

# **“Playing Rock-Paper-Scissors with AI”**

A Project Report submitted to  
JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR  
In partial fulfillment of requirements for the award of the degree of

BACHELOR OF TECHNOLOGY  
IN  
COMPUTER SCIENCE AND ENGINEERING  
BY  
**KAPU TIRUMALA SAI TEJA**                      **(18121A05A3)**

Under the Guidance of  
**Dr. B. Narendra Kumar Rao, Ph.D.,**  
Professor & HOD



Department Of Computer Science and Engineering  
**SREE VIDYANIKETHAN ENGINEERING COLLEGE**  
**(AUTONOMOUS)**

(Affiliated to JNTUA, Anantapuramu)  
Sree Sainath Nagar, Tirupathi – 517 102  
2018-2022



# **SREE VIDYANIKETHAN ENGINEERING COLLEGE**

## **(Autonomous)**

(Affiliated to Jawaharlal Nehru Technological University ,Anantapur) Sree Sainath Nagar, A.Rangampet , Tirupati – 517 102, Chittoor Dist., A.P.

### **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

### **CERTIFICATE**

This is to certify that the Project Work entitled  
PLAYING ROCK-PAPER-SCISSORS WITH AI  
is the bonafide work done by

**KAPU TIRUMALA SAI TEJA**

**18121A05A3**

In the Department of Computer Science and Engineering, Sree Vidyanikethan Engineering College, A.Rangampet is affiliated to JNTUA, Anantapur in partial fulfillment of the requirements for the award of Bachelor of Technology in Computer Science and Engineering during 2018-2022.

This work has been carried out under my guidance and supervision.

The results embodied in this Project report have not been submitted in any university or Organization for the award of any degree or diploma.

**Internal Guide & Head**

**Dr. B. Narendra Kumar Rao, Ph.D.,**

Professor & Head

Dept of CSE

Sree Vidyanikethan Engineering College

Tirupati

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

# **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

## **VISION AND MISSION**

### **VISION**

To become a Centre of Excellence in Computer Science and Engineering by imparting high quality education through teaching, training and research.

### **MISSION**

The Department of Computer Science and Engineering is established to provide undergraduate and graduate education in the field of Computer Science and Engineering to students with diverse background in foundations of software and hardware through a broad curriculum and strongly focused on developing advanced knowledge to become future leaders.

Create knowledge of advanced concepts, innovative technologies and develop research aptitude for contributing to the needs of industry and society.

Develop professional and soft skills for improved knowledge and employability of students.

Encourage students to engage in life-long learning to create awareness of the contemporary developments in computer science and engineering to become outstanding professionals.

Develop attitude for ethical and social responsibilities in professional practice at regional, National and International levels.

## **PROGRAM EDUCATIONAL OBJECTIVES (PEO's)**

1. Pursuing higher studies in Computer Science and Engineering and related disciplines
2. Employed in reputed Computer and I.T organizations and Government or have established start-up companies.
3. Able to demonstrate effective communication, engage in team work, exhibit leadership skills, ethical attitude, and achieve professional advancement through continuing education.

## **PROGRAM SPECIFIC OUTCOMES (PSO's)**

1. Demonstrate knowledge in Data structures and Algorithms, Operating Systems, Database Systems, Software Engineering, Programming Languages, Digital systems, Theoretical Computer Science, and Computer Networks. (PO1)
2. Analyse complex engineering problems and identify algorithms for providing solutions (PO2)
3. Provide solutions for complex engineering problems by analysis, interpretation of data, and development of algorithms to meet the desired needs of industry and society. (PO3,PO4)
4. Select and Apply appropriate techniques and tools to complex engineering problems in the domain of computer software and computer based systems (PO5)

## **PROGRAM OUTCOMES (PO's)**

1. Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems (Engineering knowledge).
2. Identify, formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences (Problem analysis).
3. 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 (Design/development of solutions).
4. Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions (Conduct investigations of complex problems).
5. 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 (Modern tool usage).
6. 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 (The engineer and society)
7. Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development (Environment and sustainability).
8. Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice (Ethics).

9. Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings (Individual and team work).

10. Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions (Communication).

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

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

## **COURSE OUTCOMES (CO's)**

CO1. Knowledge on the project topic (PO1)

CO2. Analytical ability exercised in the project work.(PO2)

CO3. Design skills applied on the project topic. (PO3)

CO4. Ability to investigate and solve complex engineering problems faced during the project work. (PO4)

CO5. Ability to apply tools and techniques to complex engineering activities with an understanding of limitations in the project work. (PO5)

CO6. Ability to provide solutions as per societal needs with consideration to health, safety, legal and cultural issues considered in the project work. (PO6)

CO7. Understanding of the impact of the professional engineering solutions in environmental context and need for sustainable development experienced during the project work. (PO7)

CO8. Ability to apply ethics and norms of the engineering practice as applied in the project work. (PO8)

CO9. Ability to function effectively as an individual as experienced during the project work. (PO9)

CO10. Ability to present views cogently and precisely on the project work. (PO10)

CO11. Project management skills as applied in the project work. (PO11)

CO12. Ability to engage in life-long learning as experience during the project work. (PO12)



## CO-PO 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	PSO 3	PSO 4
C01	3												3			
C02		3												3		
C03			3												3	
C04				3											3	
C05					3											3
C06						3										
C07							3									
C08								3								
C09									3							
C010										3						
C011											3					
C012												3				

**(Note: 3-High, 2-Medium, 1-Low)**

## DECLARATION

I hereby declare that this project report titled "**PLAYING ROCK-PAPER-SCISSORS WITH AI**" is a genuine project work carried out by me, in **B.Tech (*Computer Science and Engineering*)** degree course of **Jawaharlal Nehru Technological University Anantapur** and has not been submitted to any other course or University for the award of any degree by me.

Signature of the student

**1.** KAPU TIRUMALA SAI TEJA



# **SREE VIDYANIKETHAN ENGINEERING COLLEGE** **(Autonomous)**

(Affiliated to Jawaharlal Nehru Technological University ,Anantapur)  
Sree Sainath Nagar, A.Rangampet , Tirupati – 517 102, Chittoor Dist., A.P.

## **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING** **CERTIFICATE**

This is to certify that the Project Work entitled  
PLAYING ROCK-PAPER-SCISSORS WITH AI  
is the bonafide work done by

**KAPU TIRUMALA SAI TEJA**

**18121A05A3**

In the Department of Computer Science and Engineering, Sree Vidyanikethan Engineering College, A.Rangampet is affiliated to JNTUA, Anantapur in partial fulfillment of the requirements for the award of Bachelor of Technology in Computer Science and Engineering during 2018-2022.

This work has been carried out under my guidance and supervision.

The results embodied in this Project report have not been submitted in any university or Organization for the award of any degree or diploma.

**Internal Guide & Head**

**Dr. B. Narendra Kumar Rao, Ph.D.,**

Professor & Head

Dept of CSE

Sree Vidyanikethan Engineering College

Tirupati

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## **ACKNOWLEDGEMENT**

I am extremely thankful to our beloved Chairman and founder **Dr. M. Mohan Babu** who took keen interest to provide us the infrastructural facilities for carrying out the project work.

I am highly indebted to **Dr. B. M.Satish** ,Principal of Sree Vidyanikethan Engineering College for his valuable support and guidance in all academic matters.

I would like to express our indebtedness to the project coordinator,**Dr. K. Reddy Madhavi**, Associate Professor, Department of CSE for her valuable guidance during the course of project work.

I would like to express our deep sense of gratitude to the guide, **Dr. B. Narendra Kumar Rao**, Professor&Head, Department of CSE,for the constant support and invaluable guidance provided for the successful completion of the project.

I am also thankful to all the faculty members of CSE Department, Who have cooperated in carrying out our project. We would like to thank our parents and friends who have extended their help and encouragement either directly or indirectly in completion of our project work.

## **ABSTRACT**

Day-to-day life has become smarter and more intertwined with technology in the modern era. We can chat with AI chatbots we can play with AI machines. In many of the cases AI machines are defeating human players. All age group people are like to play these games. The computer gaming industry has found Artificial Intelligence as a necessary element to make more entertaining and challenging. This is a Rock-Paper-Scissors (RPS) game interaction with AI (Artificial Intelligence). Through collaboration with designers and animators, the framework is designed to be computationally light while also entertaining and visually appealing. To evaluate kinematic hand data in Python, the basic gesture recognition pipeline employs a Leap motion device and two distinct machine learning architectures. This proposed system will provide a powerful application for future research into social human-machine interaction.

## **KEYWORDS:**

Computer Vision, Python, MediaPipe, OpenCV, PyGame

## **TABLE OF CONTENTS**

<b>S.NO</b>	<b>CONTENTS</b>	<b>PAGE NO</b>
1	Introduction	
	1.1 Introduction	1
	1.2 Statement of Problem	
	1.3 Objectives	
	1.4 Scope	
	1.5 Applications	
	1.6 Limitations	
2	Literature Survey	9
3	Analysis	13
	3.1 Specific Requirements	
	3.2 Existing System	
	3.3 Proposed System	
	3.4 Requirements	
	3.4.1 Hardware Requirements	
	3.4.2 Software Requirements	
4	Design	21
	4.1 Design using UML Diagrams	
	4.1.1 Usecase Diagram	
	4.1.2 Class Diagram	
	4.1.3 Sequence Diagram	
	4.1.4 Collaboration Diagram	
	4.1.5 Activity Diagram	
	4.1.6 State chart Diagram	
	4.1.7 Component Diagram	
	4.1.8 Deployment Diagram	

## **TABLE OF CONTENTS**

<b>S.NO</b>	<b>CONTENTS</b>	<b>PAGE NO</b>
5	Implementation	33
	5.1 Approach	
	5.1.1 Build a Hand Gesture Recognizer	
	5.1.2 Working with key points	
	5.1.3 Defining Gestures	
	5.2 System Architecture	
	5.3 Algorithm	
	5.3.1 Movement segmentation Features	
	5.3.2 Shape recognition Features	
6	Execution Procedure and Testing	43
	6.1 Execution Procedure	
	6.2 Testing	
	6.2.1 Objectives of Testing	
	6.2.2 Testing Principles	
	6.2.3 Testing Strategies	
7	Results and Performance Evaluation	46
	7.1 Experimental Results	
	7.2 Performance Evaluation	
8	Conclusion and Future Work	50
	8.1 Conclusion	
	8.2 Future Work	52
9	Appendix	
	9.1 List of Abbreviations	
	9.2 List of Figures	
10	References	54

## 1. INTRODUCTION

### 1.1 Introduction

**Artificial intelligence** (AI) embedded systems that can establish interaction between human-machine via voice, gestures, facial expressions, etc. are gaining popularity now. Using hands as an input is an appealing method for establishing natural Human Computer Interaction among the various interaction techniques. Hand gestures allow users to communicate more information in less time. As a result, in order to improve the interface between users and computers. Human-computer interaction (HCI) technology is widely used. To implement real time computer vision we can use **OpenCV** (Open Source Computer Vision Library). In the types of interaction between human-machine, it is one of the most researched and popular, because in this type of interaction machine itself train the machine to understand natural human language.

Rock- Paper- Scissors (RPS) is a game which is least studied but with a fascinating tactic which is decidedly not for the faint of heart. Its simplicity differentiates it from other well-known games although its influential underlying principles can be applicable in many aspects and have intriguing repercussions. The RPS game is a simple engagement of action and response which has long piqued the interest of researchers among various disciplines. In Sociology, to study the dynamic behavior of collaborative activities and In Biology, to study the evolution of ecosystems RPS patterns are used. And this RPS patterns also helpful in psychology to analyze the various gestures that one can make and will help in revealing their patterns of making gestures in a play or will help in understanding human decision making. As discussed RPS is a game that exchanges information



between an AI system and a human player and this provided the foundation for two most important studies in engineering and robotics. In Engineering, it is about the enhancement of basic technology and In Robotics, it is about its use in developing interaction between human and robots.

**The following are the key principles to remember when playing Human-Human RPS:**

- i) Non-Randomness(Unpredictability)
- ii) Sansukumi

**1.1.i Non-randomness(Unpredictability)**

As we all think RPS contains random making of gestures from one of three hand gestures, but that's wrong RPS does not contain any random movements. The next move is totally influenced by the opponent's actions. Because of this RPS is performed within a time constraint.

- \* If there is no time limit for playing RPS or played very slowly then the players will have enough time to make a random gesture or perhaps to analyze the movements of their own and their opponents. Both of these will lead to the same outcome that may be random moves or having equal chance to win.
- \* If the RPS is performed fast or in a rhythm, players will not get time to make random gestures or to analyze the opponent moves. Hence the triumph depends on one's ability to make quick decisions, understanding the moves that may lead to failure and assess opponent's conduct.

Players play the RPS game in various ways among various regions. But among all those there is a common thing, i.e players will countdown or chant something or sets timer for some particular time. During 1990's in some parts of Japan RPS was popularly played to gain money and the game play will be like choosing the best among

three and it will be completed in 4 or 5 seconds.

### **1.1.ii Sansukumi**

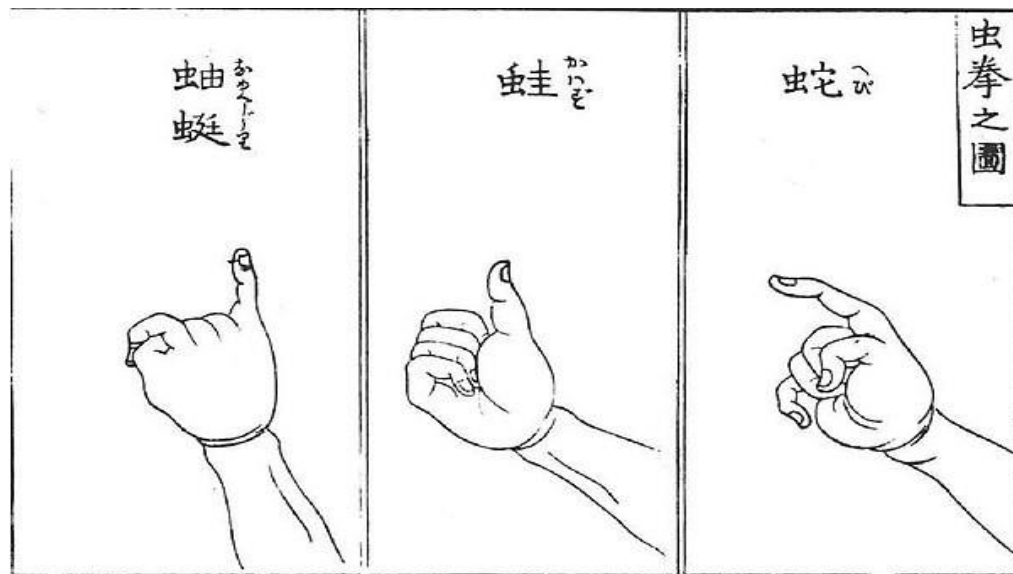
Sansukumi is a Japanese term that refers to three opposing forces that keep each other in check. This phrase comes from China, although it first occurs in Japanese literature, where it has been related with both games and a range of enlightening philosophical concepts. The citizens obey the emperor's orders, the slaves obey the citizenry's directives, and The emperor fears that slaves may revolt against him." for example. Sansukumi is often used to balance computer games; for example, in war games, the Infantry/Cavalry/Artillery sansukumi is widespread.

### **Historical and contextual information**

Games that are played based on gestures are wide group of games which are found at various times throughout the history. A plethora of both modern and historical gesture games are offered by East Asia and among all of them RPS is the most popular and successful.

RPS belongs to the family of games which are found only in Japan. RPS is known in Japan as 'janken.' From the game called mushi-ken, the janken game has originated.

In RPS we will have three options, similar to mushi-ken. So it can be called three option Sansukumi. Also there are many other sansukumi gesture games in Asia that will have 5 or even 4 alternatives. Japan is rich in variants of games that incorporate a broad meta game, based on main sansukumi. One among them is a yakyuken (in Japanese), the speciality of this game is both the players should sing songs and even they have to dance for those in between the throws!



**Fig 1.1** Mushi-hand ken's movements, 1809 depiction.

The gesture games which are not based on the sansukumi will have odd or even results as common, including odds and evens as well as more complicated varieties. It is a fascinating concept to research on does these odd or even games are fully similar to sansukumi games or not. In the history of gesture games, especially Asian sanukumi games are fascinating. The most interesting thing about the RPS is, it contains practically all of the fascinating notions of the whole group of family games that to in very less number of rules. This could be a reason to gain this much popularity.

Janken (a game in japan) has spread throughout the world since the mid-twentieth century but in several names such as Paper, Scissors, Stone, Roshambo, etc. and each one of these will have its types and rules. Janken game gained much seriousness during the 1980s and many extensive kinds of research are done on the strategies used and also several large-scale tournaments are conducted in China and Japan. Regrettably, the game's popularity is gradually reduced over the last 20 years. But still, there are some other large-scale games same as this game, although there are no powerful regulations in Japan or anywhere. Simultaneously, Playing RPS with AI challenge has been expanded...

## **Tactic**

A game plan for RPS is a much more complicated concept that is not impossible to comprehend completely by a single person. We can partition the tactic into three subcategories:

- \* **The Main Tactic:** The crucial strategy or tactic revolves around working to develop a fast and mostly an automatic attitude and behavior in which one can adapt their throws against their opponent without taking much time for thinking in order to be successful more than half the time.
- \* **A tactic for the periphery:** In this, we have to cover timing, strategy for team play, and psychology of opponents
- \* **The meta-game:** Events that may occur around the game area but not depending on the game also have an effect on the final result or achievement

## **The Main Tactic**

Serious players generally model the Main Tactic, not in the random throws of rock-paper-scissors, but in the following ways:

1. The player who won the preceding throw will end up making the same move as before.
2. The player who won the previous time may consider making the move that beats the last move.
3. The player who won the previous round, consider making the move that would end up losing for the last move (i.e., the opponent's earlier move).
4. The player who loses the last throw replicates a move to win.
5. The player who loses a previous round, make a plan for their move which will be preferable to win the round.
6. The player who loses a previous round, make a plan for their move to beat a winning move.

The actual symbols of rock-paper-scissors would not have the same level of resonance as concepts such as "if a player wins then the player cannot

continue with the same move” or “if a player loses then there is no surety that the player will lose every time with the same move”. The above 6 listed options are the building blocks for the RPS game.

### **A Tactic for the periphery**

*A Tactic for the periphery* is a main aspect of the game. In general, a Tactic for the periphery depends on the timing of the gameplay which is about the flow of the game and the ability to handle the draws. And the other one is team play. RPS relies heavily on timing. If the RPS is performed faster there is more chance to make default throws (choosing a throw more often which may lead them to win). Playing RPS faster provides an equal winning rate for both the players. This type of variation is also seen in Japanese and American chants. The difference is,

In Japanese chants, they use “Jan, ken, pon!” and throwing when the third syllable occurred and In American chants, they use “one, two, three, go” and throwing when the fourth syllable occurred.

RPS gains a new dimension with team play. Some people perform worse than others for unknown reasons. Mainly the games that are played as a team will have a huge impact when there is a change in the order of play or swapping the team members.

The thoroughness of the RPS rules is crucial. Rules should not be extremely strict or looser because if the rules are strict, the players who may lose will have few options. If the rules are looser, the players who may lose may divert the opponent by changing hands or making

gestures after watching the opponent hence there will be a high chance of breaking the rules.

Hence, common estimation rules have a magnificent impact. In most cases, RPS is played as best of three sets but not every time. We have to consider the following questions, how many times a player can make

a move before the given time is up? What is the frequency of winning of a dominant player? The RPS game is played seriously when there is a time constraint and played as three or five best of three games and there should be no gap in-between. If the RPS is performed as mentioned then players will get enough time to understand the behavior of their opponent and analyse their throws to win in the game.

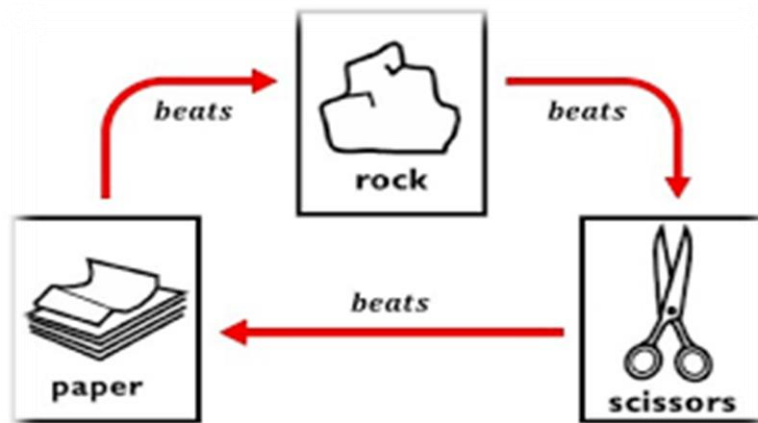
## **Metagame**

RPS game tactic will not be completed without mentioning the “meta-game”. To understand this let’s consider the thing that RPS is oddly the same as golf. The meta-game is nothing but the events or actions that may not be part of the game or depends on some other aspects but this still has a huge impact on the gameplay. Consider the following vital finds:

- \* Experienced players play with more random moves when compared to the serious players, making them both more difficult to win and more difficult to lose.
- \* Players of RPS game are susceptible to “tilt” in poker. It is a proclivity for error that may be caused because of an unfair RPS game. The players who are known as tilted will frequently change their moves or repeat the same patterns unnecessarily since they believe they “ought to” win. These players will take a rest and takes time to restart in team play.
- \* Money: RPS was played for large sums of money as betting or tournament in the last 20 years. Managing money is critical when playing a game and will be added extra worries to all of the players. To avoid an emotional impact on players when they are playing to gain money, money will be handled by the ones who are organized as leaders in a team or players’ friends.

- \* Alcohol : This is unfortunate yet correct that not all of the players are strong physically, and if the game is played in pubs or bars, alcohol is very common. Some analysts, for instance, believe that most of the losing throws are repeated by the RPS players who consumed alcohol.
- \* Bluffing and mind-games: The player approaches their attacker and says, "Planning on throwing out the rock again?" then "will be ready in a minute". Tolerance for this type of tactic varies by location, just as it does in poker.

## 1.2 Statement Of The Problem



Designing a game interface between a human and a system that uses OpenCV techniques and the Python programming language to recognise a person's hand gesture and play against the person by selecting one of three hand gestures.

## 1.3 Objectives

To build a framework for reading human inputs.

- To detect the presence of human hands in the frame.
- To classify whether the hand gesture is a paper, rock, or scissors based on human input.
- To select the victor by using winning criteria.
- To determine the final winner, who first reaches a score of 3.

## 1.4 Scope

This framework for playing Rock, Paper, Scissors (RPS) game with AI embedded system, which let users play with the computer and the future scope of this is really vast as everything getting digital. With little equipment, this can be implemented easily. Additional features can be added like linking this game with robotics and letting robots learn to make hand gestures and this project can be incorporated into many institutions.

### **1.5 Applications**

1. Entertainment purposes for Children, adults, etc. RPS with AI can attract more people than normal RPS.
2. Can be used in training Robots.
3. Can be made into a commercialized product for the children.
4. Can construct images using gestures.

### **1.6 Limitations**

1. Works only on Desktop.
2. Clarity depends on the camera resolution and the lighting.



## 2. LITERATURE SURVEY

Numerous studies have been done that have focus on human motion recognition. They have applied different techniques for analysis and achieved different probabilities for different methods.

**Heike Brock 1, Javier Ponce Chulani<sup>2</sup>, Luis Merino 2, Deborah Szapiro<sup>3</sup>, And Randy Gomez 1(2020),**

“Creating a Scalable Rock-Paper-Scissors Framework for Human-Robot Collaborative Gaming”. The implementation of the Rock-Paper-Scissors game interaction with a computer is presented. With the teamwork of both designers and animators, this game is designed to the final stage. And software is designed to be light to run, as well as good-looking animations were integrated. To guess the random pattern in the hand data on the fly, the gesture detection pipeline uses a Leap motion device and two different machine learning architectures. Human motion is detected by the first architecture and makes RPS play. Whereas second architecture detects the hand sign as Rock, paper, or scissor. The tabletop robot will make its choice to win or lose and also draw states to different animated gestures and vocalizations created by animators. Both learning architectures are trained with different feature and classifier types, and the performance of both architectures is thoroughly examined for accuracy, reliability, and speed. We calculate the approach during a collaborative RPS game between a robot and a human. The system will analyze the user play pattern in the real world. It's a great window to a new way of interaction between a human and a machine.

**H. Brock, S. Sabanovic, K. Nakamura, and R. Gomez(2020),**

"Robust real-time hand gestural recognition for nonverbal communication with tabletop robot Haru," was presented. The hand gesture recognition system is created by Leap motion to train the machine learning architecture for user hand gesture recognition. The existed system will activate based on the speed of the user's hand and movement of the index finger, and with an early detection scheme, it labels the detected movements. The system can recognize gestures without movement limits by combining multiple gestures labels. The system is evaluated using data from simulations of real-world human-robot interactions, we do investigate the behaviour of the system with help of the performance variables like the pattern of movement, timing, and posture.

**Farag and H. Brock (2019),**

" Learning motion disfluencies for automatic sign language segmentation," a novel technique for automatically detecting words within sentence expressions in Sign Language of japan from 3-dimensional body joint positions, was introduced. First, the spaces between line segments of inter-joint pairs are used to determine the flow of the sentence data in sign language within a temporal neighborhood. By the extracted space and time features, a frame-wise binary random classifier is trained to differentiate between words and non-words frames. The classifier's output is used to propose an automatic word generation method that achieves reliable and very accurate sentence formation with a good average frame-wise F1 score of 0.89. The system approach can also be easily adapted to differentiate between motion transitions and motion primitives for a harsh domain, according to the evaluation of the baseline data set.

**C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. Guang Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg, and M. Grundmann (2019),**

Proposed "A framework for making pipelines for perception". They created applications to perceive the world around them even though it's difficult. Developers must select and develop similar machine learning algorithms and models, create some prototypes and demos, maintain a balance between resource consumption and solution quality, and identify and make no severe issues for complex cases. All of these are maintained by the Media-Pipe framework. Developers can use Media-Pipe to make prototypes by combining existing components, then upgrade them to optimize in multi-platform applications while measuring system behavior in performance and resource consumption on different platforms. We give demos on how all these features. Make a developer focus on algorithm or model making while using Media-Pipe as an environment for exponentially developing their application with results that are reliable and reproducible on multiple platforms and devices.

**T. Simon, H. Joo, I. Matthews, and Y. Sheikh (2017),**

Presented a "by using multiview bootstrapping in single images the detection of hand key point." The method of training fine-grained detectors for key points that are prone to blocking, such as hand joint bones, using a multi-camera system. A key point detector is used to generate noisy labels in different aspects of the hand. The noisy detections are then eliminated by triangulation in three dimensions using multiview geometry or marked as different objects as detached from the main subject. Finally, the projected triangulations are used to improve the detector as new labeled training data. This process is

repeated, with each iteration producing more labeled data. We derive an analytical result relating the least number of aspects required to achieve target true and false-positive rates for a particular detector. For single images.

The key point detector operates in real-time on color images and has accuracy comparable to depth sensor methods. The single view detector triangulated in multiple views allows for 3D markerless hand motion capture with some difficult object interactions.

**K. Ito, T. Sueishi, Y. Yamakawa, and M. Ishikawa (2016),**

Focused on "Tracking and recognition of a human hand in a dynamic motion for janken (rock-paper-scissors) robot." The human hand's motion is dynamic so it must be tracked at high speed and images with no blur in them, in order to achieve reliable and exact recognition. For natural recognition, a large area of recognition is required. To accomplish these goals, they built a new advanced sensing system which can track and also recognize the human hand gesture using a system with high-speed vision. The rock-paper-scissors game between a human and a machine requires high-speed recognition, and they focused

on this task to demonstrate the concluded system. In this case, we track a human hand at high speed and correctly detect its sign (rock, paper, or scissors) in each frame. Our tests revealed that the robot

**Y. Hasuda, S. Ishibashi, H. Kozuka, H. Okano, and J. Ishikawa (2007),**

A Robot is produced to play the game "rock, paper, scissors". The game is played by a robot and a human facing each other. To recognize the shape of the user's hand, they created the image processing system. They also programmed the robot to express human-like emotions through its voice, body gestures, and facial expressions that changed depending on the result of the game.

### 3. ANALYSIS

The goal of this chapter is to investigate the specific requirements for hardware, software, design, and its functions.

#### 3.1 SPECIFIC REQUIREMENTS

Google Colaboratory or Jupyter Notebook that supports (Python version 3.7) as we are using Machine Learning and Data Analysis. Google Colaboratory and Jupyter Notebook are more flexible environments for executing these '.ipynb' files. We can even use Python IDLE for executing this.

##### **Functional Requirements**

- \* Accurate hand detection
- \* Accurate Finger detection using Landmarks
- \* Perform mathematical computation
- \* Approximate coordinate calculation
- \* Real-time tracking
- \* Display of result

##### **Non-Functional Requirements**

- \* Scalability
- \* Accuracy
- \* Robustness
- \* Performance
- \* Reliability
- \* Response Time
- \* Maintainability

### **Logical Design:**

Representing the flow of data, inputs, and outputs of a system conceptually are known as the Logical design of a system. Logical design can be applied through modeling and this helps to design the model of the existing system. There is also a discussion about the implementation of system design. In the implementation of logical design UML diagrams are used.

### **Physical Design:**

Compared with Logical design, Physical design is not about the flow but, it is about how the inputs and outputs are processed. It can also be defined as the processing of output from the inputs given to the machine or system. Consider the below system requirements which are crucial in physical design.

1. Input Requirements
2. Output Requirements
3. Storage Requirements

## **3.2 EXISTING SYSTEM**

Python is a multifunctional language that may be used for almost anything. Python can also be used to create video games. Implementation of the Rock-Paper-Scissor game Without utilizing any external game libraries such as PyGame or OpenCV. In this game, the user has the first opportunity to choose between Rock, Paper, and Scissors. After that, the computer selects one of the remaining two options (at random), and the winner is determined according to the rules.

The following are the winning rules:

Consider Human Player as the user and the computer as AI.

- \* If the user makes a paper gesture and AI selects Scissors then AI will win in that throw.
- \* If the user makes a rock gesture and AI selects Scissors then the user will win in that throw.
- \* If the AI selects rock and the user makes a paper gesture then the user will win in that throw.
- \* If AI selects rock and the user also makes the same gesture – rock then it will be tied in that throw.
- \* If AI selects paper and the user also makes a paper gesture then this will also lead to a tie.
- \* If AI selects scissors and the user also makes a scissors gesture then it will also be considered a tie.

In this game, the inbuilt function `randint()` is used to generate random integer values within the given range. The main aspect here is how we provide the computer input or make a choice. It may be selecting one of the three pictures displayed by clicking on it or entering the spelling or number of our choice. However, gamers will find this monotonous because it is merely a simple technique of providing input and there will be very little interaction between both the player and the computer.

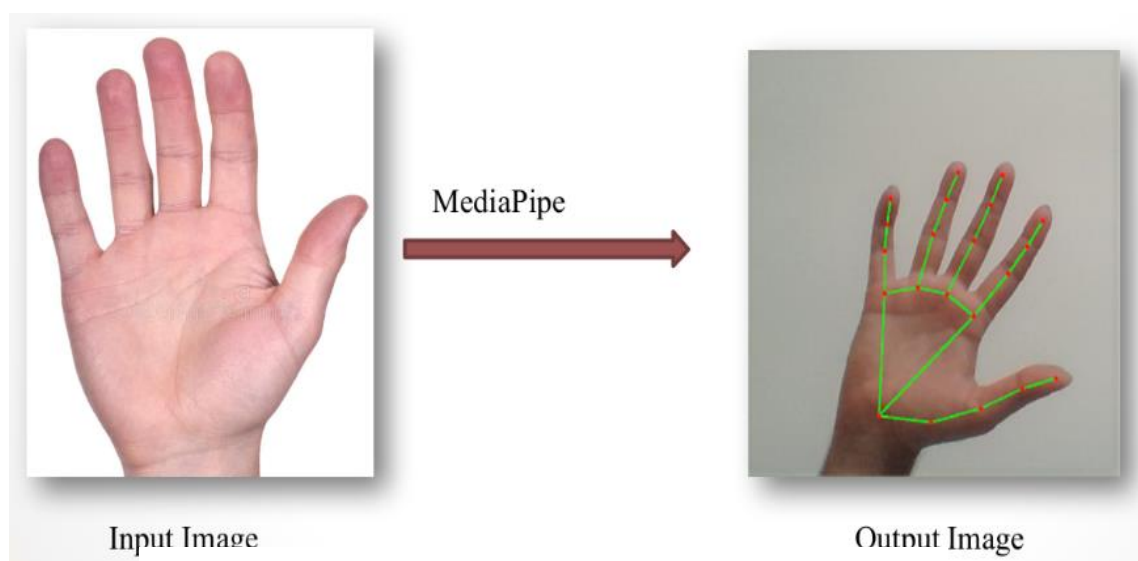


**Fig.3.2.1** Here, players can select from the three images displayed (rock, paper or



### 3.3 PROPOSED SYSTEM

Rock, Paper, and Scissors (RPS) becomes a lot more intriguing thanks to computer vision. It will be a two-way conversation between a computer and a human. Winning Rock, Paper, and Scissors (RPS) is based on random moves, so it is not a model of a winning AI system as no strategy is used. The system's only intelligence would be in the visual recognition of hand signs. To implement real-time computer vision then we can use a library called OpenCV (Open Source Computer Vision Library). OpenCV is compatible with Windows, Linux, macOS, FreeBSD, NetBSD, and OpenBSD. The library contains over 2400 of the top algorithms, including a comprehensive collection of classic and Advanced computervision and algorithms used in ML (Machine Learning). These algorithms and methods are widely used to distinguish and in the detection of faces, objects, or things and categorize the actions of humans in a video, monitor the movements in the camera, and implement 3D from them. MediaPipe library provides effective hand tracking and finger tracking systems, with a higher resolution. With the help of Machine learning (ML) which is a part of AI (Artificial Intelligence), one can point out 21 3D landmarks from the palm in one shot. Related to the Hands module in MediaPipe,



**Fig.3.3.1** Recognizing Hand Gesture using MediaPipe

the palm detection model and Hand Landmark model are present. The researchers created a new model to detect the objects in the frame, which is an optimized model for mobile real-time purposes comparable to the model implemented in MediaPipe to detect faces using Face Mesh, to detect initial palm placements. The diagram below describes the detection of a hand over the complete image and will point out all of the 21 3D coordinates in the detected palm which is a direct coordinate prognostication.

### **OpenCV :**

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as When you use Numpy, a highly efficient library for numerical operations, because all of the operations that we can do using NumPy will be more powerful when it is integrated with OpenCV.

### **Applications.**

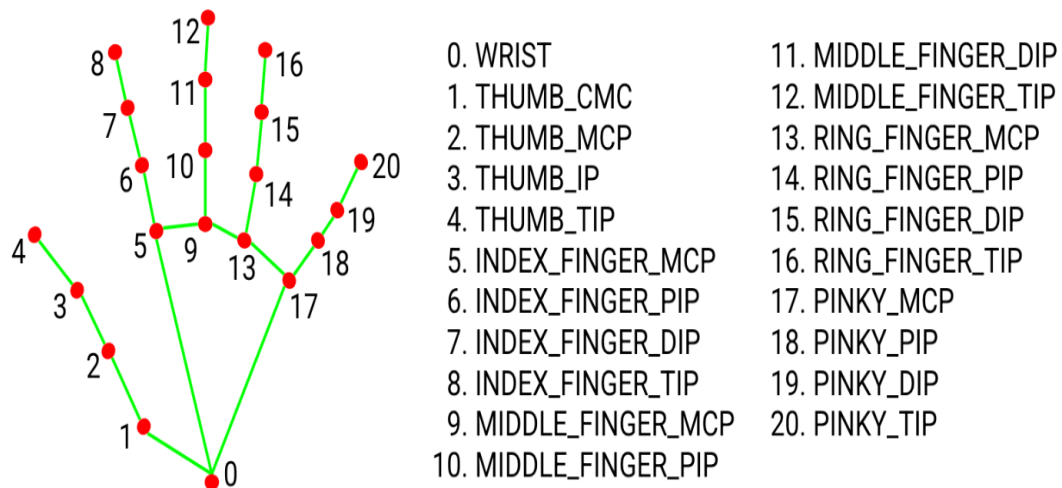
- Color palette with trackbars for OpenCV BGR
- Using OpenCV we can convert an image to a pencil sketch
- With OpenCV, we can display the points on which we clicked from the image.
- OpenCV can be used to give objects as input and extract from them.

### **MediaPipe:**

Hands by MediaPipe is a high-resolution hand and finger tracking solution. Machine learning (ML) is used to deduce 21 3D characteristics of a hand from a single shot. Our solution delivers real-time performance on a mobile telephone, and even scales to several hands, whereas existing state-of-the-art systems rely mostly

---

on powerful desktop environments for inference. We anticipate that making this hand perception capabilities available to the broader research and development community would spur the creation of new applications and studies.



The landmarks present on each finger can be used to track the position of the finger. It can converted to necessary coordinate values in the given image frame.

- \* **static\_image\_mode:** It specifies whether the supplied images should be processed as static images or as video streams. False is the default setting.
- \* **model\_complexity:** It is used to indicate the posture landmark model's complexity: 0, 1, or 2. The precision and latency of landmarks increase as the model's complexity grows. 1 is the default value.
- \* **smooth\_landmarks:** By filtering pose landmarks across distinct input photos, this parameter is utilised to reduce jitter in the prediction. True is the default value.

- \* **min\_detection\_confidence:** It's used to determine the minimum confidence level at which the person-detection model's detection should be judged successful. We can enter any value from 0.0 to 1.0 [0.0, 1.0] and if nothing is specified then its value will be 0.5.
- \* **min\_tracking\_confidence:** It is the lowest confidence level at which the detection of the landmark model can be treated as valid. We can enter any value from 0.0 to 1.0 [0.0, 1.0] and if nothing is specified then its value will be 0.5.

## REQUIREMENTS

Requirements provide a notion of what is required for the proposed system, and they play a critical part in the creation of any system. This chapter explains what application software is and why it's important for the system's development. Software Prerequisites The first step in the software development process is to create a specification. As the system became more complicated, it became clear that the overall aim of the system could not be clearly understood. As a result, the requirement step became necessary. The SRS is a tool for converting client thoughts into a formal document. The SRS's (Software Requirement Specification) goal is to minimize gaps in the communication between clients and developers hence it contains all of the requirements mentioned by the client in a way that developers could understand and reduce the misunderstanding. SRS is a tool that allows

Clients and users to precisely specify their requirements. Any system's need specification can be stated in general terms as follows:

- The system must be able to communicate with other systems.
- It must be trustworthy.

- And it must outperform the current system.

The most creative and hard component of system development is to design the system. The procedure of establishing a device with its architecture, functions, relationships, and input and output data for a computer to meet all of the mentioned requirements utilizing various approaches and concepts.

### 3.4 HARDWARE REQUIREMENTS

**Operating Systems:** Windows 10

**Processors:** Any Intel or AMD x86-64

**RAM:** 4 GB

**Hard disc or SSD:** More than 500 GB

**Camera:** Internal or external

### 3.5 SOFTWARE REQUIREMENTS

**Coding Language:** Python3

**Tools:** Jupyter Notebooks, Anaconda Navigator, Google Colab,  
Python IDLE

**Packages:** Mediapipe, Numpy, OpenCV, Pygame, Time.

### SETUP

#### Jupyter Notebook

1. To install Jupyter Notebook, open Anaconda Navigator and click the Install Jupyter Notebook button.
2. Launch it and open its terminal after it has been successfully installed.
3. From the file menu, select new notebook to start a new notebook.
4. You may also open an existing notebook from the file menu by selecting open notebook.
5. We can now begin coding because all of the relevant files have been generated.

6. The code can be broken down into little cells, making it easier to run.
7. We can run the cells by pressing CTRL+Enter or by clicking the run button.

Once all the cells executed successfully we can see our output.

### **Google Colab**

1. Using Google Chrome, navigate to [www.colab.research.google.com](http://www.colab.research.google.com).
2. Start it up and open the terminal.
3. From the file menu, select new notebook to start a new notebook.
4. You may also open an existing notebook from the file menu by selecting open notebook.
5. We can now begin coding because all of the essential files have been prepared.
6. We can break the code down into smaller cells that are easier to execute.
7. We can run the cells by pressing CTRL+Enter or by clicking the run button.

Once all the cells executed successfully we can see our output.

## 4. DESIGN

System Design is the most innovative and annoying aspect of system development. The technique of the usage of one-of-a-kind methodologies and concepts to establish a device, architecture, modules, interfaces, and data for the device to fulfill mentioned requirements is recognized as "System Design".

The most creative and difficult component of system development is system design. System design is nothing but establishing a device, architecture, interfaces, module, and data for a system that needs to meet specified criteria utilizing various approaches and concepts.

### 4.1 DESIGN USING UML DIAGRAMS

**UML (Unified Modeling Language)** is a standard language for describing, designing, building, and documenting software objects.

The important thing to creating a UML diagram is connecting shapes that constitute an object or class with other shapes to demonstrate relationships and the waft of information and data. By the usage of UML diagrams following goals can be accomplished they're:

- \* To use object-oriented principles to depict full systems (rather than just the software portion).
- \* To create a system that has clear concept coupling as well as executable code.
- \* To consider the scaling characteristics that are inherent in complex and important systems.
- \* To develop a modelling language that can be used by both people and machines.

### **Unified Modeling Language (UML) Building Blocks:**

There are three sorts of building blocks in the UML. They are as follows:

- Diagrams
- Things
- Relationships

#### **Diagrams:**

The diagrams are the visual representations of the models, these included symbols and text. In the context of the UML diagram, each symbol has a particular meaning. Structural Diagrams, Behavioral Diagrams, and Interaction Diagrams are the three types of UML diagrams.

#### **Things:**

Something being a tangible entity or item is referred to as a Thing. There are four different categories of things. Structured Things, Behavioral Things, Grouping Things, and Annotational Things are the four types.

#### **Relationships:**

It shows how things are connected in significant ways. It depicts the relationships between things and defines an application's functionality. Dependency, Association, Generalization, and Realization are the four forms of relationships.

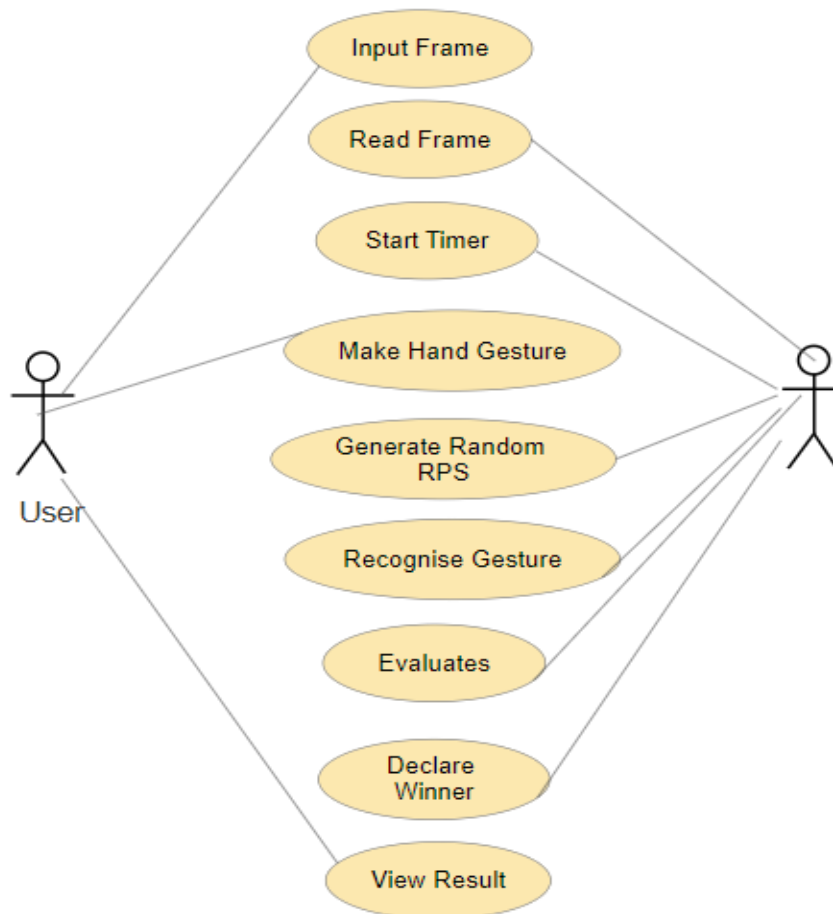
### **4.1.1 USE CASE DIAGRAM**

A use case diagram is a diagram that demonstrates the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their connections. It specifies the activities, capabilities, and functionalities that a framework of an application demands. It reflects the absolute functionality of a system as well as how a user interacts with that as well.



*The accompanying are some guidelines to consider while creating a use case diagram:*

1. The actor or use case of a system should be given a contextually relevant name.
2. An actor's interaction with a use case must be defined in a comprehensible manner.
3. Appropriate notations that will be employed as and when needed.
4. Among the ongoing engagement between the use case and actors, the most significant relationships should be depicted.



**Fig 4.1. Use Case Diagram**

### 4.1.2 CLASS DIAGRAM

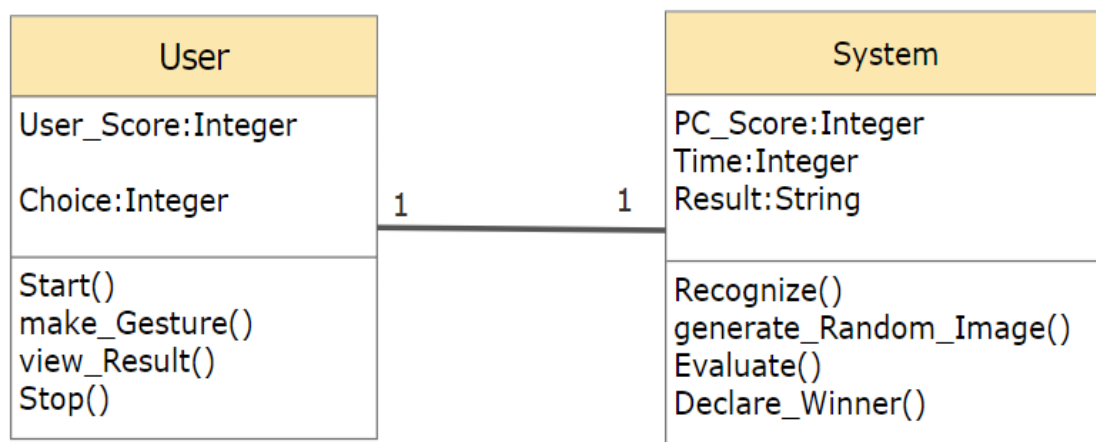
One type of static diagram is the class diagram. It depicts an application's static view. A class diagram is used not only for picturing, defining, and documenting many parts of a system, but also for creating executable code for a piece of software.

A class diagram depicts a class's attributes and operations, as well as the system's limitations. Because class diagrams are the only UML diagrams that can be directly mapped with object-oriented languages, they are frequently utilised in the modelling of object-oriented systems.

A collection of classes, interfaces, affiliations, collaborations, and constraints are shown in a class diagram. A structural diagram is another name for it.

Every class is represented by a rectangle with three sections labelled name, attributes, and operation.

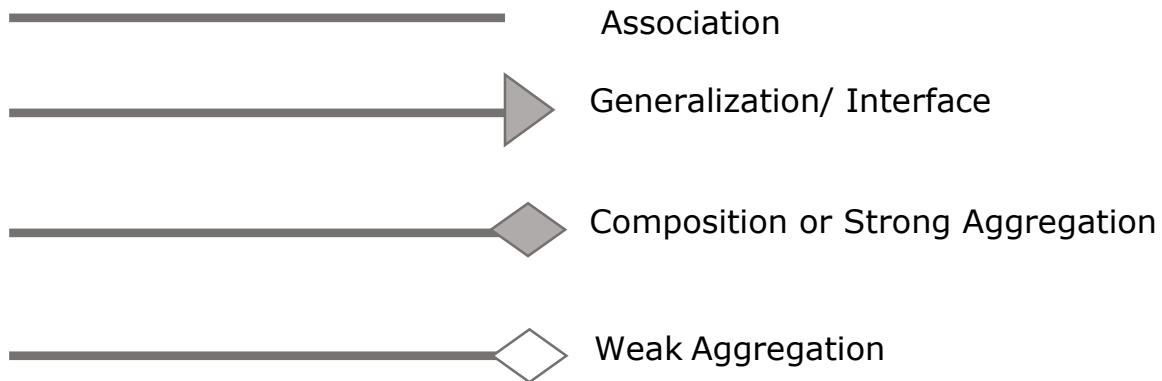
Modifiers are used to determine the visibility of attributes and operations. There are three sorts of modifiers.



**Fig4.2.** Class Diagram

- \* '+' denotes public visibility.
- \* '#' denotes protected visibility.
- \* '-' denotes private visibility

The Meaning of the Arrows in Detail:



#### 4.1.3 SEQUENCE DIAGRAM

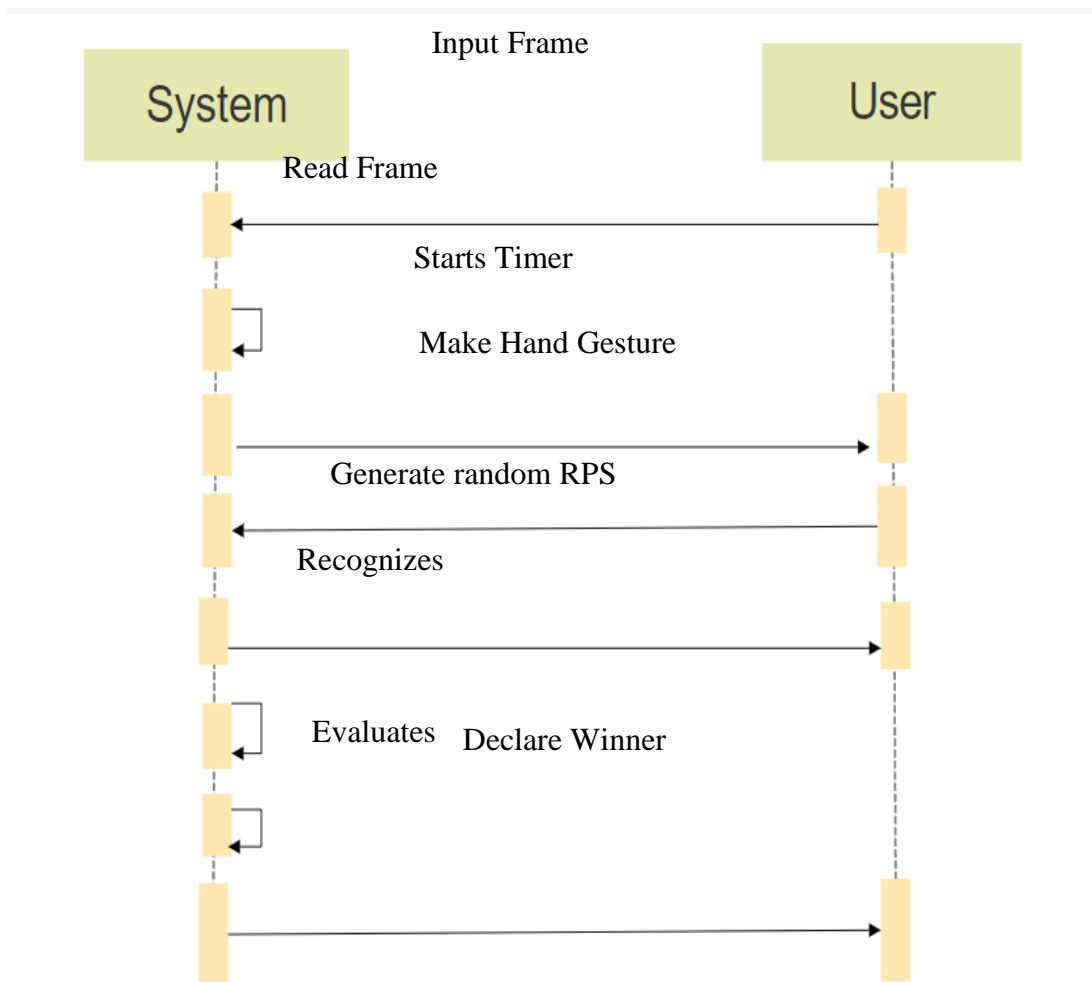
The most popular type of interaction diagram is a sequence diagram.

##### Interaction diagram -

An interaction diagram is a diagram that depicts a system's interacting behavior. We

employ several types of interaction diagrams to capture various features and components of interaction in a system since visualizing the interactions in a system can be difficult.

The flow of messages through the system is depicted in the *sequence diagram*, also known as an event diagram. It assists in the display of a wide range of dynamic environments. It portrays communication between any two lifelines as a time-ordered set of events, as if they were represented simultaneously.



**Fig4.3. Sequence Diagram**

- \* In this Sequence design, User first give the input frame to the system.
- \* The user then begins making hand gestures, at which point the system must generate random RPS.
- \* Now the system must analyse and recognise the features before determining the winner based on the winning rules.

#### 4.1.4 COLLABORATION DIAGRAM

The collaboration diagram depicts the relationship between the objects in a system. The sequence and collaboration diagrams both constitute the same information, but in different ways. It illustrates the architecture of the object residing in the system rather than the flow of messages since it is relied on object-oriented programming. An object has several characteristics. Multiple objects in the system are linked to one another. The collaboration diagram, also known as a communication diagram, is used to depict the architecture of an object in the system.

When object organization is the major focus, the collaboration diagram is employed, and these are better suited for portraying simpler interactions of a smaller number of items.



**Fig4.4. Collaboration Diagram**

#### 4.1.5 ACTIVITY DIAGRAM

Instead of showing the implementation, the activity diagram is used in UML to show the system's flow of control. It can imitate both concurrent and sequential activities. The flow of work from one action to the next is visualized using the activity diagram. It concentrated on the state of flow

**ROCK-PAPER-SCISSORS WITH AI**

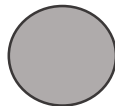
as well as the order in which it occurs. Activity diagrams provide the same basic goals as the other four diagrams. It depicts the system's dynamic behavior. The other four diagrams illustrate message flow from one object to the next, but the activity diagram shows message flow from one action to the next.

The control flow from a start point to a finish point is depicted in an activity diagram, which shows the numerous decision routes that exist while the activity is being performed. An activity diagram can be used to show both sequential and concurrent processing of activities. They are commonly used in business and process modelling to represent the dynamic features of a system. A flowchart and an activity diagram are extremely similar.

Let us consider the notations or symbols used in the activity diagram.

**Notations for Activity Diagrams–**

- \* Initial State-The initial state represents the state that exists before an activity takes place.



Notation of Initial State

- \* Action or Activity State- An activity is the result of an action being performed on or by objects. An activity is represented by a rectangle with rounded corners. An activity is a representation of any action or event that takes place.

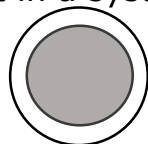


Notation of Activity State

- \* Action Flows or Control Flows — Also known as routes and edges, action flows or control flows are a type of flow. They're utilised to depict the change from one activity state to the next.

—————> Notation of Action Flow

- \* Final State or End State - A Final State or End State is the state that the system reaches when a specific operation or action is completed. In a state machine diagram, the final state is represented by a filled circle within a circle notation. Multiple end states might exist in a system or process.

 Notation of Final State

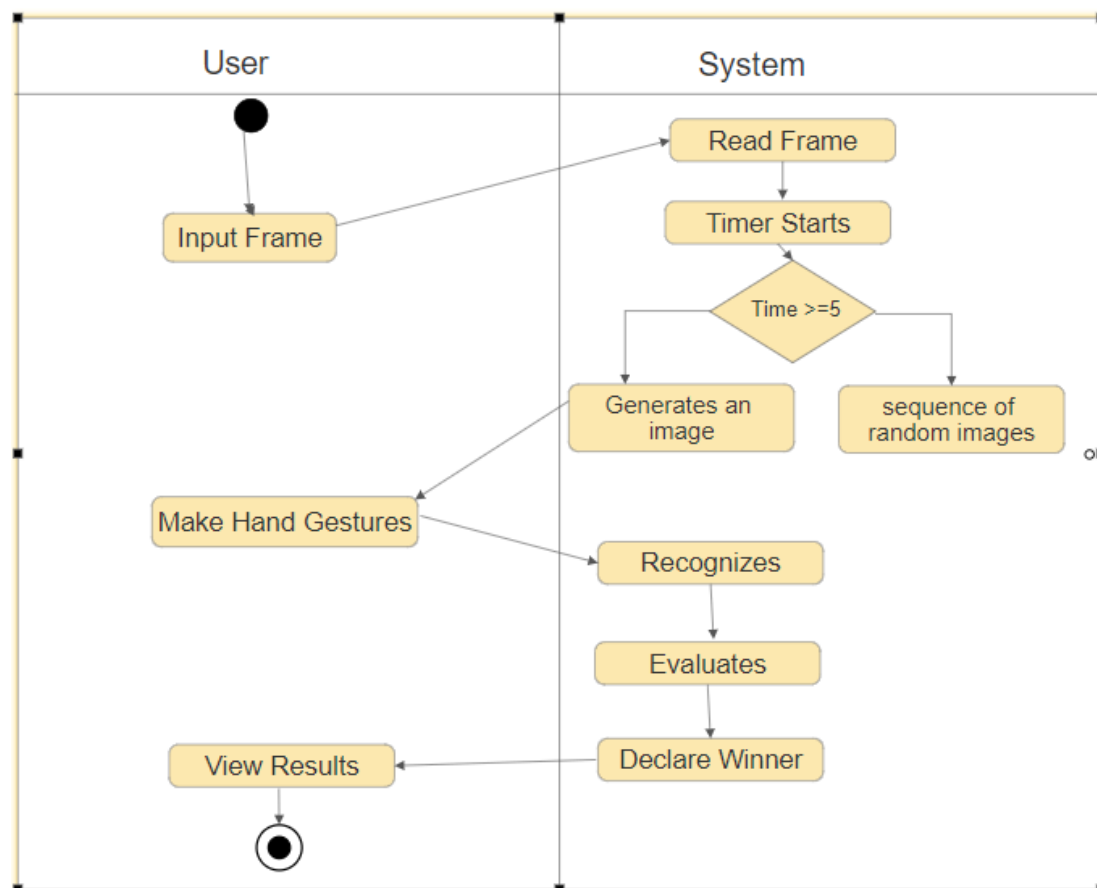


Fig4.5. Activity Diagram

#### 4.1.6 State chart diagram

A state diagram is used to depict the state of a system or a section of a system at a specific point in time. It's a behavioural diagram that uses limited state transitions to illustrate the activity. State diagrams are sometimes known as state machines or state chart diagrams. These terms are frequently interchanged. Simply put, a state diagram is a representation of a class's dynamic behaviour in reaction to time and changing external stimuli. We can say that every class has a state, but we don't use State diagrams to model every class. We prefer to model states in groups of three or more.

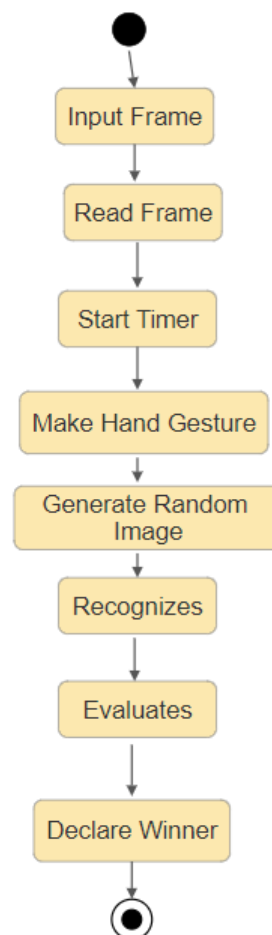


Fig4.6. State Diagram

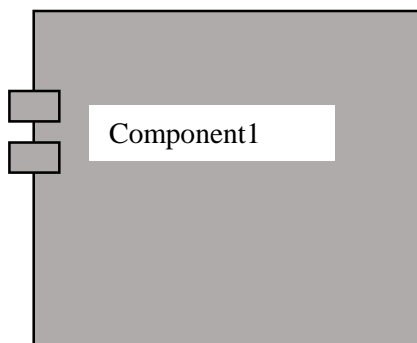


- \* The current state of an event or action is defined by its status.
- \* The start state symbol denotes the start of a procedure.
- \* The end state symbol represents the outcome of a procedure.
- \* Transitioning an operation from one state to another is a response to a specific event.

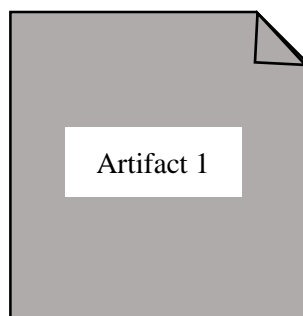
#### 4.1.7 Component Diagram

Component diagrams are used to depict the organisation of system components as well as their interdependencies. They provide a high-level perspective of a system's components. A software component, such as a database or user interface, a hardware component, such as a circuit, microchip, or device, or a business unit, such as a supplier, payroll, or shipping, are all examples of components.

##### Shapes Used in Component Diagram are:



This Notation represents a component



This Notation represents an Artifact

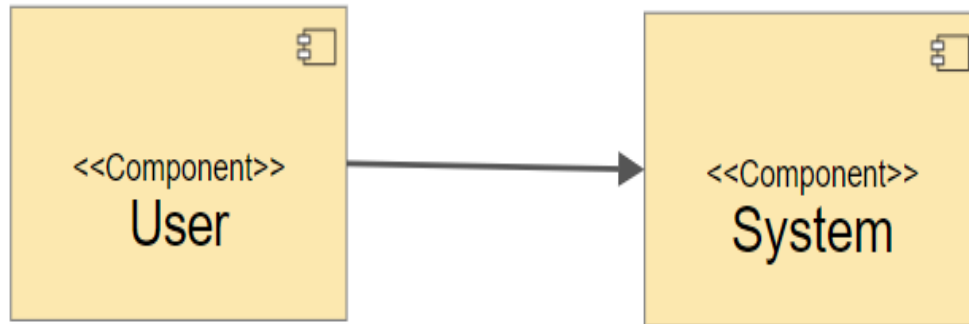


Fig4.7. Component Diagram

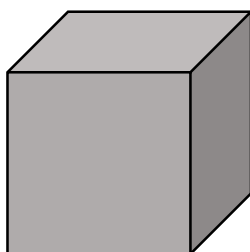
### 4.1.8 Deployment Diagram

The actual hardware that will be used to run the application is depicted in the deployment diagram. It depicts the static deployment perspective of a system. It includes the nodes as well as the connections between them. It controls the manner in which software is deployed on hardware. It links the software architecture defined during the design process to the physical system architecture, where the programme will operate as a node.

The goal of deployment diagrams can be summarized as follows:

- \* Examine the system's hardware topology.
  - \* Describe the physical components that are used to distribute software.
- Specify the nodes that process data at runtime.

#### Shapes Used in Deployment Diagram are:



This Notation represents a Node

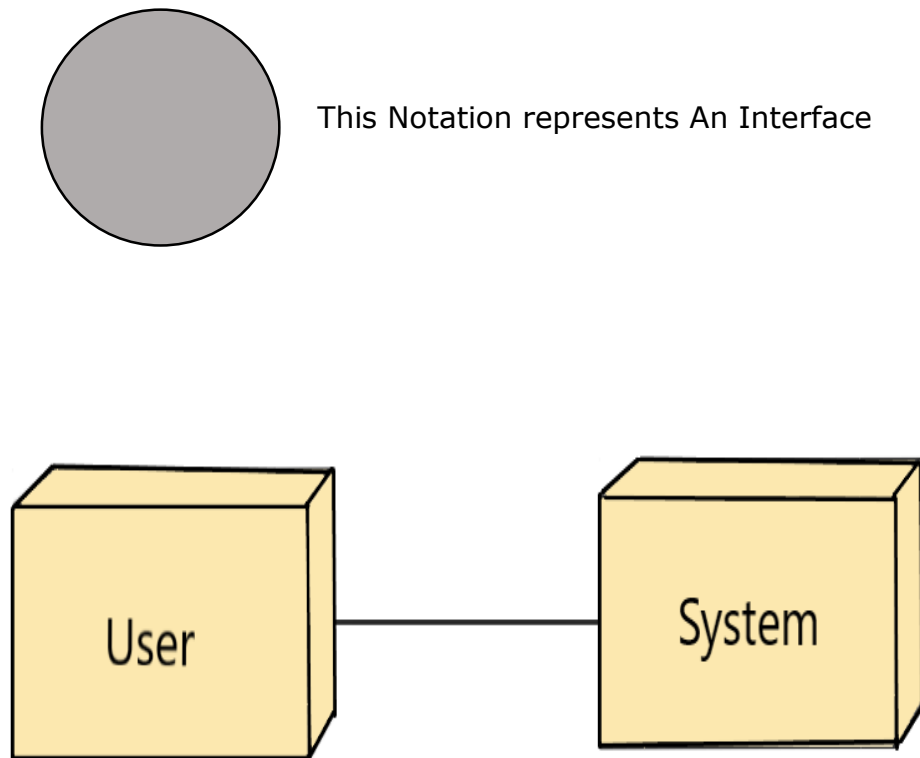


Fig4.8. Deployment Diagram

## 5. IMPLEMENTATION

### INTRODUCTION:

The suggested system design is turned into a working system during the project's implementation stage. As a result, it might be considered the most important stage in achieving a successful output, a new system that is both successful and gives the user confidence that it will perform properly and effectively.

### SETTING UP JUPYTER NOTEBOOK ENVIRONMENT

- \* Install the latest version of Anaconda for python 3.
- \* After installing the Anaconda we can install Jupyter using following command

->pip install jupyter

- \* These commands will install all the necessary folders required for an jupyter project.
- \* It will install all the related dependencies required for the project
- \* To create a new notebook go to FILE in navigation menu and click on new notebook.

File-->new notebook

- \* To open an existing notebook go to FILE in navigation menu and click on open notebook and choose the existing notebook. File-->open notebook
- \* In this the entire notebook is sub divided into cells each cell is having some code
- \* We can execute our cells by using run button on each cell or Press CTRL+Enter.
- \* Once all the cells executed successfully we can see our output.

## SETTING UP GOOGLE COLABORATORY ENVIRONMENT

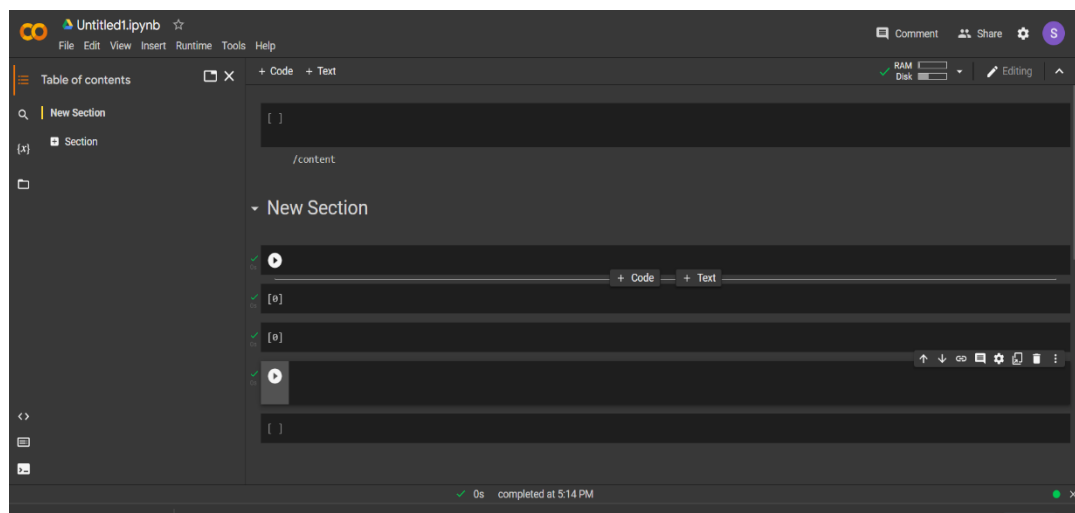
- \* First we have to browse [www.colab.research.google.com](http://www.colab.research.google.com) in Google chrome.
- \* To create a new notebook go to FILE in navigation menu and click on new notebook

File-->new notebook

- \* To open an existing notebook go to FILE in navigation menu and click on open notebook and choose the existing notebook.

File-->open notebook

- \* We can also upload notebook to different platforms like github go to FILE and click on upload notebook.



**Fig.5.1** Google Colab

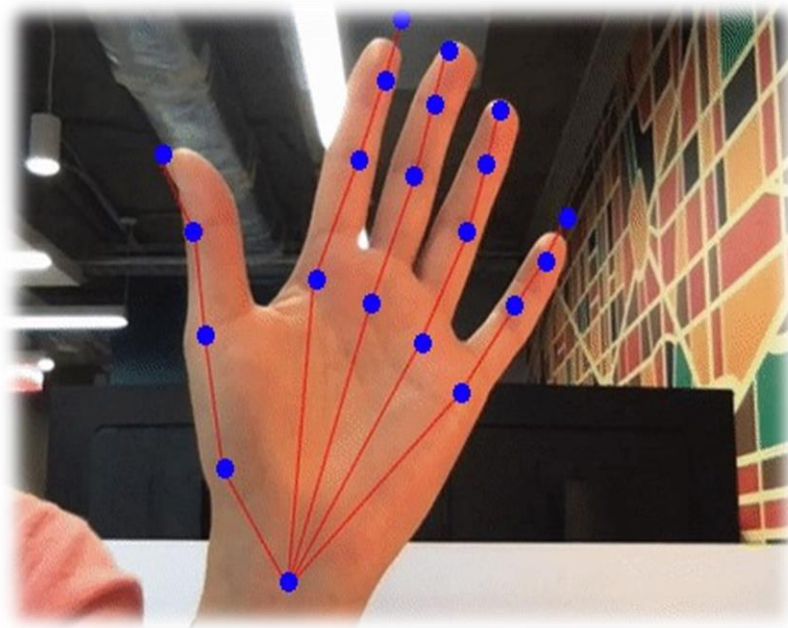
- \* This is how our Google Colab project looks like.
- \* In this the entire notebook is sub divided into cells each cell is having some code.
- \* We can execute our cells by using run button on each cell or Press CTRL+Enter.

## 5.1. APPROACH

- 1 Build a hand gesture recognizer
- 2 Working with key Points
- 3 Defining Gestures

### 5.1.1. Build a hand gesture recognizer

- \* The primary issue in creating a RPS game is to recognise the three hand gestures inside a camera picture 🖐 🖕.
- \* The initial step is to determine whether or not a hand is visible within the camera image.
- \* We estimate the position of all finger joints to trace the hand skeleton if a hand is observed.



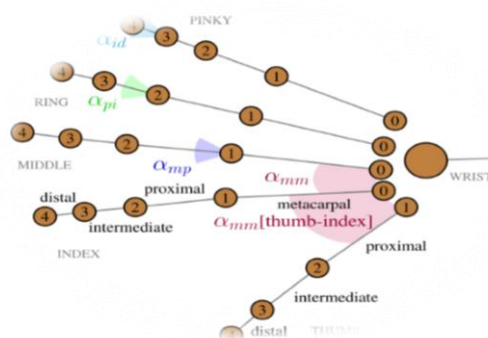
**Fig.5.2** Hand Recognizer

## Function to detect hands in the frame:

```
def findHands(self, img, draw=True):  
    imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)  
    self.results = self.hands.process(imgRGB)  
    if self.results.multi_hand_landmarks:  
        for handlms in self.results.multi_hand_landmarks:  
            if draw:  
                self.mpDraw.draw_landmarks(img, handlms,  
                    self.mpHands.HAND_CONNECTIONS)  
        return img
```

### 5.1.2. Working with key Points

- \* The hand skeleton detector returns the following 21 key locations (sometimes known as "landmarks"): Each finger has four joints, plus the wrist.
- \* The key points are 2D coordinates that inform us where each skeletal point in the image is located.
- \* It is not a very helpful way to describe a hand gesture because it is so difficult to compare two hand movements based on joint position.
- \* A hand can be present in any location in the image, it can be rotated, and people can be left-handed or right-handed.



**Fig.5.3** Key Points

- \* As a result, utilizing natural language to describe a hand motion is a preferable strategy.
- \* As an example, consider the "Thumbs Up" gesture 👍: "All four fingers fully curled and pointed to the left or right" is how it's characterized. The thumb should not be curled or pointed upwards."
- \* A hand gesture can be described considerably more succinctly by using the terms curl and pointing direction. They are unaffected by the hand's size and position in the camera image, and both may be easily determined from the raw 2D coordinates.

### **Function to determine the position of hand:**

```
def findPosition(self, img, handNb=0, draw=True):  
    lmList = []  
    if self.results.multi_hand_landmarks:  
        myHand = self.results.multi_hand_landmarks[handNb]  
        for id, lm in enumerate(myHand.landmark):  
            #print(id, lm)  
            h, w, c = img.shape  
            cx, cy = int(lm.x * w), int(lm.y * h)  
            lmList.append([id, cx, cy])  
        if draw:  
            cv2.circle(img, (cx, cy), 15, (255, 0, 255), cv2.FILLED)  
    return lmList
```

### **The next steps in the detection procedure are as follows:**

Using the key points, describe the curl and pointing direction of each detected finger.

Finally, examine this characterization to a selection of well-known hand motions to see which one matches the best.



### 5.1.3. Defining Gestures

**You basically make a fist in the rock gesture:**

- \* You curl your fingers under and bend them into your palm until the tip of each finger reaches its matching base.
- \* The thumb is next bent down and placed across the top parts of the index and middle fingers.

**A "rock" gesture is defined as follows in this rule:**

- \* All of our fingers are fully curled.

**Gesture for Paper:**

- \* You must spread out all of your fingers and your thumb to produce a "paper" motion.

**Gesture for Scissors:**

- \* The "scissors" sign looks a lot like a "winning" sign. The index and middle fingers are extended. The ring and pinky should be curled in half or completely.

## Code to count the number of fingers opened:

```
if len(lmList) != 0:
    fingers = []      # Thumb
    if lmList[tipIds[0]][1] < lmList[tipIds[4]][1]:
        if lmList[tipIds[0]][1] < lmList[tipIds[0] - 1][1]: # if 4 is on the left
            of 3 then it is closed)
            fingers.append(1)
        else:
            fingers.append(0)
    else:
        if lmList[tipIds[0]][1] > lmList[tipIds[0] - 1][1]: # if 4 is on the left
            of 3 then it is closed)
            fingers.append(1)
        else:
            fingers.append(0)
    for id in range(1, 5): # other fingers
        if lmList[tipIds[id]][2] < lmList[tipIds[id] - 2][2]: #landmark 8 and
            6 and get the value of the y (the max value at the max height is 0 : start
            from the top)
            fingers.append(1)
        else:
            fingers.append(0)
    # print(fingers)
    totalFingers = fingers.count(1)
```

## 5.2 SYSTEM ARCHITECTURE

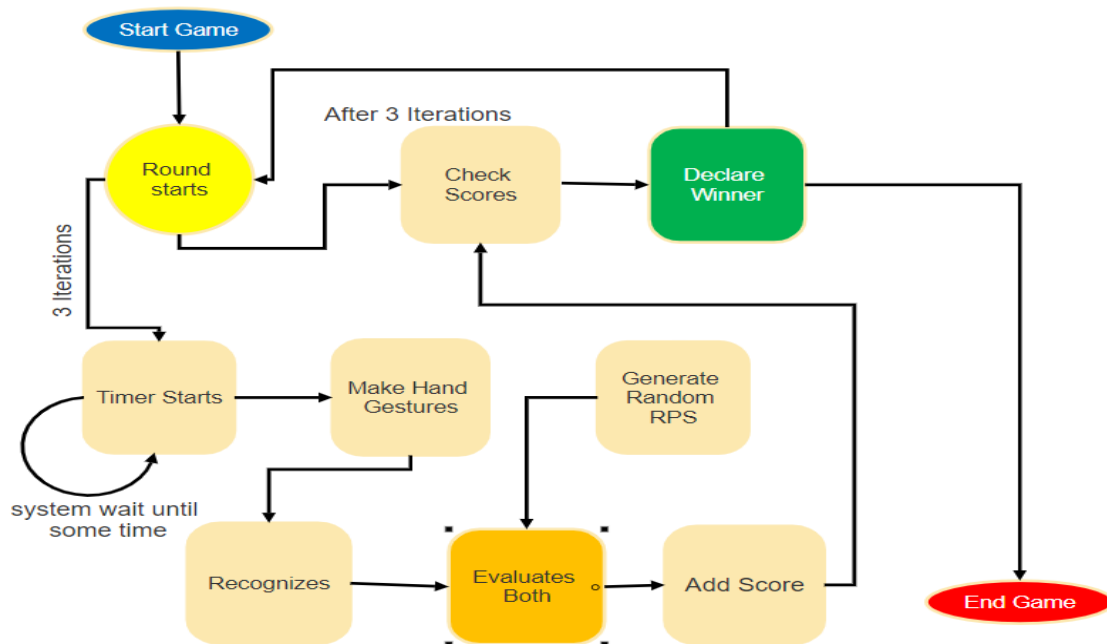


Fig.5.4 System Architecture

## 5.3 ALGORITHM

### FEATURE TRACKING AND PROCESSING :

Simple and robust features should be available for detecting and tracking faults in the submitted hand data. Engineer the features to be retrieved quickly and readily to avoid computational overload and overall signal processing delays.

Furthermore, all features are based only on the hand joint locations to prevent reliance on the internal feature computing procedures.

- 1) MOVEMENT SEGMENTATION FEATURES
- 2) SHAPE RECOGNITION FEATURES

### 5.3.1 MOVEMENT SEGMENTATION FEATURES:

Create a collection of feature representations The basic velocity and direction vectors of a player's  $\vec{v}_j$  hand are used by FSeg. These describe the velocity of any joint representing the velocity in 3D space

$$\vec{v}_j = \begin{bmatrix} v_{j_x} \\ v_{j_y} \\ v_{j_z} \end{bmatrix},$$

whereas J denotes the set of palm and fingertip joints, and  $\vec{u}_d$  represents relative changes in the vectors  $d \in D$  as

$$\vec{u}_d = \begin{bmatrix} u_{d_x, i-1} \\ u_{d_y, i-1} \\ u_{d_z, i-1} \end{bmatrix} - \begin{bmatrix} u_{d_x, i} \\ u_{d_y, i} \\ u_{d_z, i} \end{bmatrix},$$

I [2,..., n], where n is the length of a motion sequence, and D denotes the set of normal and direction vectors of the palm. The final feature set is defined as follows:

$$\mathcal{F}_{\text{Seg}} = \{\vec{v}_j, \vec{u}_d\} \quad \text{for all } j \text{ and } d \text{ respectively.}$$

### 5.3.2 SHAPE RECOGNITION FEATURES

The angular and distal relations of the classes 'Rock, Paper, and Scissors' can be uniquely defined using the positions of finger joints in 3-dimensional space.

The collection of attribute representations The normalised inter-joint angles computed from a hand's raw joint positions are represented by FShape.

Compute the angle  $\alpha$  of the length-normalized inner product of the two  $\vec{b}_{J_3J_4}$  vectors  $\vec{b}_{J_1J_2}$  and for all the neighboring finger bones represent the respective bones (J1, J2) and (J3, J4) as

$$\alpha = \arccos \frac{\overrightarrow{b_{J_1J_2}} \cdot \overrightarrow{b_{J_3J_4}}}{||\overrightarrow{b_{J_1J_2}}|| \cdot ||\overrightarrow{b_{J_3J_4}}||}.$$

This gives us the angles mm between metacarpal finger bones (for the thumb and index finger, respectively proximal and metacarpal bone), as well as the angles 'amp' between metacarpal and proximal bone, 'api' between proximal and intermediate bone, and 'aid' between intermediate and distal bone of each finger.

- \* Combine the angles to the final feature set
  - o FShape1 = {amm, amp, api, aid}.
- \* For subsequent comparison, furthermore, build a second set
  - o FShape2 = {app[thumb-index], amp, api}
  - o with reduced morphological information.

## 6. EXECUTION PROCEDURE AND TESTING

### 6.1 Execution Procedure

We'll need some tools, and a server to complete the job. The following are the requirements and procedures:

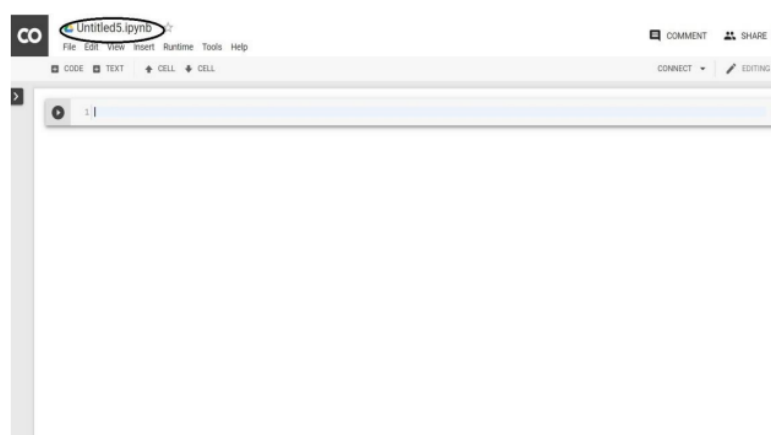
To complete the project, we can also use Google colab.

Google Colab is a free cloud-based service that now includes free GPU support! Setting up an environment with Anaconda in a computer is time-consuming although it has a GPU and troubleshooting the issues of installation is also tough. But this will be resolved by using Colaboratory which is a free Jupyter notebook developed by Google.

Working with Google Colab is easy, first, we need to sign in to a Google account and then navigate to the website "<https://colab.research.google.com>."

#### Opening Jupyter Notebook:

When you open the website, you will see a pop-up window with the following tabs.



**Fig 6.3.1** Jupyter Empty Notebook

### **Notebook's Description:**

To generate a new Jupyter notebook, click on the option create a new notebook with the name Untitled).ipynb and we can save that file to Google Drive in some folder. Because all Jupyter notebook commands work in Google Colab. However, there will be more information in the original document of "Getting started with Jupyter Notebook". Select "Runtime" from the dropdown menu. Choose "Change runtime type." Now, in the "Hardware accelerator" dropdown menu, select whatever you want (GPU, CPU, None).

To begin the game, run or execute the Finger counting.py file.

## **6.2 Testing**

When developing any Software Product considering Software Testing is very crucial. Software Testing assures the software quality. In any aspect maintaining the software quality is the main thing and it can be possible only when any software is tested thoroughly. But Testing thoroughly is not possible but if we can test against the requirements then it will be efficient Software Testing. Testing is a procedure of finding errors or bugs and flaws in the product.

There are multiple types of tests. Each type of testing will have its own specific requirement to check. The goal of testing is very simple, it is used to find the flaws or errors in the product. Let us consider some types of testing, Unit testing is about testing each module separately. Integration testing is about testing the multiple modules by combining the individual modules. Testing ensures that the developed product will meet customers' expectations.

### **6.2.1 Objectives of Testing:**

- \* Testing refers to the procedure of executing a software program to detect flaws.
- \* A test case can be called a good test case when it discovers an error that is not detected still.
- \* The test suite (set of test cases) will be called successful when it uncovers various types of errors in the shortest amount of time and with the least amount of effort. A secondary advantage of testing is that it shows that the software appears to be working as specified in the specifications.
- \* The data gathered during testing can also provide insight into the software's dependability and quality. However, testing does not ensure that there is no defect in the developed product.

### **6.2.2 Testing Principles:**

- \* Before starting to test, test cases should be designed first.
- \* Testing a product thoroughly is impossible.
- \* An Independent third party should test the product to become a more efficient test case.
- \* Test cases should be in a way to trace back to end-user requirements.

### **6.2.3 Testing Strategies**

A software testing strategy incorporates software test cases into a set of well-planned procedures that lead to a successful build. Verification and validation are two terms for the same thing in software testing. Verification is the process of ensuring that software performs a specific function accurately. Validation is a set of operations that ensures that the software that has been produced can be traced back to the needs of the customer.

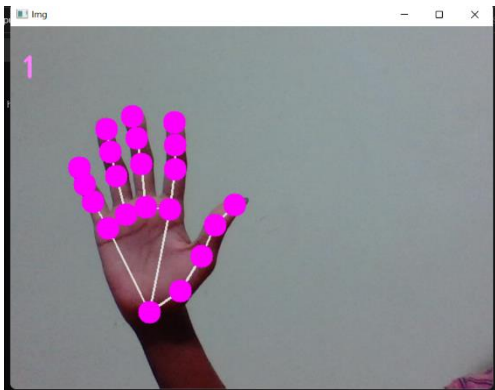


## 7. RESULTS AND PERFORMANCE EVALUATION

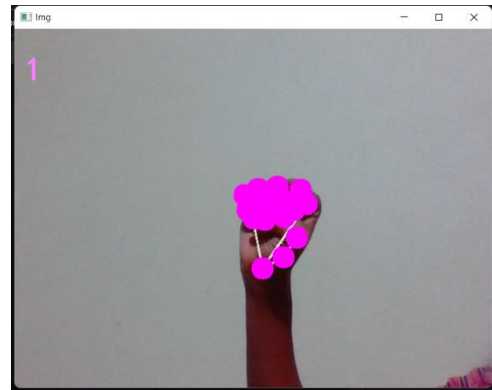
### 7.1 EXPERIMENTAL RESULTS

This application enables the user to construct images in air using their fingers. The landmarks of the hand are detected and images are constructed by keeping track of the position of the finger.

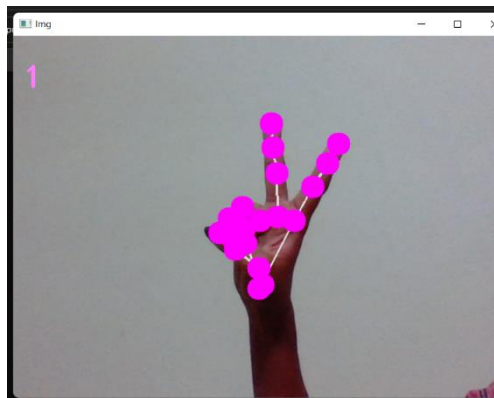
When the Hand\_tracking\_module.py is run or opened:



**Fig 7.1.1 Paper**

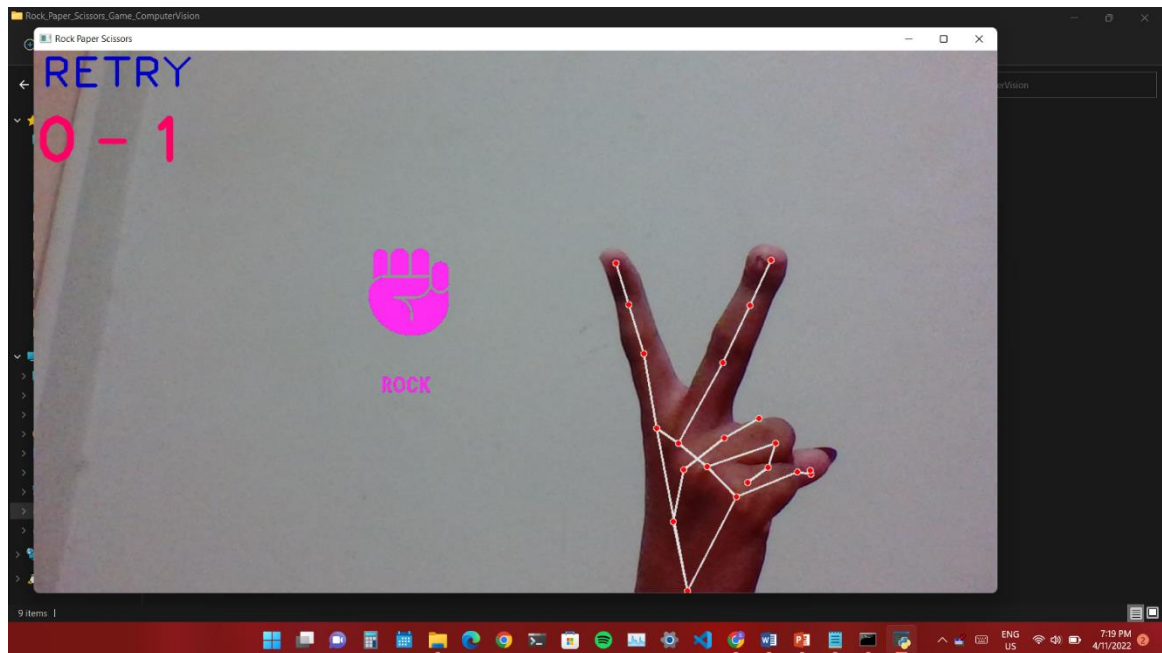


**Fig 7.1.2 Rock**

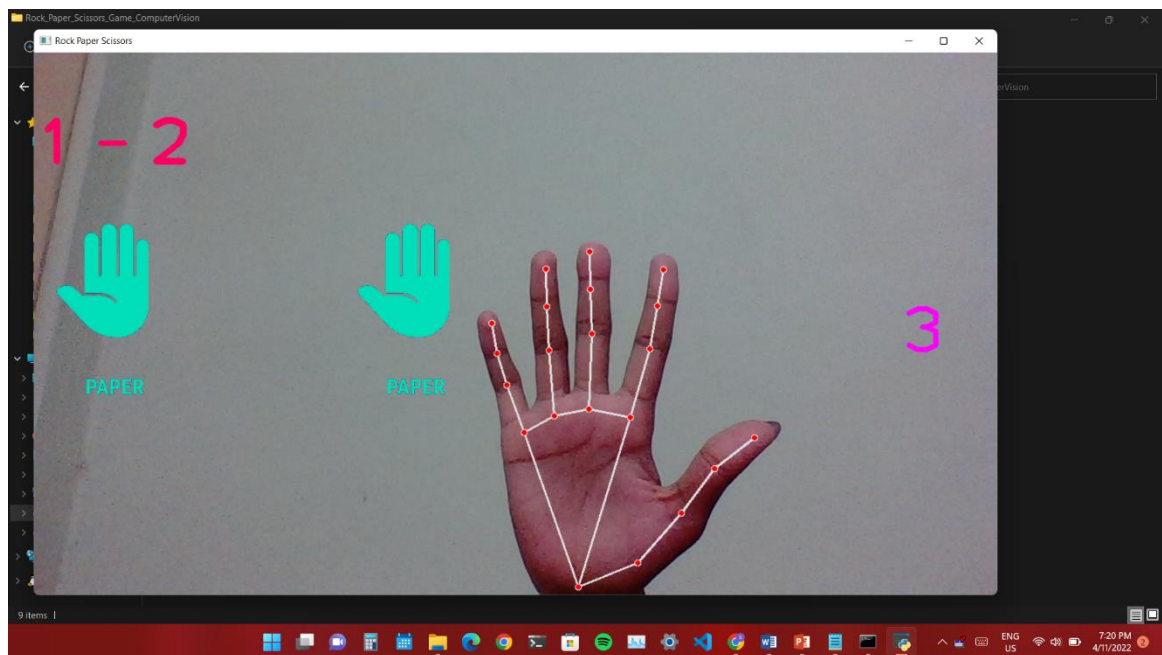


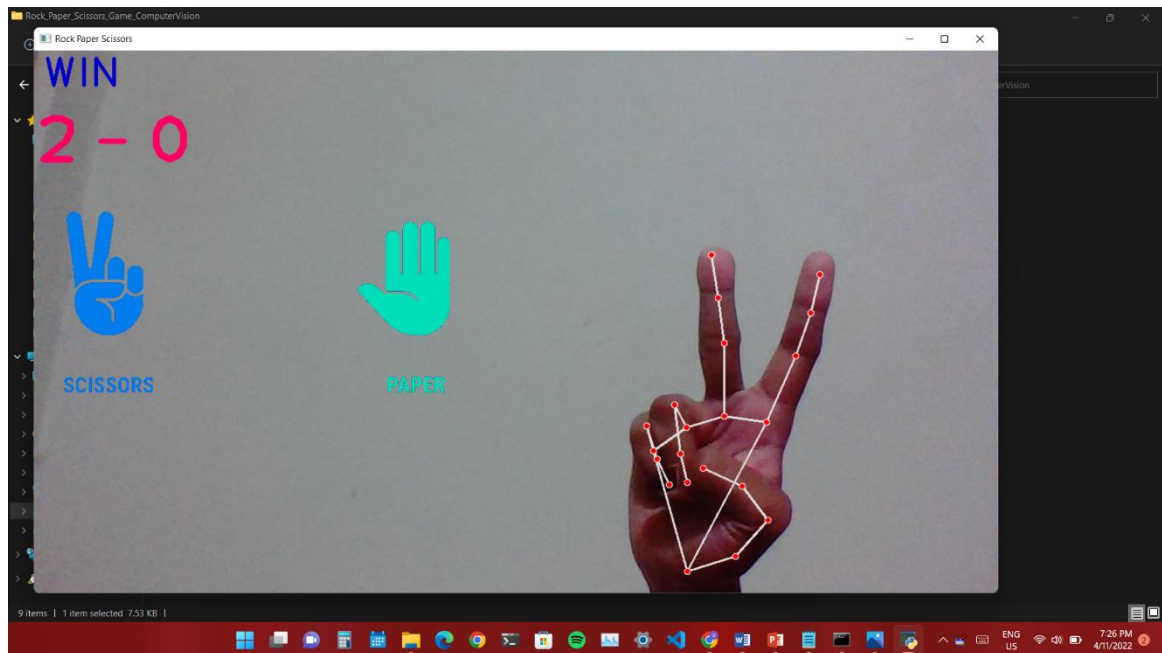
**Fig 7.1.3 Scissors**

When the finger\_counting.py is run or opened:



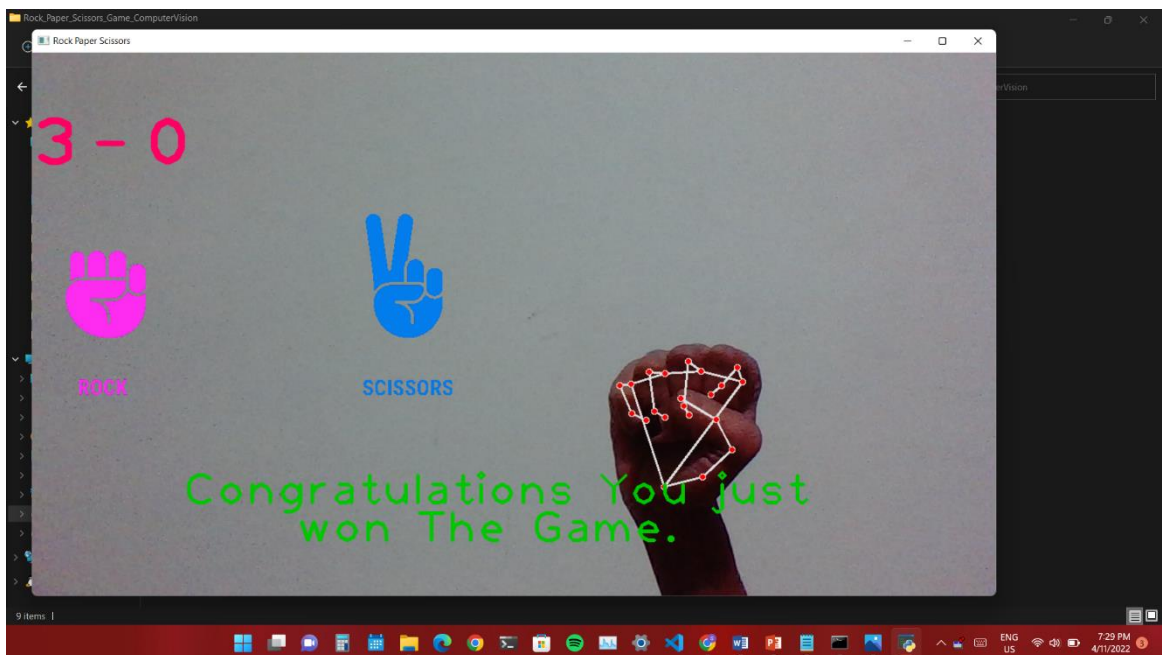
If the system could not identify or human player makes hand gesture after the stipulated time then it will show "**Retry**"





This image shows that the system is recognizing PAPER gesture

This image shows that the system recognizes Scissors gesture and it beats the paper gesture made by system.



This image shows the Declaration of Final Winner

## **7.2 PERFORMANCE EVALUATION:**

On both desktop and mobile devices, MediaPipe provides models with high accuracy and low latency. It gives a 3D Hand Landmark model that uses machine learning techniques to predict 21 points from a single picture and can be used on desktops, mobile devices, and browsers, among other platforms. The palm detection model has a precision of 95.7 percent on average.

## **8. CONCLUSION AND FUTURE WORK**

### **8.1 CONCLUSION:**

An OpenCV and MediaPipe-based RPS(Rock-Paper-Scissors) is proposed in this paper. The palm detection model and hand landmark model were used to identify the hand and its landmarks. OpenCV has in-built image processing algorithms which preprocess the frames captured and display output as required by the user. Therefore, the application provides an interactive environment between the human player and AI (computer).

### **8.2 FUTURE WORK**

Future work based on this project has a much broader scope. A few are listed here:

- \* This application can be converted into a mobile application.
- \* This can be enhanced by linking this software with the Robotic Hand to play against the user.
- \* This will be helpful in training the Robots to make relevant Hand Gestures.
- \* In this we can include Machine Learning models to predict the user and generate hand gestures according to that, other than making random gestures.

## **9. APPENDIX**

- 9.1 List of Abbreviations / Nomenclature
- 9.2 List of Figures

## 9.1 List of Abbreviations / Nomenclature

S.No	Abbreviations	Full Form	Page No
1)	AI	Artificial Intelligence	1
2)	OpenCV	Open Source Computer Vision Library	1
3)	HCI	Human Computer Interaction	1
4)	RPS	Rock-Paper-Scissors	1
5)	ML	Machine Learning	16
6)	SRS	Software Requirement Specification	19
7)	UML	Unified Modeling Language	21

## 9.2 List of Figures

<b>S.No</b>	<b>Program Name</b>	<b>Page No</b>
4.1.1	Use Case Diagram	23
4.1.2	Class Diagram	24
4.1.3	Sequence Diagram	25
4.1.4	Collaboration Diagram	26
4.1.5	Activity Diagram	28
4.1.6	State Diagram	29
4.1.7	Compound Diagram	31
4.1.8	Deployment Diagram	32
5.4	System Architecture	40



## 10. REFERENCES

- [1] Heike Brock<sup>1</sup>, (Member, IEEE), Javier Ponce Chulani<sup>2</sup>, Luis Merino<sup>2</sup>, (Member, IEEE), Deborah Szapiro<sup>3</sup>, (Member, IEEE), And Randy Gomez<sup>1</sup>, (Member, IEEE), "Developing a Lightweight Rock-Paper-Scissors Framework for Human-Robot Collaborative Gaming" Oct. 2020.
- [2] H. Brock, S. Sabanovic, K. Nakamura, and R. Gomez, "Robust real-time hand gestural recognition for non-verbal communication with tabletop robot haru," in Proc. 29th IEEE Int. Conf. Robot Human Interact. Commun. (RO-MAN), Aug. 2020, pp. 891–898.
- [3] R. Gomez, K. Nakamura, D. Szapiro, and L. Merino, "A holistic approach in designing tabletop robot's expressivity," in Proc. IEEE Int. Conf. Robot. Automat. (ICRA), May/Aug. 2020, pp. 1970–1976.
- [4] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. Guang Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg, and M. Grundmann, "MediaPipe: A framework for building perception pipelines," 2019, arXiv:1906.08172. [Online]. Available: <http://arxiv.org/abs/1906.08172>
- [5] I. Farag and H. Brock, "Learning motion disfluencies for automatic sign language segmentation," in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), May 2019, pp. 7360–7364.
- [6] R. Gomez, D. Szapiro, K. Galindo, and K. Nakamura, "Haru: Hardware design of an experimental tabletop robot assistant," in Proc. ACM/IEEE Int. Conf. Human-Robot Interact., Feb. 2018, pp. 233–240.
- [7] I. A. S. Filho, E. N. Chen, J. M. da Silva Junior, and R. da Silva Barboza, "Gesture recognition using leap motion: A comparison between machine learning algorithms," in Proc. ACM SIGGRAPH Posters, New York, NY, USA, 2018, pp. 1–2. S.
- [8] T. Simon, H. Joo, I. Matthews, and Y. Sheikh, "Hand keypoint detection in single images using multiview bootstrapping," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 1145–1153