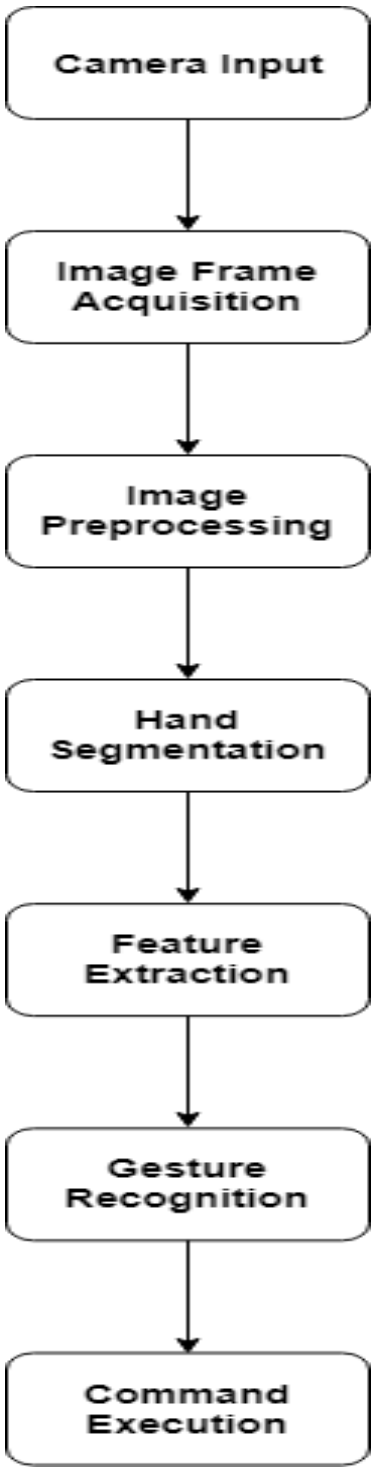
4.4 Software Requirements

- Operating System: Windows XP or above/Linux
- Drivers: Camera Drivers
- Programming Tool: Python 3.x
- Modules: opencv (2.4.1 or above), numpy (1.10 or above), arcade (2.0.0), cx_Freeze (5.1.1)

4.5 Hardware Requirements

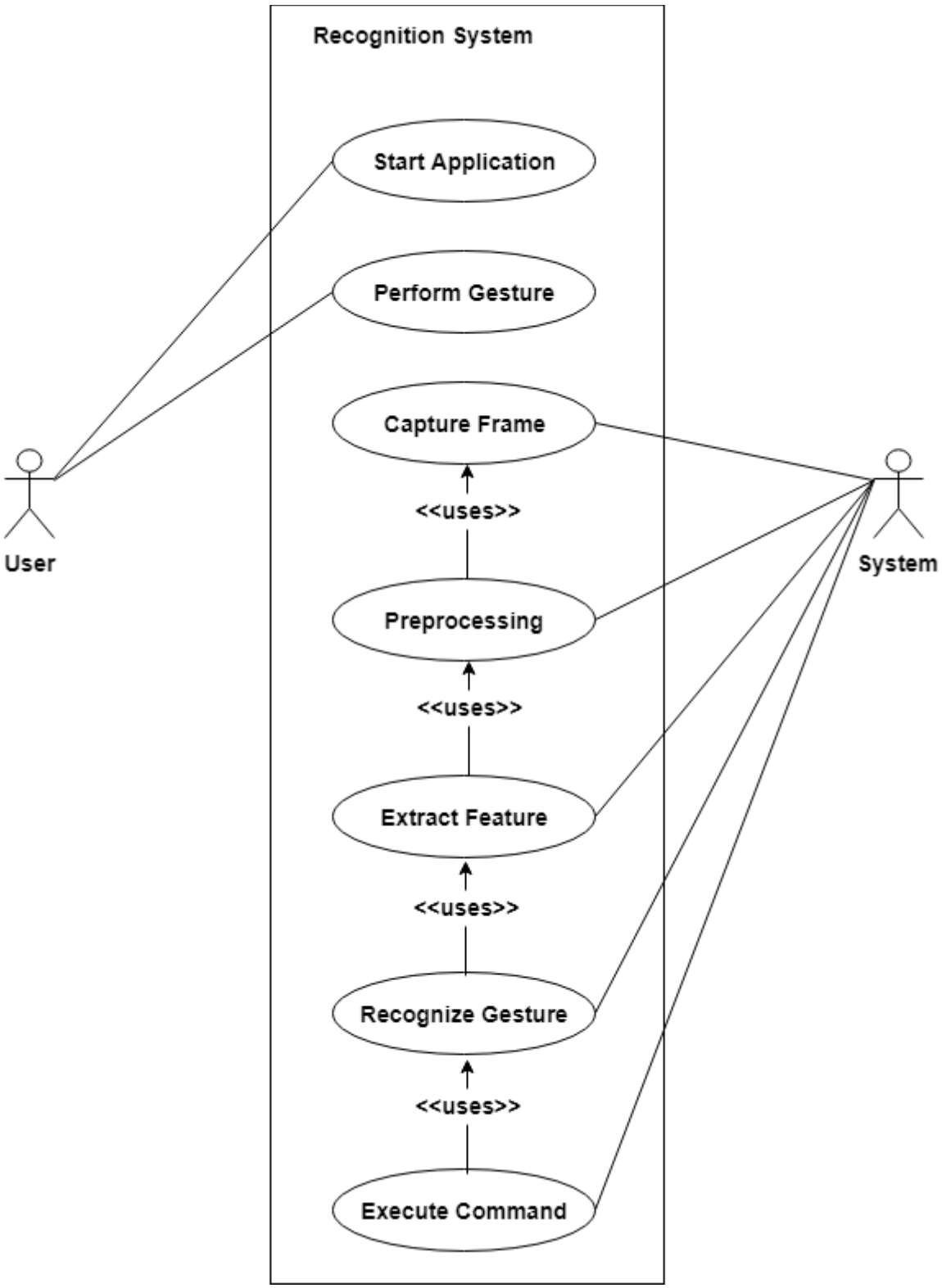
- Device: Camera/ Laptop Webcam
- RAM: 1 GB
- Memory Requirement: 400 MB
- Processor: Dual Core

5. Block Diagram

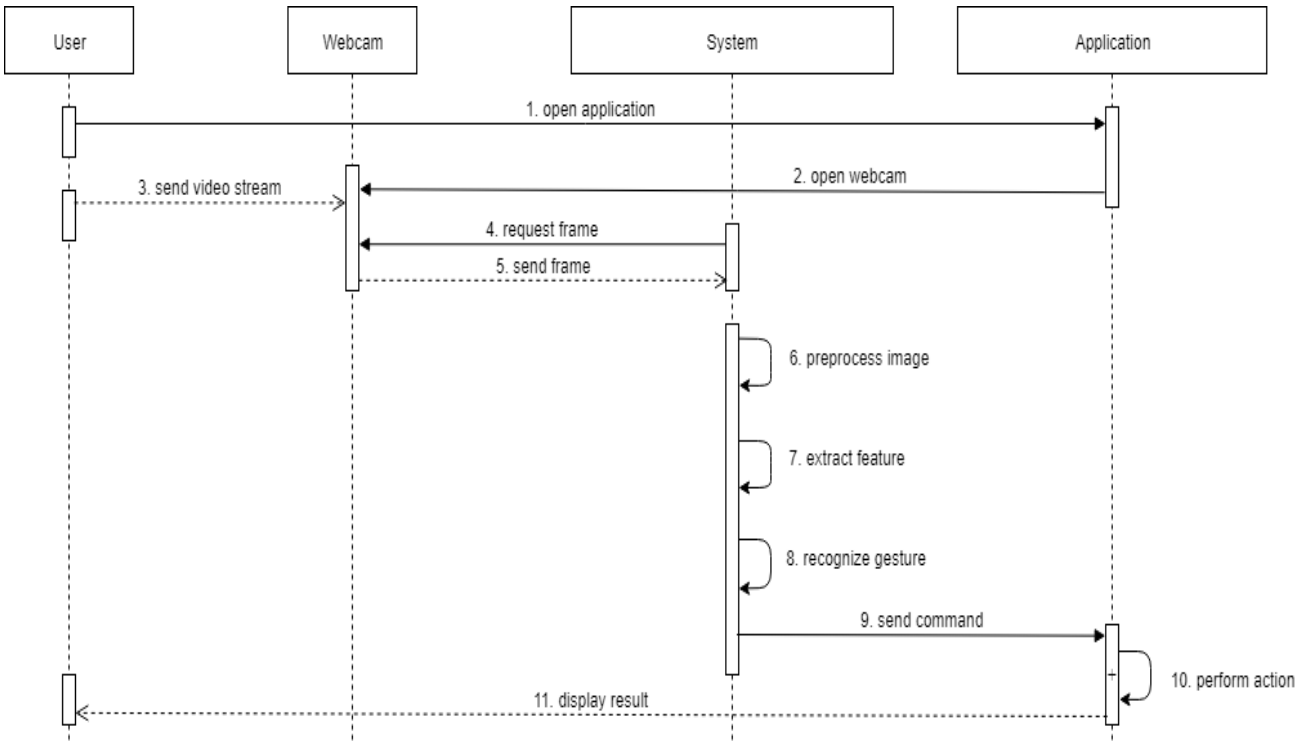


6. UML Diagrams

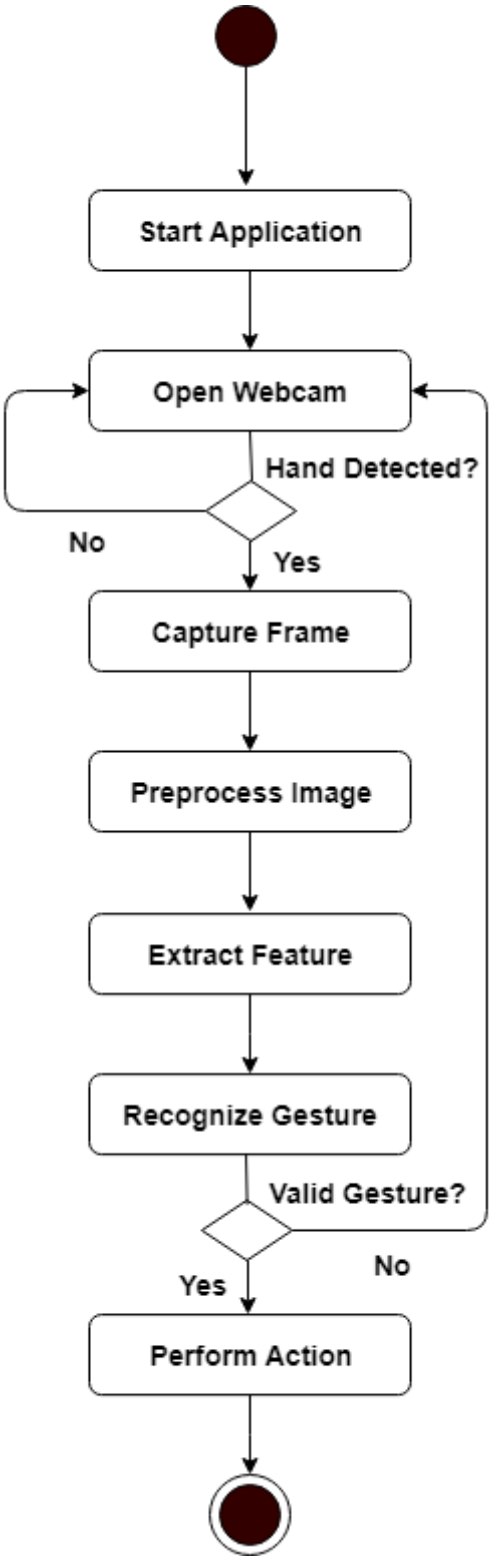
6.1 Use-case Diagram



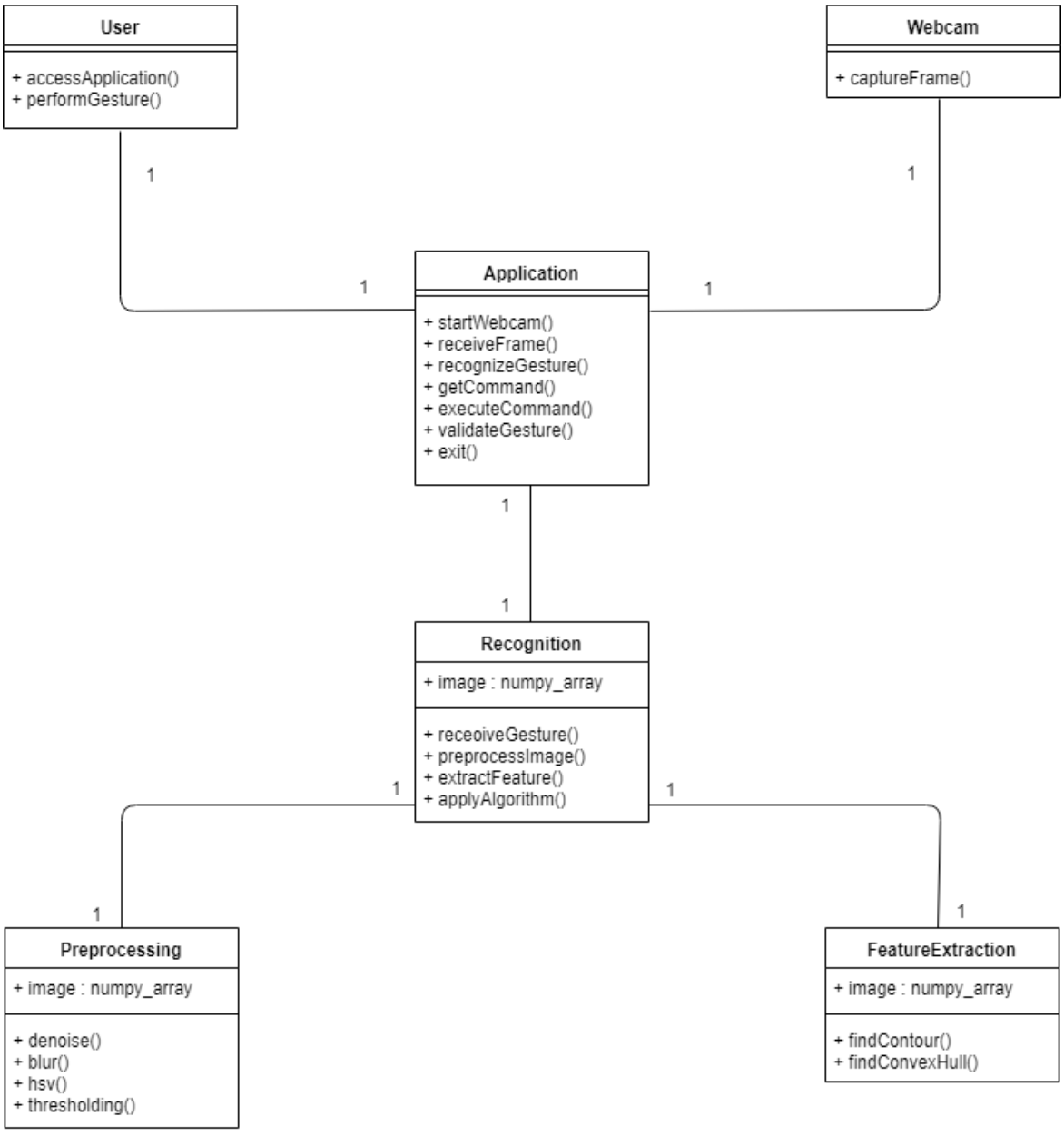
6.2 Sequence Diagram



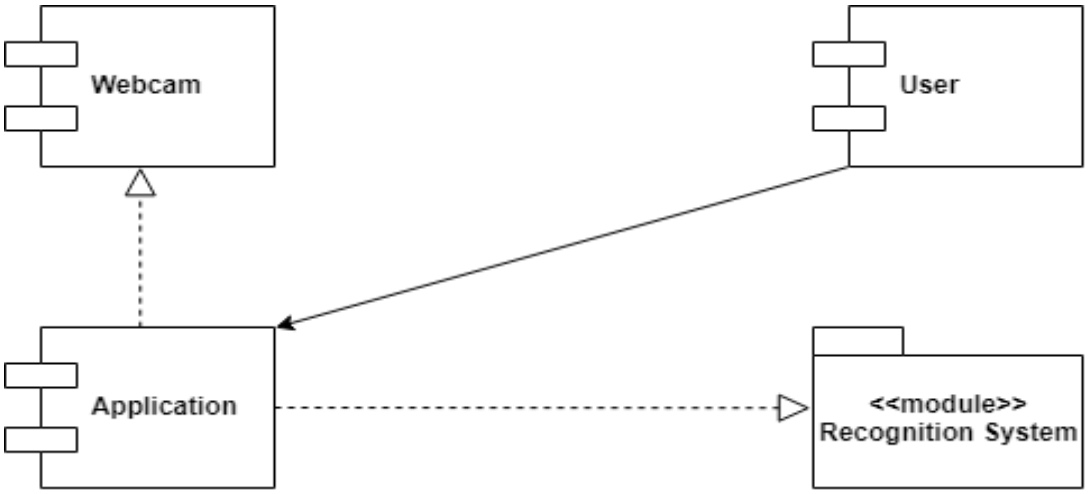
6.3 Activity Diagram



6.5 Class Diagram



6.6 Component Diagram



6.7 Deployment Diagram

7. Image Processing

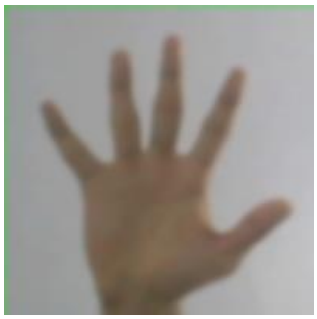
7.1 Image Acquisition

This is the very first step required to be performed. We are only interested in the picture of the hand. So, we define a Region of Interest (ROI) which is part of the original image where you put your hand in and perform the gestures. Only this part will be considered for further processing.

ROI

7.2 Image Pre-processing

To improve accuracy, we must first reduce the image noise by applying blur. Secondly, RGB colour model is not suitable for applications which need detection or segmentation. HSV colour model helps to improve the hand detection (skin segmentation in this case).



Blurred

HSV

7.3 Skin Segmentation and Morphological Filtering

After obtaining HSV image, skin segmentation is done by providing skin colour range in HSV such that every pixel with intensity out of that range is made black i.e. Background Subtraction. It is possible that not all regions inside the hand are considered to be in the skin colour range. This situation creates holes inside the hand segmented. To fill these holes, perform Morphological operations such as dilation and erosion. (Pictures below do not have any holes) Image below is the Bitwise AND output of ROI image and skin mask.

Skin Segmentation

Morphological Filtering

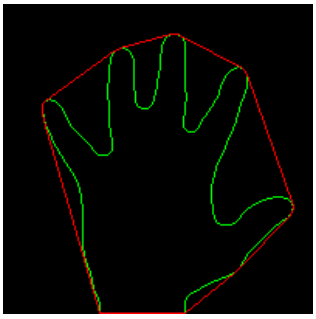
7.4 Image Thresholding (Binarization)

This is a simple straightforward technique used to create binary images. Binary image is needed because it is easy to detect objects, their shapes and finding contours of those objects. Binary image has only two intensity values 0 (Black) and 1 (White).

Binarized

7.5 Feature Extraction (Finding Contour)

A contour is a continuous curve formed along shape of the object (outline or border). This extracts the important features of the object. Such as its shape, patterns, etc. Gesture recognition is made easier when performed on contours. A convex hull is the smallest polygon formed that contains all the points of the given object (Hand). In this case, outermost pixels in the hand are the edges of the convex hull. (Green outline gives the contour and Red outline gives the convex polygon)



Contour & Convex Hull

8. Mathematical Processing

8.1 Convexity Defects

Convexity defect is a cavity in an object or contour segmented out from an image. That means the area which does not belong to the object but located inside the convex hull (in this case valleys between fingers are convexity defects as they are within convex hull). Finding the number of convexity defects will determine how many fingers are out.

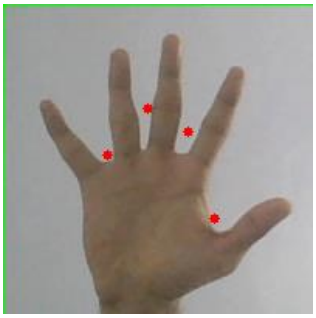
Formulae are,

- If $P(x_1, y_1)$ and $Q(x_2, y_2)$ then Euclidean distance is given as

$$\boldsymbol{d(P,Q) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}$$

- Using cosine rule, find the angle between the fingers

$$\boldsymbol{\theta = \cos^{-1}\left(\frac{b^2+c^2-a^2}{2bc}\right) \times \frac{180}{\pi}}$$



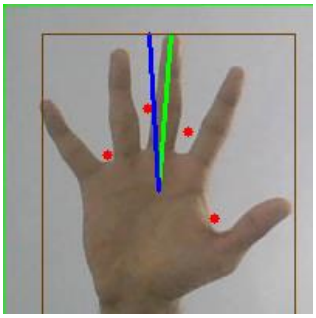
Red dots show the farthest point in the defects

8.2 Angle

Here, three points are considered:

- Farthest point (F) from the centre of the hand (centroid of the convex hull) which lies on the convex hull,
- Centroid (C) of the convex hull, and
- Mid-point (M) of upper side of the rectangle which encloses the entire hand.

Using these points, find the angle made at the centroid by the farthest point. Find the angle using above formulae.



Green line is made by points F and C whereas Blue line is made by points M and C.

8.3 Ratio

Calculate the ratio of area of the hand contour to the area of convex hull. This is necessary because the size of the hand will vary. But every gesture performed will have the consistent ratio.

Formula is,

$$\textbf{\textit{Ratio}} = \frac{\textbf{\textit{area of hand contour}}}{\textbf{\textit{area of convex hull}}} \times \mathbf{100}$$

8.4 Side

This simply determines on which side of the centroid the farthest point is.

if *farthest_point.x < centroid.x* ***then***

Side is Left;

else

Side is Right;

The gesture recognition is based on these four fields. Each gesture performed has its own set of values for the above explained fields. Therefore, it is easier to classify them without ambiguity.

9. Maze Generation

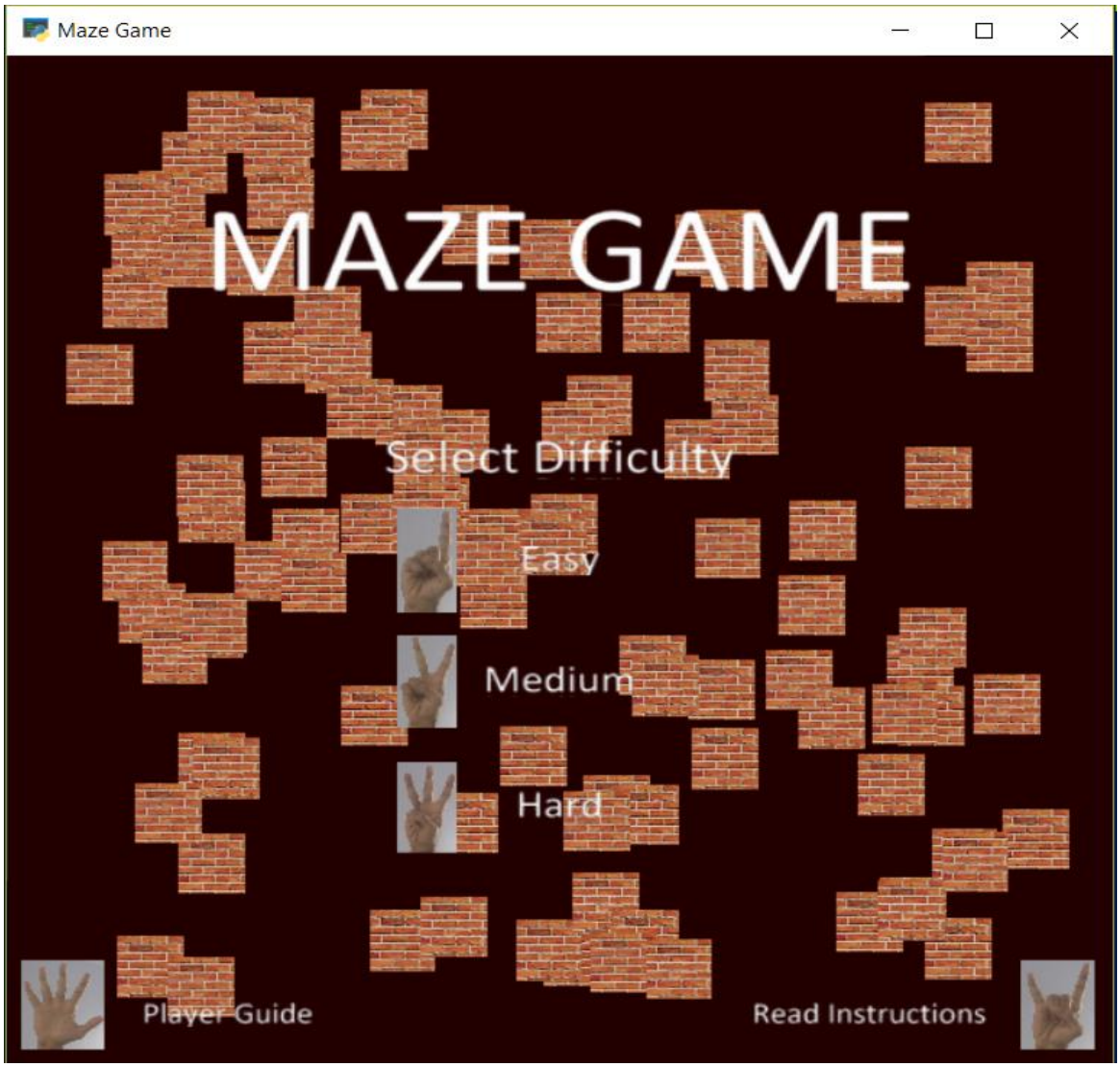
Depth-First Search Algorithm using Backtracking

1. Create an array **Maze** with odd number of rows and columns. 1 means wall and 0 means path. (Origin is at the bottom)
2. Initially the array is set all the cells to 1. Everything is a wall.
3. Select any random cell within the array. Set that cell to 0. Set it as current cell. Let x and y be the current position of the cell i.e. current cell is Maze[x][y].
4. Select any one of the four directions (Up, Down, Left, Right). Let it be D. Always check 2 cells ahead.

```
for number of directions do
    if D = Right and x + 2 < columns and Maze[x+2][y] = 1 then
        Maze[x+1][y] = 0;
        Maze[x+2][y] = 0;
        Set Maze[x+2][y] as current cell;
        Go to step 4;
    else if D = Left and x - 2 > 0 and Maze[x-2][y] = 1 then
        Maze[x-1][y] = 0;
        Maze[x-2][y] = 0;
        Set Maze[x-2][y] as current cell;
        Go to Step 4;
    else if D = Up and y + 2 < rows and Maze[x][y+2] = 1 then
        Maze[x][y+1] = 0;
        Maze[x][y+2] = 0;
        Set Maze[x][y+2] as current cell;
        Go to Step 4;
    else if D = Down and y - 2 > 0 and Maze[x][y-2] = 1 then
        Maze[x][y-1] = 0;
        Maze[x][y-2] = 0;
        Set Maze[x][y-2] as current cell;
        Go to Step 4;
Go to Previous cell;
```

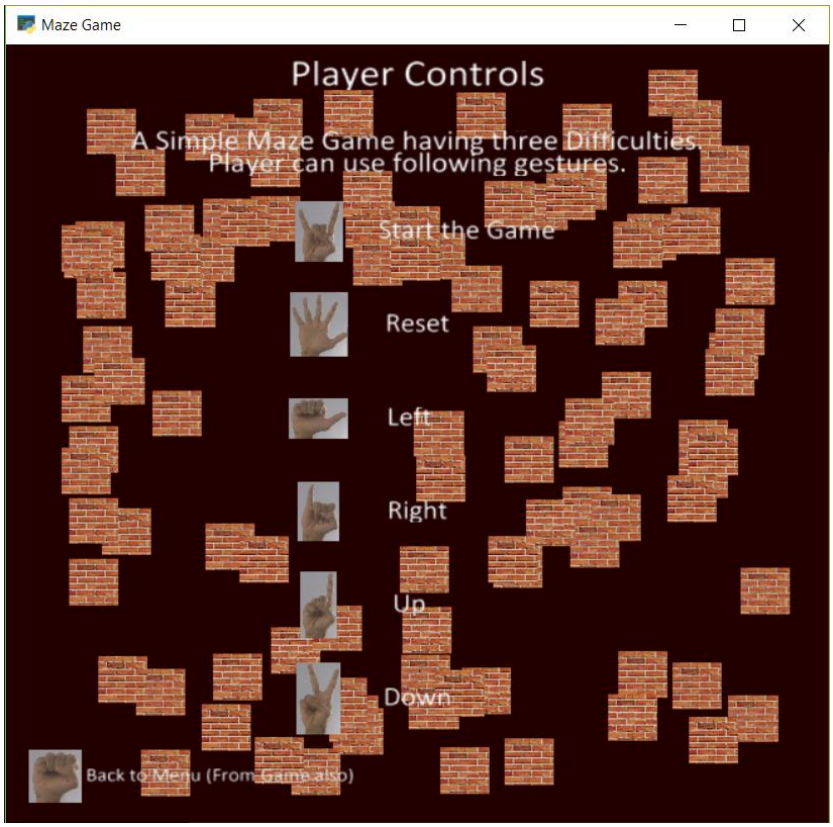
10. Graphical User Interface (GUI)

10.1 Menu Screen

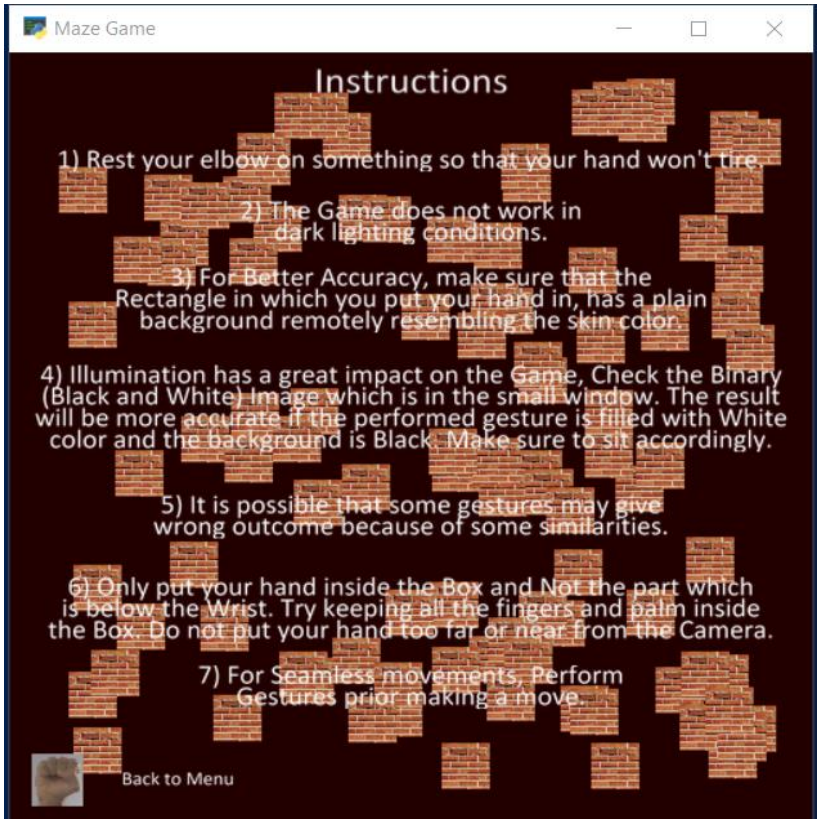


10.2 Input Screens

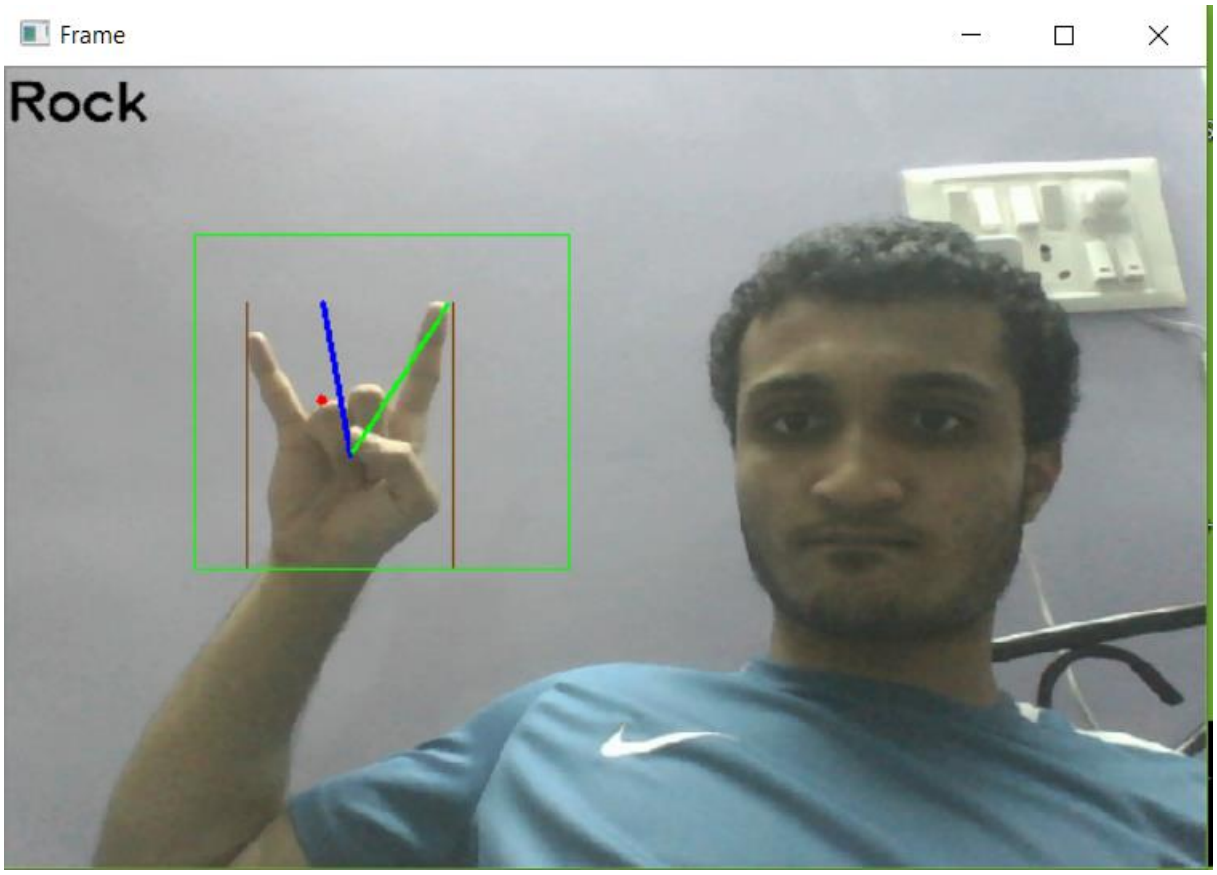
- **Player controls** displays all the gestures and their actions.

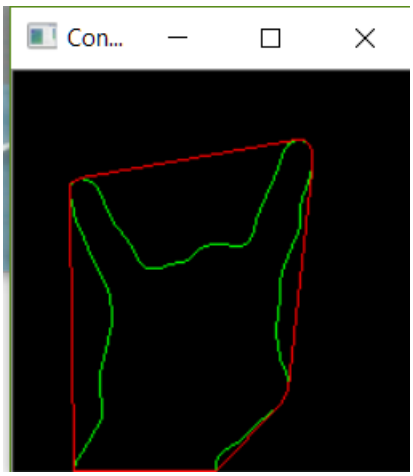


- **Instructions** page shows tips while playing the game.

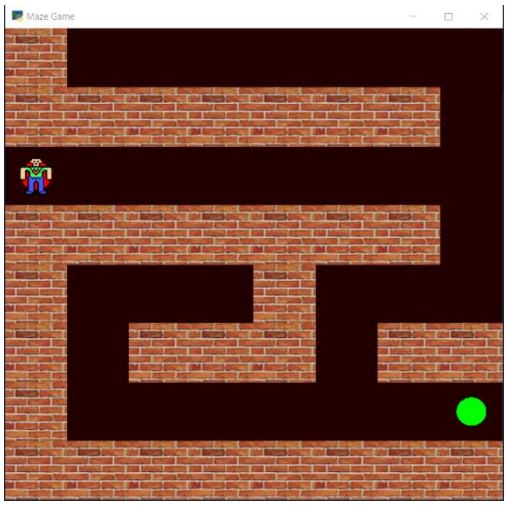


- **Camera Input** Window along with **Binary** and **Contour-Convex Hull** images.



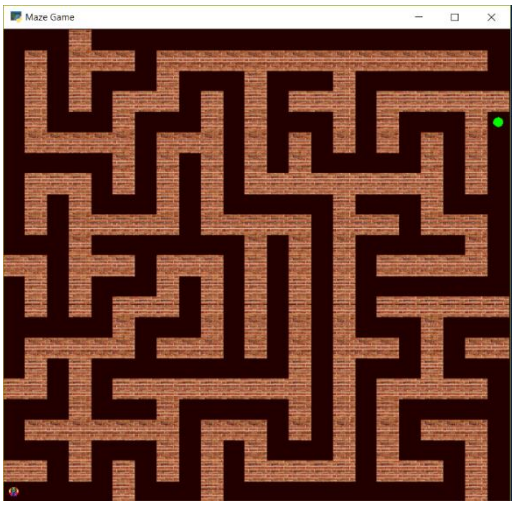


- **Easy** Maze.



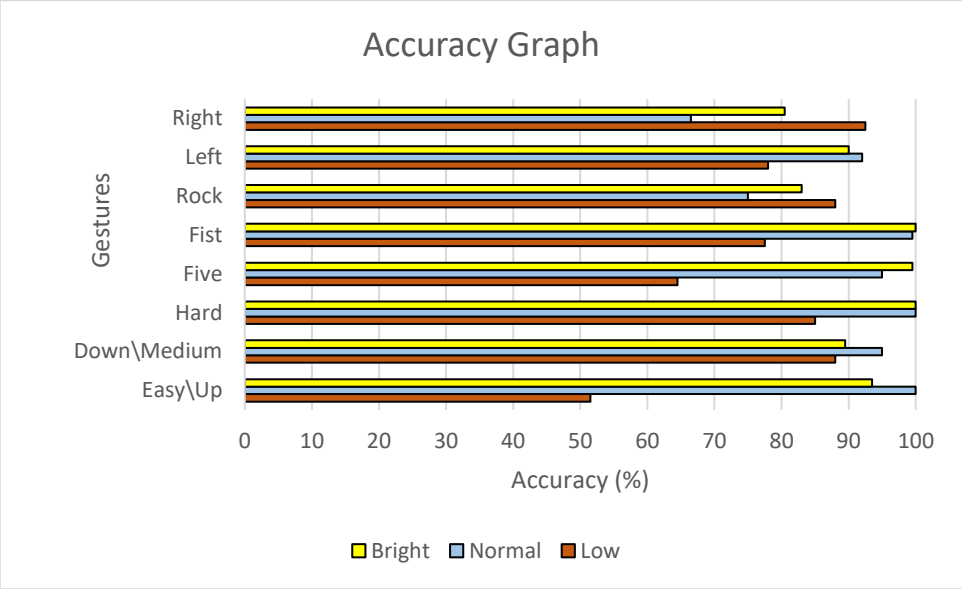
- **Medium** Maze.

- **Hard** Maze

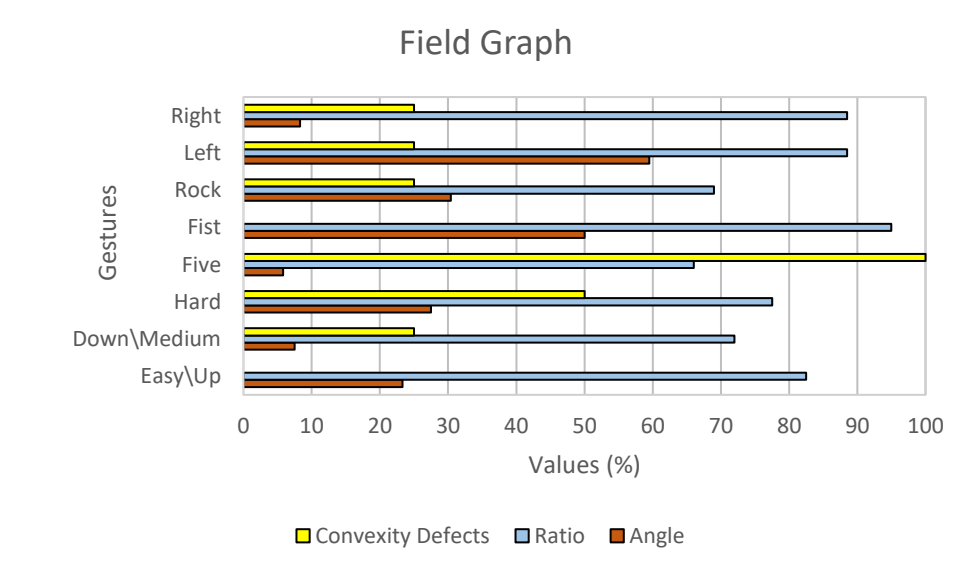


10.3 Graphs

- Following is the Accuracy Graph. Gestures are performed in different lighting conditions. Image is noiseless and the background in every situation has homogeneous colour which is remotely resembling the skin colour.



- Following graph shows different values in percentage (average) for fields used in Gesture Recognition. Maximum Angle in gestures is 120. Maximum convexity defects are 4. Maximum ratio is 100.



11. Testing Strategy

Aim	Testing of code Editor.
Method	Check whether code editor creates, edits and deletes files. Checking if it is saved in proper directory with valid extension.
Expected Result	Python file is created.
Comment	Passed.

Aim	Interpretation of python code.
Method	Check whether the interpreter is executing the code properly and displaying errors. Checking after successful compilation changes are reflected in expected files.
Expected Result	Displaying errors, Changes are reflected.
Comment	Passed.

Aim	Image Acquisition.
Method	Checking if the camera is detected. Checking if the frames are received by the program.
Expected Result	Frames are received.
Comment	Passed.

Aim	Image Pre-processing.
Method	Checking if the images are denoised and converted HSV.
Expected Result	Denoising and Conversion successful.
Comment	Passed.

Aim	Feature Extraction.
Method	Checking if the contour and the convex hull is formed.
Expected Result	Features are extracted.
Comment	Passed.

Aim	Mathematical Processing.
Method	Checking if the angle, ratio and convexity defects are obtained. The formulae used are appropriate.
Expected Result	Angle, ratio and convexity defects are valid.
Comment	Passed.

Aim	Hand Detection.
Method	Checking if the hand is detected in image processing. If not, appropriate message is displayed.
Expected Result	Detection successful.
Comment	Passed.

Aim	Gesture Recognition.
Method	Checking if the processed image is obtained. Checking if the Mathematical processing is done. Checking if the functions returns appropriate action/command according to the gesture.
Expected Result	Frames are received.
Comment	Passed.

Aim	GUI Generation.
Method	Checking if the Hand Gesture Module is loaded. Checking if the media is loaded before displaying windows.
Expected Result	All windows are displayed.
Comment	Passed.

Aim	Maze Generation.
Method	Checking if the selected maze difficulty is loaded from the files as well as Player sprite is set at the starting point on GUI.
Expected Result	Maze Generation successful.
Comment	Passed.

Aim	Move Player.
Method	Checking if the Gestures actions are received. Checking if the player moves accordingly.
Expected Result	Player moved.
Comment	Passed.

Aim	Collision detection.
Method	Checking if the player is colliding with walls. Checking if player does not move through the walls. Checking if the sound is made when collided.
Expected Result	Collisions Detected. Player cannot pass through the walls.
Comment	Passed.

Aim	Run Application
Method	Checking if the camera is detected. Checking if the camera output is show with binary image as well as contour image. Checking if the main menu is shown.
Expected Result	Application is running.
Comment	Passed.

Aim	Conversion of application to executable.
Method	Checking if the required modules are loaded. Checking if all the required data is in proper directories. Checking if all the dependencies are loaded.
Expected Result	Folder with an executable file of the application.
Comment	Passed.

12. User Manual

12.1 Player Controls

Select Easy / Up

Select Medium / Down

Select Hard

Guide / Restart

Start Game / Show Instructions

Back to Menu

Left

Right

12.2 Instructions

- Rest your elbow on something so that your hand won't tire.
- The Game does not work in dark lighting conditions.
- For Better Accuracy, make sure that the Rectangle in which you put your hand in, has a plain background remotely resembling the skin colour.
- Illumination has a great impact on the Game, Check the Binary (Black and White) Image which is in the small window. The result will be more accurate if the performed gesture is filled with White colour and the background is Black. Make sure to sit accordingly.
- It is possible that some gestures may give wrong outcome because of some similarities.
- Only put your hand inside the Box and Not the part which is below the Wrist. Try keeping all the fingers and palm inside the Box. Do not put your hand too far or near from the Camera.
- For Seamless movements, Perform Gestures prior making a move.

13. Drawbacks and Limitations

- The illumination as well as image noise has a great impact on how application performs.
- The accuracy is compromised in computer-vision based applications according to the surroundings. Gesture must be performed in homogeneous background for better results.
- Gestures must be performed bearing in mind the position of your hand.
- The main drawback of gestures is that the same gesture may have more than one meaning.
- While conventional commands only have to be recognized, Gestures need to be known and memorized.
- Mid-air gestures especially involve more muscle involvement, thus leading to fatigue in your hand after some time.
- Quality of the camera also plays an important role.
- Computer-vision based applications do not provide Robust methods.

14.Future Works

- Implement the Hand Gesture Recognition using Artificial Neural Networks (ANN).
- Enhance the Recognition capability for various lighting conditions.
- Achieving more accuracy under heterogeneous backgrounds.
- Implementing Hand Tracking instead of putting your hand inside the box.
- Adding a Feature to add, delete and change gestures.
- Usage in Internet Applications or Web Development.
- Implementation of this game for Mobile Devices.
- Making the game more interactive, 3D and more user friendly.
- Making internet-based game which can be played in the browser.

15. Bibliography

[1] Numpy Tutorial https://www.tutorialspoint.com/numpy/

[2] Opencv Tutorial

- https://docs.opencv.org/2.4/doc/tutorials/tutorials.html
- Opencv computer vision with python, Joseph Howse.
- https://www.learnopencv.com/

[3] Skin Segmentation https://www.pyimagesearch.com/2014/08/18/skin-detection-step-step-example-using-python-opencv/

[4] Convex Hull https://www.learnopencv.com/convex-hull-using-opencv-in-python-and-c/

[5] Convexity Defects

- https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_contours/py_contours_more_functions/py_contours_more_functions.html,
- http://creat-tabu.blogspot.com/2013/08/opencv-python-hand-gesture-recognition.html

[5] Game Development Arcade Tutorial http://arcade.academy/index.html

[6] Maze Generation http://www.migapro.com/depth-first-search/

[7] Readings

- Real –Time Hand Tracking and Gesture Recognition for Human-Computer Interaction, Cristina Manresa, Javier Varona, Ramon Mas and Francisco J. Perales.
- Hand Gesture Recognition Based on Convex Defect Detection; Yanan Xu, Dong-Won Park, GouChol Pok.
- A Study on Hand Gesture Recognition Technique; Sanjay Meena.
- Cursor Movements Controlled By Real Time Hand Gestures; K. Madhuri, L. Praveen Kumar.
- Real Time Hand Gesture Recognition System for Dynamic Applications; Siddharth S. Rautaray, Anupam Agrawal.
- Computer Vision: Algorithms and Applications; Richard Szeliski.
- https://www.slideshare.net/shounakkatyayan/hand-gesture-recognition-42105322
- https://www.slideshare.net/AayushAgrawal25/ppt-of-gesture-recognition.
- https://www.slideshare.net/SurajRai17/gesture-recognition-technologyseminar-ppt

T HANK YOU