# Different Strategy's & Performance Analysis on Rubik's Cube Puzzle Solver

**Sumeer Anand, Student, Vidyalankar Institute of Technology, Mumbai, India**

**Shivam Bhatt, Student, Vidyalankar Institute of Technology, Mumbai, India**

**Pritik Jain, Student, Vidyalankar Institute of Technology, Mumbai, India**

**Prof. Shruti Agarwal, Assistant Professor, Vidyalankar Institute of Technology, Mumbai, India**

**Abstract: The 3D-puzzle referred to as Rubik's cube have enticed many people since its release in mid 1970s. With 43 quintillion possible combinations the Rubik's cube present a posh problem that needs both reasoning and memorization. Recently Rubik's Cube community has seen an increase of robots which will automatically solve a Rubik's Cube. Similarly, Application will also help to solve the cube. The objective is to design an application that can teach a human or train a human to know the steps while approaching a rubik's cube and to undergo this phase different technologies have been represent like the person need to solve a rubik cube (unsolved/scrambled cube) show all the six faces the application can take to pictures of all the six sides and using algorithms the cube gets solved and all the actions of rotation are showed to the person so he can try to solve to cube physically by its own. Therefore, it might be helpful to possess a program which will read within the current state of the Rubik's Cube and suggest an algorithm for solving it. In order to recognize colors on each face of the cube, Image processing was used. For image processing we used OpenCV.**

***Keywords – Image Processing, Artificial Intelligence (AI), Colour Recognition, Cube Solve, Rubik Cube, Kociemba Algorithm, Friedrich Method, Two Phase Solver, CFOP Method***

## I. INTRODUCTION

Stumbling upon a difficult problem to solve is a rubik's cube. Due to their complexity, and in some cases lack of information on this subject. With the help of technology helping the person who gets stumbled to solve these problems and analyze various technique. We decided to make this process to be solved in helpful approach. By breaking the technique in parts will help to access different technique to solve the solution. Rubik's cube solver is an autonomous system which requires accuracy or in other words requires perfection. The most important part of solving a Rubik's Cube is understanding how it works. Rubik's cube is one of the most challenging and most famous puzzles of all time. It continues to remain in the top of among the puzzle games because of the design of the puzzle i.e., one correct solution out of the 43 quintillion other possibilities. An average person takes 48 to 100 tries to solve the Rubik's cube and requires more time. Only 5.8% of world population can solve the Rubik's cube efficiently. But AI & ML concept is able to do so in under a second with minimum number of moves.

Our aim is to develop an autonomous system which is capable to identify colors on the scrambled cube, generate string, developing array apply various algorithms on it and then solve it in minimum time and minimum number of steps. Solving any Rubik's cube has three major parts. First is identifying the positions of different colors at different positions. Second is to develop a series of steps which can be used to solve the cube and third is to implement these steps on the cube to get the final result i.e. to get final solved cube with the development and implementation of image processing and development of algorithm is performed to solve the cube. This paper deals with all these processes that are involved. Image processing plays the most silent role of all, and thus its accuracy is considered very vital. Identifying the colors is essentially the first process and generation of algorithm is the next step.

## II. PROBLEM STATEMENT

There are many puzzles solving game to approach like Sukudo, Chess. But Rubik cube is a game where the are many chances of failure can happen and had happens for beginners or even some know people thus it's very important of some of the unsolved mesh and go with some required procedure and try to figure the mistaken you have made. Hence, we have developed and android application to help you out to solve the cube with different feature with the help on image recognition the algorithm can help the user to help the cube solver in minimal steps. The proposed

 DOI : 10.35291/2454-9150.2022.0085 

system is to improve efficiency of the person to make this project interesting and helpful and knowledgeable to understand the trend in a rubik cube and gather the technique to solve it. The system will be image processing based where the image will pe used to detect the colour of the small cuboids inside to cube. The image will be captured by the person of the cube or else he can manually input the color inside it.

## III. LITERATURE SURVEY

A literature survey in a report is the section which shows the various analyses and research made in the field of project and the results already published, considering the various parameters of the project and the extent of the project. It is the most important part of report as it gives a direction in project domain. It helps to set a goal for analysis. This section contains analysis of eight papers and the future scope of the papers which we can implement in our project. [1] The authors have discussed image processing of detecting all the corner and the whole cube to make and animated paper and for this all face of being clicked and get inserted to the application the small cuboid or blobs are the main for detecting the face of the cube. Further the Filtration of image to get it redefined will be used to get helped for fetching the color in the cube as before the image clicked will be converted in black and white scale.[2] The authors have discussed various methods for the detection of Rubik's cube. Parameters like color recognition and its saturation are taken into consideration. As the cube is six sides figure configuring 6 colors. Colour will help to solve to cube as it types of representation that all colour gets matched of each face of the cube. Different strategy are calculation Like the image chopped, image corelate, and it saturation and gradient.[3] The author have discussed about the solving the rubik's cube in a algorithmic way as the process is little different is used as same fundamentals same of the cube but the conclusion will be same to solve the cube and similarly the cube will be solved.[4] The author has discussed different terminologies and orientation layout used in Rubik's Cube theory and analysis the structure of the cube and the function of solving the cube the layer architecture is shown of all the layer and next mixed permutations & combinations ordering the faces in the cube. [5] The author have discussed the mapping to image and getting reconstructed variety of cube design has been discussed and mapping those cubes in mechanism and further carried out the desired output. [6] The author has discussed the method of solving the cube in programmable way Different algorithms are being explain two phase solver, CFOP method are used to solve the cube. [7] The authors have discussed proposes an algorithm that solves the Rubik's cube in an optimal way this algorithm is having different chronology it follows that it consider the all the face of cube as a group and to generate different n number of combination it sub divides the cube into subgroups. [8] The author has discussed the way to solve the cube in formation for example the top layer has been rotated twice but to check before rotation the middle or the lower layer doesn't get disturbed to get solved this way have been knowledge and different algorithms explained

## IV. PROPOSED SYSTEM

Technology helping the person who gets stumbled to solve these problems and analyze various technique. We decided to make this process to be solved in helpful approach. By breaking the technique in parts will help to access different technique to solve the solution. Here we have decided to approach the app such a manner that there will be four sections.

- Camera Input
- Manual Input
- Scrambler Generator
- Cube Timer (1v1 cube game )

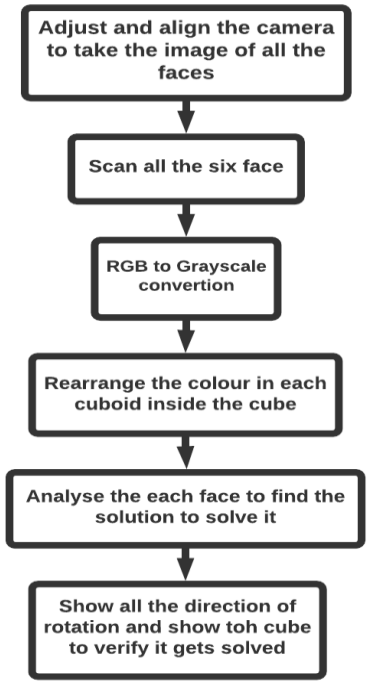


**Fig 1**. IP Process Flow

The purpose of this project is to create an autonomous system to solve the Rubik cube with just with the help of a mobile hence we called our project called "Cube C" were "C" denoted as Cracker. The camera is responsible for capturing the initial configuration of the Rubik's Cube and responsible for processing the video frame information and generating a matrix model for the initial state of the cube. After generating a matrix model for the initial cube, the computer will then apply Kociemba algorithm and Fredrich Method (an optimal algorithm used to solve a Rubik's

 DOI : 10.35291/2454-9150.2022.0085 

Cube) to generate a solution sequence that can be processed sequentially and to show the solver in form of animational movement and in terms of Rubik cube terminologies also.

## V. SYSTEM DESIGN

### A. Capturing the Cube with OpenCV

***1) Camera Orientation:*** Cameras will be used for the user's mobile phone. Each side of the cube will be facing i.e. all the six will be captured like the front, top, and the left face and then back, bottom, and the right faces in this sequence the image will get gathered and it somehow the image color of the physical cube and the one extracted cube done not get match a filter bottom to set the color to the right place will be placed to attain the accuracy

***2) Utilizing OpenCV for Image Processing:*** OpenCV is an open source computer vision and machine learning software library We will be leveraging this library to perform image analysis on a Rubik's Cube, which will allow us to generate a matrix model for any cube orientation. The OpenCV library is used which will enable us to capture and analyze video frames from the captured images

Capturing the faces of a cube requires **three** primary operations:

***Grayscale conversion*** **-** A video frame will be captured and converted to Grayscale using OpenCV's *cvtColor* function Grayscale conversion transforms RGB pixel values into black and white intensity values. Converting to Grayscale allows an easy transformation to a binary (black and white) mage, which is used to filter out features that are not important.

***Canny edge detection*** **-** Canny edge detection is a multi-stage algorithm used to detect a wide range of edges in an image. We will be using OpenCV's *Canny* function to identify the edges of the Rubik's Cube, which, when combined with contour filtering, will allow us to dynamically identify the planes in the frame that represent the three faces of the cube.

***Contour filtering*** **-** Contour filtering will be used to identify the contours of the Rubik's Cube within the image. OpenCV's *findContours* function will allow us to identify the region of space in the image where the Rubik's Cube resides. Likewise, it will enable us to identify individual cubelets on each of the three faces of the cube.

| COLOR | CHARACTER |
|---|---|
| WHITE | 'W' |
| RED | 'R' |
| BLUE | 'B' |
| GREEN | 'G' |
| ORANGE | 'O' |
| YELLOW | 'Y' |

**Table 1.** Cubelets Colour to ASCII Colour Mapping

### B. Kcube and the Solution Sequence

Kcube is a C++ application developed by Greg Schmidt that utilizes Kociemba's two-phase algorithm which uses two stages of an iterative depth first search algorithm. The Kcube application will be used to generate the solution sequence needed to solve the Rubik's Cube that was captured during the image processing phase. The matrix model generated from the image processing phase will allow us to provide Kcube's command-line interface with the cube representation needed to generate a solution sequence. Kcube's command-line interface takes six parameters, one for each face of the cube. The values for these parameters are the colour characters at each of the cubelet locations for that face.

```
             |************|
             |*U1**U2**U3*|
             |************|
             |*U4**U5**U6*|
             |************|
             |*U7**U8**U9*|
             |************|
|************|************|************|************|
|*L1**L2**L3*|*F1**F2**F3*|*R1**R2**F3*|*B1**B2**B3*|
|************|************|************|************|
|*L4**L5**L6*|*F4**F5**F6*|*R4**R5**R6*|*B4**B5**B6*|
|************|************|************|************|
|*L7**L8**L9*|*F7**F8**F9*|*R7**R8**R9*|*B7**B8**B9*|
|************|************|************|************|
             |************|
             |*D1**D2**D3*|
             |************|
             |*D4**D5**D6*|
             |************|
             |*D7**D8**D9*|
             |************|
```

**Fig 2.** Matrix Representation of 3x3x3 Rubik Cube

## VI. METHODOLOGY

The application till now contains four sections:

**Camera Input**: To Solve my capturing the image of the physical cube

**Manual Input**: To add colors manual picking colour and adding to the raw or blank cube

**Cube Game**: 1vs1 game different pattern will be given to each player and they must solve with stopwatch

**Scrambler Generator**: Getting the cube scramble and solution of the same pattern to know the setup and rotation pattern

**SOFTWARE ARCHITECTURE: [A]** After clicking the image, conversion of the captured image to a binary image is completed. We are handling all the six colours separately. then, blob detection is completed followed by determination of the centroid of every blob. **[B] [C]** Sequence of colours there on face of the cube is obtained by sorting the centroids of the blobs. A string is generated by the sequence obtained for the Kociemba algorithm. Image is being processed by the code. We are taking the pictures of all the faces of the cube. The sequence during which the image is taken is decided by the Kociemba algorithm.

 DOI : 10.35291/2454-9150.2022.0085 

### [A] Detecting the Cube

This step is to create a mask for the locations of the faces on the Rubik's cube from an input RGB image. We apply a high pass filter to each of the three colour channels to help enhance the black edges separating the faces on the cube. Then we convert the filtered RGB image into grayscale and apply a threshold to isolate the faces of the cube.

We check the regions that were created from the thresholding, discarding the regions that are not an acceptable size, do not have an acceptable shape (eccentricity), or are not solid. At this point, we should be left with only faces from the cube. However, some of the faces on the edge may have the black border visible and thus were not detected. We will need to recover these faces by estimating their location from the detected faces.

We divide the detected faces into three groups based on orientation. This will help us help us identify which side each of the faces belong to. Finally, for each of the sides, we fill in any potential missing sides to create the mask for the image.

### [B] Identifying the Colours of the faces

Once we have the mask for the image, we transform the image from RBG space into YUV space. The pixel colour components are normalized with respect to the pixel's intensity

### [C] Detecting the State of the Cube

we determine the state of the cube by maximizing the probabilities determined in the previous section. We also take into consideration restrictions on the colours of the pieces of the cube. Examples of the restrictions are that the blue center piece must be on opposite as the green center piece, there is only one blue-yellow side piece, a red-orange side piece does not exist, etc.

Once the Rubik's cube was detected, we had a 94 % accuracy in detecting the pixels. Errors existed in bunches because of our assertion on the number of pieces that that the Rubik's Cube can have for each colour. Therefore, we didn't have any errors in 69% of our test images but in the cubes with errors in the detection, we had accuracy percentages of as low as 52%. In the cases of errors, we usually mixed up similarly colored pieces such as the blue and green pieces or the red and orange pieces.

| PARAMENTERS | DETECTION | ACCURACY |
|---|---|---|
| Sunlight | 50 % | 96% |
| Lamp | 83% | 87% |
| Flash | 100% | 97% |
| Scrambled | 75% | 94% |
| Unscrambled | 100% | 95% |
| Noisy Background | 71% | 88% |

**Table 2.** Detection of Pixels with their Accuracy

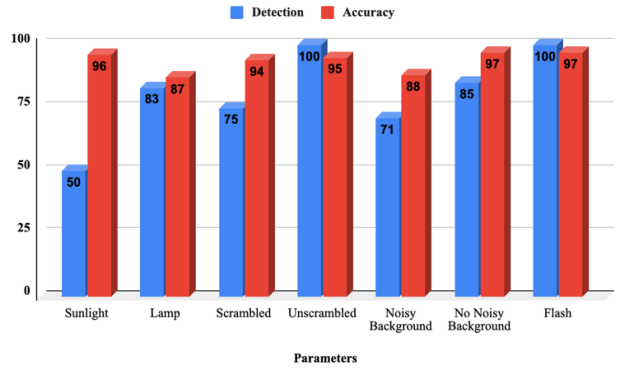


**Fig 3.** Visual Representation of Pixels Detection

## VII. IMPLEMENTATION & RESULTS

The overall level of accuracy in all the tests was high, this could be attributed to the fact that all measurements recorded were made accurately, such as documentation of moves taken, and time taken. Starting the stopwatch and finishing it at the exact right time was planned carefully to ensure a fair test. Once again, the fact that the human interference with the experiment was minimized, greatly improved the accuracy of the test, preventing human errors or inconsistencies by making using an application to solve the cube, and noting down the moves and solving the cube by hand, it can't eliminate any room for human error on inconsistencies.

Kociemba Algorithm, CFOP Method, with our case we have implemented of two Algorithm and get the accuracy of the algorithm with the help of the time taken by each of them.

In our cases both the cubes have given the similar pattern to get solved and each cube have solved using the above algorithms.

| TEST CASES | NO OF MOVES | TIME TAKEN |
|---|---|---|
| #1Test | 29 | 2.35 |
| #2Test | 25 | 2.26 |
| #3Test | 29 | 2.35 |

**Table 3.** Kociemba Algorithm:

| TEST CASES | NO OF MOVES | TIME TAKEN |
|---|---|---|
| #1Test | 56 | 4.36 |
| #2Test | 93 | 5.50 |
| #3Test | 56 | 4.36 |

**Table 4.** CFOP Method:

### Camera Input

- Scan all the six sides of the cube and click on the button called "Solve"
- Algorithm will find minimum step to get the cube solved

 DOI : 10.35291/2454-9150.2022.0085 

- As the camera is moved to the cube colour extraction is taken and capture will take place and to take the next side pick the face

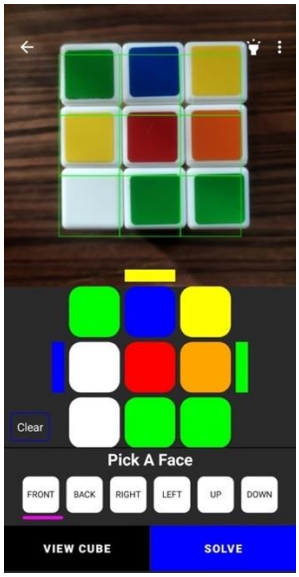


**Fig 4.** Camera Input

**Manual Input**

- There insert all the six colour desired to the combination.
- Enter the Colours in the screen and insert them in the cube.
- All the sides can be used to access to your desired colours
- After inserting all the sides in the cube pattern will help to solve the cube.

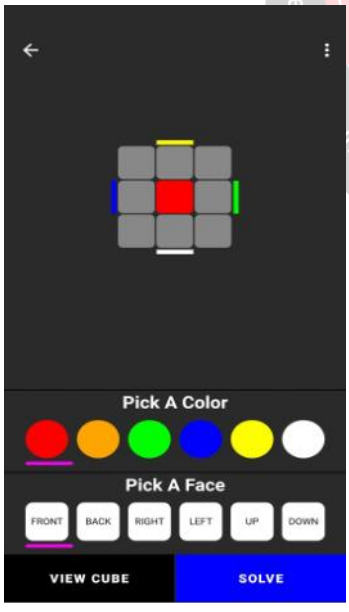


**Fig 5.** Manual Input

**Scrambler Generator**

- There are set of patterns of the cube which will be generated.
- User can set the pattern to its regular cube and try to solve the cube.

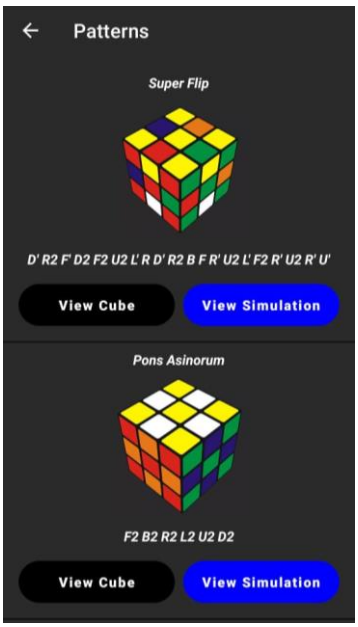


**Fig 6.** Scrambler Generator with patterns list

- "View Cube" shows you the pattern of the specific cube you have seen above.
- "View Simulation" show you the pattern with the exact animation of rotation of the cube.
- While solving the cube you have you have two forward one is the single move and other is full forwards.
- These forwards shows you the moving and highlights you to the extract move you are doing at that time
- And the speed setter to show you the cube rotation speed which you can slow it or fast it as it on you

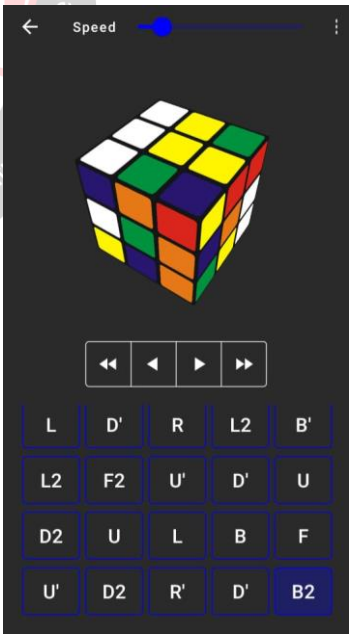


**Fig 7.** Simulation of Cube with pattern list

 DOI : 10.35291/2454-9150.2022.0085 

**Cube Timer (1 Vs 1 Game)**


**Fig 8.** 1v1 Game Page

- One can challenge your friend to play with you Here we have a screen in such a wayside opposite side to each other
- For both of you to start the game both must touch on their side on the same time on start and anyone when finisher first tap on the side of the timer and timer stops.
- Create a session with both of your name and play
- Points get allots the winner side

## VIII. CONCLUSION

The Rubik's cube still proves to be a very difficult challenge for humans to solve, but it is even more challenging to make an application solve it. With the knowledge of image processing, and algorithm development it could be made possible. Implementing them all in one process is a hard yet intuitive task since this project is a rightful experience for implementing the knowledge in respective fields. For accurate and faster solution for both of you to start the game both must touch on their side on the same time on start and anyone who finishes first tap on their side to stop their side timer. As the algorithm is a major part of of solving a raw or scrambled cube. Kociemba & Fredrich Methods help to solve the cube with the help to pattern matching process.

## IX. FUTURE SCOPE

We can enhance your application like we can be handle rather than scanning and physical cube a virtual cube like a 3D Structure like in an augmented space. Also we have used only 3x3x3 cube for furthermore cube sizes can be added. Engagement of more players can be included. Also the application can be set online to add more players. Online tournament can be a part of further enhancement and global contribution and collaboration will help the application the move further and reach maximum users.

## X. REFERENCE

[1] Justin Ng,2013," Automated Rubik's Cube Recognition", Stanford University, Issued in 2013

[2] Harshad Sawhney, Sakshi Sinha, Anurag Lohia, Prashant Jalan, Priyanka Harlalka, 2013, Autonomous Rubik's Cube Solver Using Image Processing, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 02, Issue 10 (October 2013)

[3] N. M. Ali, M. Rashid, N. K. A. M. Rashid, and Y. M. Mustafah, "Performance comparison between RGB and HSV colour segmentations for road signs detection," Applied Mechanics and Materials, vol. 393, pp. 550-555, September 2013.

[4] E. S. Toshniwal and Y. Golhar, "Rubik's Cube Solver: A Review," *2019 9th International* Conference *on Emerging Trends in Engineering and Technology - Signal and Information Processing (ICETET-SIP-19)*, 2019

[5] Zeng, M. Li, J. Wang, Y. Hou, W. Lu and Z. Huang,," Overview of Rubik's Cube and Reflections on Its Application in Mechanism" Chinese Journal of Mechanical Engineering, vol. 31, no. 1, December 2018

[6] T. Rokicki, "Twenty-five moves suffice for Rubik's cube," arXiv preprint arXiv:0803.3435, March 2008.

[7] R. E. Korf, "A Program that Learns to Solve Rubik's Cube," Proceedings of the National Conference on Artificial Intelligence, pp. 164-167, August 1982.

[8] Pawan Pawar," Insights to agile methodology for software development", https://hackernoon.com/a-case-study-type-insight-into-agile-methodologies-for-software-development-cd5932c6

[9] "The Mathematics of the Rubik's Cube", https://web.mit.edu/sp.268/www/rubik.pdf March17,2009

10] Herbert Kociemba. *The Two-Phase Algorithm* [Online]. Available: http://kociemba.org/cube.htm

[11] Autonomous Robotic Rubik's Cube Solver "https://my.ece.utah.edu/~kstevens/3992/reports/rubik-cube-robot.pdf

[12] Guinness World Records. *Fastest robot to solve a Rubik's Cube* [Online]. Available: http://www.guinnessworldrecords.com/ world-records/fastest- robot- to- solve- a- rubiks- cub

[13] OpenCV Developers Team. *About OpenCV* [Online]. Available: http://opencv.org/about.html

[14] OpenCV Developers Team. *Miscellaneous Image Transformations* [Online]. Available: http://docs.opencv.org/modules/imgproc/ doc/miscellaneous transformations.html#cvtcolor

[15] Wikipedia contributors. *Grayscale* [Online]. Available: http://en.wikipedia.org/w/index.php?title=Grayscale&oldid=652694198

[16 ] Wikipedia contributors. *Canny edge detector*

[Online]. Available: http://en.wikipedia.org/w/index.php?title=Canny edge detector&oldid=655662708

[17] OpenCV Developers Team. *Feature Detection* [Online].Available:http://docs.opencv.org/modules/imgproc/doc/featuredetection.html?highlight=canny#canny

## APPENDIX

 DOI : 10.35291/2454-9150.2022.0085 

To solve a cube, it is standard to define the terminology and orientation layout used in Rubik's Cube theory and analysis. This section describes the basic notation that is used throughout this document.

*A. Faces*

A Rubik's Cube is composed of six faces: right (R), left (L), up (U), down (D), front (F), and back (B). The exact colour of each face is relative to the orientation in which you are holding the cube. For example, if you align the blue face towards you then the blue face is defined as the front face. Each face can be rotated in two different directions: *clockwise* or *counter-clockwise*. These rotations are defined as the direction of rotation when looking directly at that face.

*B. Fundamental Moves*

The most fundamental moves are 90-degree clock-wise rotations for each of the faces outlined above. These moves are outlined below:

- **R** - Indicates a 90-degree clockwise rotation of the right face such that the side on top rotates towards the back.
- **L** - Indicates a 90-degree clockwise rotation of the left face such that the side on top rotates towards the front.
- **U** - Indicates a 90-degree clockwise rotation of the upper face such that the side in front moves to the left.
- **D** - Indicates a 90-degree clockwise rotation of the downward face such that the side in front moves to the right.
- **F** - Indicates a 90-degree clockwise rotation of the front face such that the side on top moves to the right.
- **B** - Indicates a 90-degree clockwise rotation of the back face such that the side on top moves to the left.

*C. Modifiers*

For each of the fundamental moves above, there are modifiers that can be appended to the move to change the rotation of the face. My example below uses L as the base move, but these modifiers can be applied to any of the fundamental moves.

- L' - Indicates a 90-degree counter-clockwise rotation of the left face such that the side on top rotates towards the back (opposite direction as that defined above).
- L2 - Indicates a 180-degree rotation of the left face (two rotations).

*D. Cubelets*

A cubelet refers to a particular piece on the cube. Cubelets are categorized based on their position. There are three types of cubelets: center cubelets, edge cubelets, and corner cubelets A center cubelet is unique. All other cubelets revolve around the center cubelets, they never move (go ahead, try, and move the center piece). Edge cubelets connect two face pieces together at an edge. A corner cubelet connects three pieces together at the corner of the cube.

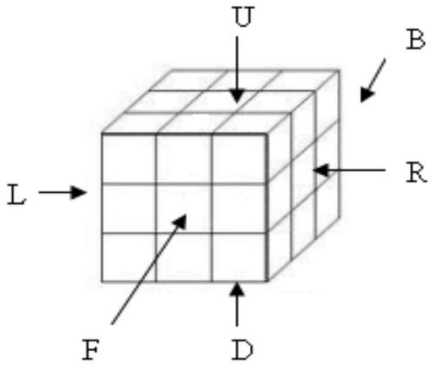


**Fig 9.** Structure side of a 3x3x3 Cube

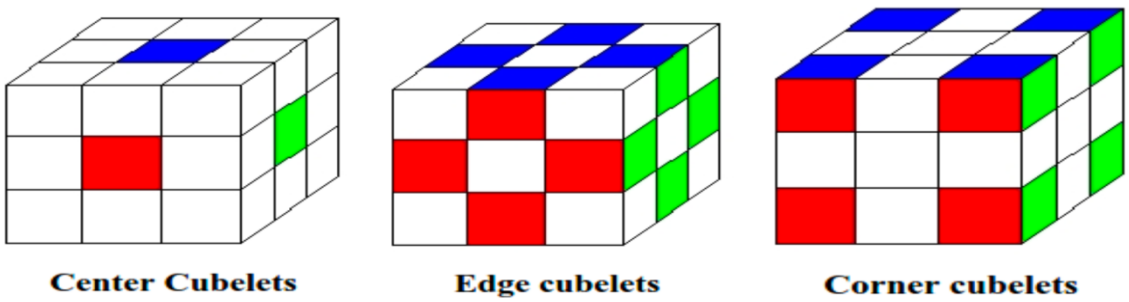


**Fig 10.** Color Representation of Center, Edge, Corner Cubelets

 DOI : 10.35291/2454-9150.2022.0085 