International Journal of Scientific Research and Engineering Development– Volume2 Issue 3, May –June 2019

Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

# Autonomous Rubik's Cube Solver Bot

S. P. Rohith*, A. Mohamed Sharif*,S. Jayasankar*,M. Harikrishnan*

*(Department of Robotics and Automation, PSG College of Technology, Coimbatore, India)

---

## Abstract:

Rubik's cube is considered to be the most challenging puzzle developed for humans. Building a robot to solve such a puzzle is a challenging task. In this paper, the design of such an Autonomous Rubik's cube solver using Arduino Mega 2560 has been discussed. The paper deals with the overall process flow for solving the cube. Color recognition is used for detecting the initial orientation of the cube. Segmentation is used to obtain the HSV mask and thus the color pattern of the scrambled cube. Dot sampling is incorporated to improve the accuracy of the color recognition and segmentation processes. Finally, the categories of solution using Layer by Layer (LBL) algorithm for solving the cube has been shown.

*Keywords* **—Autonomous System, Color Recognition, Dot Sampling, Rubik's cube.**

---

## I. INTRODUCTION

The Rubik's cube has been one of the most interesting and challenging puzzles ever built. It continues to remain in the top among the puzzle games because of the design of the puzzle i.e., one correct solution out of the 43 quintillion other possibilities [1]. This Rubik's cube is a cubic structure with 3x3 cubes inside it. The six faces of the cube have six different colors and thus the entire structure is made up of different colored smaller cubes. The cube has been designed in such a way that the smaller cubes can be moved anywhere. This challenging puzzle can take any time from seconds to even hours to complete.

With the development of mechanical structure, implementation of image processing and development of algorithm, a robot could be made to solve it. This paper deals with all these processes that were involved. Image processing plays the most salient role of all, and thus its accuracy was considered very vital. HSV color range was finalised because of its simple yet accurate range for different colors [2][3]. With the color range defined, each color could be well differentiated and thus identified. Identifying the colors is essentially the first process and generation of algorithm is the next step. A pre-programmed set of possibilities and the corresponding responses are referred and the right algorithm is thus generated. These algorithms thus generate signals to run the corresponding actuators of faces of the cube that needs to be rotated. The right algorithm must be supported by a good mechanical design, and thus a cage-like mechanism was made which enables easy loading and unloading of the cube. The process flow of the Autonomous Rubik's cube solver bot is shown if Fig.1.

## II. LITERATURE SURVEY

Rubik's cube has attracted people from all around the world and also has been used for research purposes by scholars in various disciplines in account of its unique feature [4]. In [5] & [6] the authors have discussed about programs that learn to solve the Rubik's cube in the least possible steps. The paper [7] proposes an algorithm that solves the Rubik's cube in an optimal solution length of 18 moves using pattern

  

databases.In [8] & [9] the authors have discussed various methods for the detection of color for the Rubik's cube.In [10] & [11] the authors discuss in detail about the design of a robot for solving the Rubik's cube using pneumatics and mechatronics systems respectively. In [12] & [13] the authors present a cube solver robot based on STM 32 and a multi-fingered hand respectively.

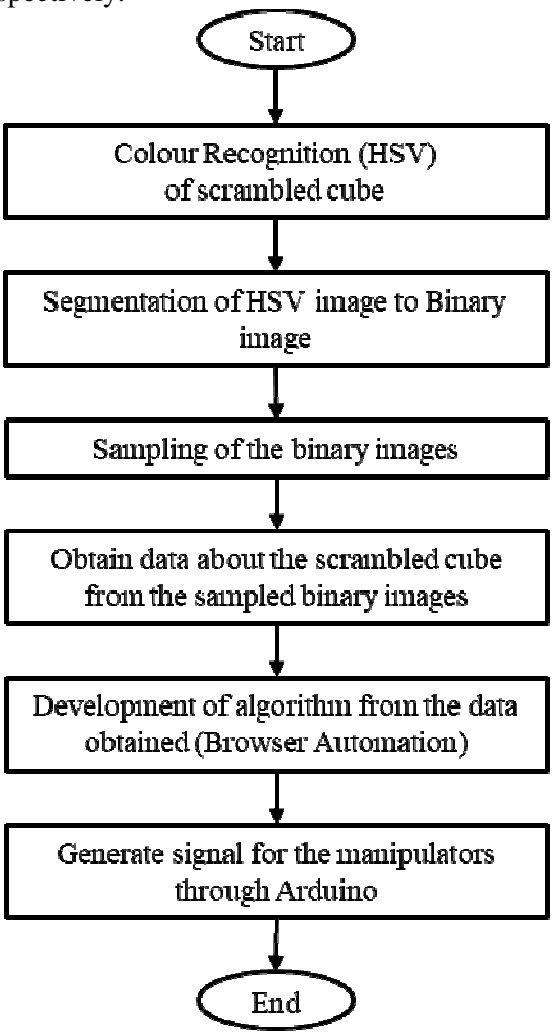


Fig. 1 Process flow of the autonomous system

## III. PROPOSED WORK

### *A. Hardware Implementation*

The block diagram of the system is shown in Fig. 2. The complete system is enclosed with acrylic sheets to form a cubic structure that holds the six servos. The acrylic sheets were machined using cutting wheel and jigsaw. The Rubik's cube is connected to the continuous rotation 360 degree servo motors using aluminium couplers. The couplers were custom designed using abrasive cutting machine and vertical drilling machine.

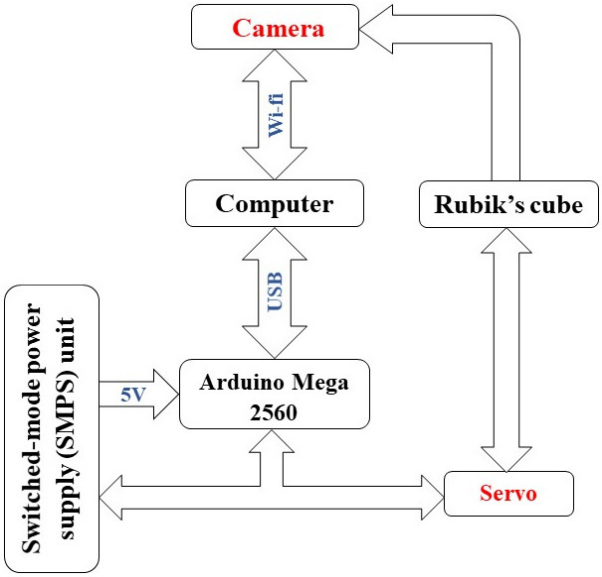


Fig. 2 Block Diagram of the proposed system.

The image of the cube is captured manually on all sides using a camera. The captured images are sent automatically to the computer via wi-fi. The images are then analysed using OpenCV to find the position and color of each piece. The acquired data is then processed to generate the algorithm required to solve the cube which is fed to the servos via an Arduino controller. The servos and Arduino are powered using a switched-mode power supply unit.

### *B. Color Recognition*

The Rubik's cube has six sides namely the Face (F), Back (B), Up (U), Down (D), Right (R)

TABLE 1
SIDE AND COLOR MAPPING

| SIDE | COLOR | (H, S, V) VALUE |
|---|---|---|
| Face | Green | ($120^0$, 75%, 100%) |
| Back | Blue | ($195^0$, 100%, 100%) |
| Up | White | ($0^0$, 0%, 100%) |
| Down | Yellow | ($60^0$,100%, 100%) |
| Right | Red | ($5^0$, 100%, 100%) |
| Left | Orange | ($28^0$, 100%, 100%) |

and Left (L). Each side corresponds to a color that varies based on the viewer's perspective.In this paper for explanation, as listed in Table 1, we have fixed the color for each side i.e. the cube is always fixed in thisorientation. The flat view of the unscrambled cube with the considered orientation is shown in Fig. 3.

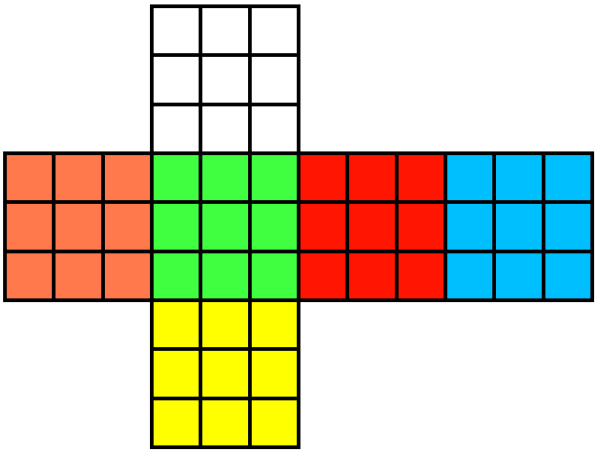

Fig. 3 Flat view of unscrambled cube.

The image of each side of the scrambled cube is captured using the camera and sent via wi-fi to the computer. The images are first resized to a square of fixed length and converted to HSV format. HSV color segmentation offers better accuracy when compared to RGB color mode. Segmentation is performed on each image to differentiate the color on each side of the cube.The obtained images are stored in the workspace for further processing.

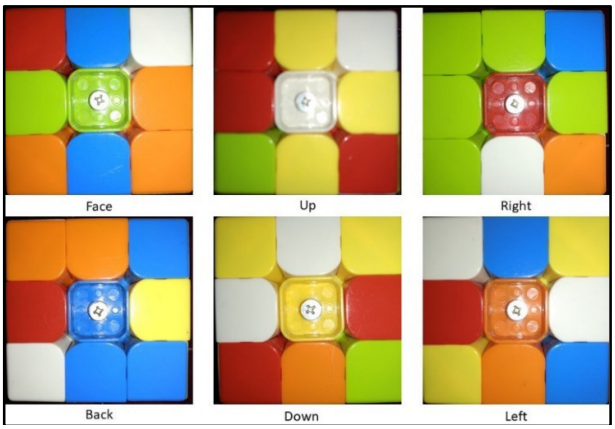


Fig. 4 Example set of input images.

Fig. 4, presents an example set of input images of a random unscrambled cube captured using the camera. The segmented images for each color of the Face side are shown in Fig. 5, similarly the segmented images for the other sides are also obtained and stored in the workspace. Color segmentation is performed by comparing each pixels of the image with an optimally selected lower range and upper range of HSV value and return a binary image where 1 indicates that the pixel value is within the range and 0 indicates that the pixel value is outside the range as shown in (1). Consider $(A, B, C)$ as the HSV value of any random pixel selected, let $(H_{min}, S_{min}, V_{min})$ and $(H_{max}, S_{max}, V_{max})$ represent the lower range and higher range of the considered color.

$$D(x,y) = \begin{cases} 1, if\ (H_{min}, S_{min}, V_{min}) < (A, B, C) < (H_{max}, S_{max}, V_{max}) \\ 0, otherwise \end{cases} \quad (1)$$

where, $D(x,y)$ represent the intensity of the pixel after segmentation is performed i.e., the intensity of the binary image.

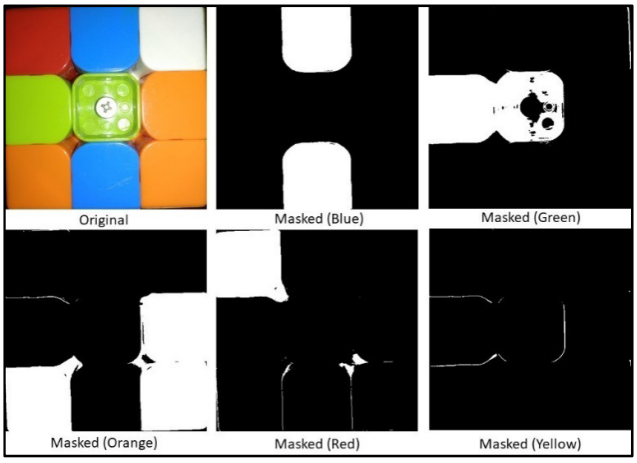


Fig. 5 Original and masked images of the Face side.

### *C. Sampling*

The binary image obtained after color segmentation contains a small percentage of error which here is removed using Dot sampling. First, each masked image is resized into a fixed range, here we have considered a 500x500 image. The resized image is divided into a 3x3 matrix, thus

  

forming 9 squares each of size ($^{500}/_{3}$, $^{500}/_{3}$). The color for the centre is fixed for all input images, thus it is not considered for the sampling process. A good number of sample pixels are selected randomly from each of the remaining 8 squares. The intensity value of each selected pixel is noted. Intensity value of 1 corresponds to the presence of color and 0 corresponds to the absence of color in each selected pixel. The number of pixels with intensity value 1 is compared with an optimal value and thus the decision of whether the color is present or absent in the square is made.

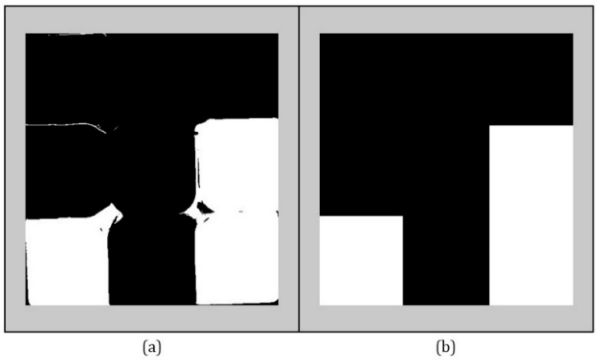


Fig. 6 Binary image of the Face side (orange segmented) (a) before and (b) after sampling

As shown in Fig. 6 this method gives a perfect binary image, making further processing a lot easier. Thus, color matrix is obtained from the binary image which can be further processed for developing the algorithm to solve the Rubik's cube.

The above sampling process is performed for all binary images i.e., for all colors, for all the sides. The data from these sampled binary images is processed to obtain the color for each square in each side, thus an exact representation of the scrambled cube can be obtained. The flat view of the example set of input images obtained from the above processing techniques is shown in Fig. 7. The exact representation is stored as a list collection data type for further processing in Python programming language.

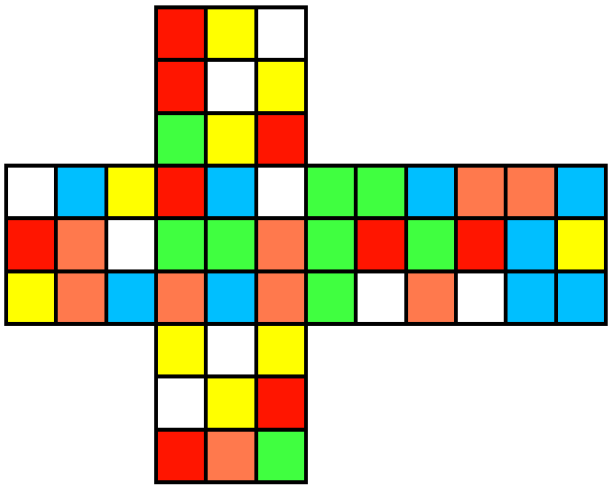

Fig. 7 Flat view of the considered scrambled cube example

### *D. Algorithm*

The data fed to Python as a list data type is further used to solve the Rubik's cube. The LBL (Layer by Layer) solution is used here. The solution has been classified into the following five categories. The projected view of a sample scrambled Rubik's cube subjected the following steps is shown in Fig. 8. The algorithm is converted into Python code and fed to the data fed to create signals for the actuators which controls the cube rotation.

*1)* ***Top edges:*** The first step is to form the cross on the top layer. This step orients and places the 4 edge pieces on the top layer.

*2)* ***Top corners:*** The second step is to orient and place top corners. This solves the first layer (top surface) of the cube.

*3)* ***Middle edges:*** This step involves placing the middle edge pieces at the corresponding position. This solves the first two layers of the cube.

*4)* ***Bottom edges:*** In this step the bottom edge pieces are oriented and placed at the correct position. This solves the bottom cross on the bottom layer.

5) ***Bottom corners:*** The final step is to place the bottom corners at their corresponding positions and orient them. Thus, the Rubik's cube is completely solved.

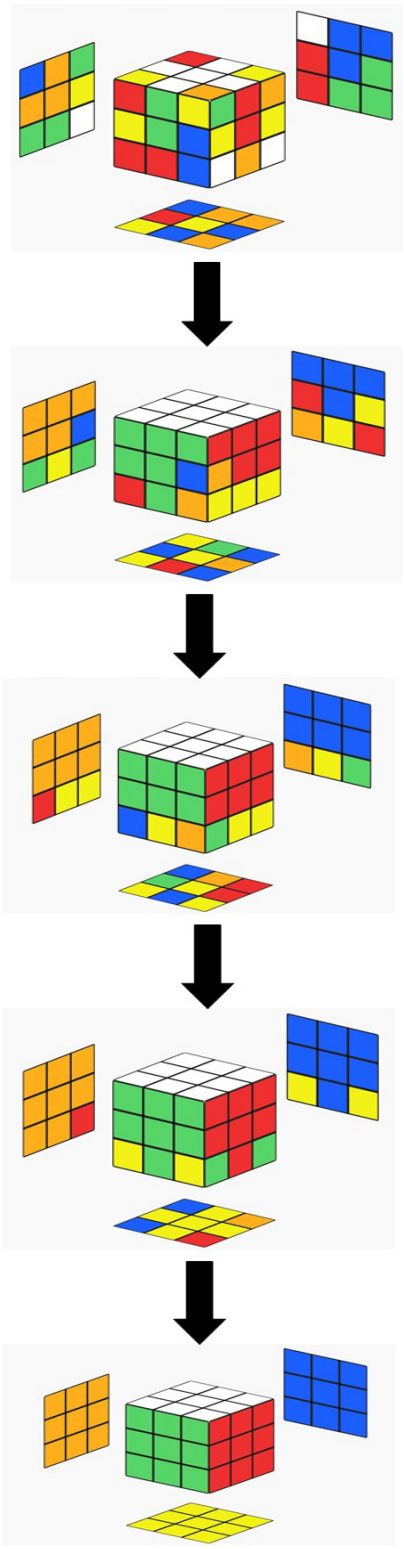

Fig. 7 Projected view of a sample unscramble cube subjected to the Layer by Layer (LBL) solution steps.

## IV. 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 a robot solve it. With the knowledge of image processing, robot kinematics 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, in future, Browser Automation (Robotic Process Automation) using Selenium WebDriver can be implemented, which involves a process of sending the current cube status to the browser and thus receiving better and faster solution algorithms.

## ACKNOWLEDGMENT

We gratefully acknowledge the Department of Robotics and Automation Engineering, PSG College of technology for providing us the required facilities and support.

## REFERENCES

[1] Rubiks.com. (2019). *Rubik's America | Cubes, Puzzles, How-to-Guides & More*. [online] Available at: https://www.rubiks.com/ [Accessed 24 Mar. 2019].

[2] P. Chmelar and A. Benkrid, "Efficiency of HSV over RGB Gaussian Mixture Model for fire detection," *2014 24th International Conference Radioelektronika*, Bratislava, 2014, pp. 1-4.

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

[4] D. 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.

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

[6] 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.

[7] R.E. Korf, "Finding Optimal Solutions to Rubik's Cube Using Pattern Databases", *Proceedings of the American Association of Artificial Intelligence National Conference*, pp. 700-705, July 1997.

[8] S. Liu, et al, "Color Recognition for Rubik's Cube Robot," *arXiv preprint arXiv:1901.03470*, January 2019.

[9] F. Ďurovský, "Robust Rubik's Cube Detection Using Hough Transform and Advanced Clustering Functions," *Applied Mechanics and Materials*, vol. 613, pp. 253-264, Trans Tech Publications, 2014.

[10] J. Varga, F. Ďurovský, and J. Kováč, "Design of pneumatical Rubik's cube solver," *Applied Mechanics and Materials*, vol. 613, pp. 265-272, Trans Tech Publications, 2014.

[11] S.L. Lu, M. Huang, and F.R. Kong, "The Design of a Rubik's Cube Robot," *Advanced Materials Research*, vol. 709, pp. 432-435. Trans Tech Publications, 2013.

[12] G. Zhao, H. Tu, and Y. Liu, "A Design of Magic Cube Robot Based on STM32," *IOP Conference Series: Materials Science and Engineering*, vol. 428, no. 1, p. 012064, IOP Publishing, 2018.

[13] R. Higo, Y. Yamakawa, T. Senoo and M. Ishikawa, "Rubik's Cube Handling Using a High-Speed Multi-Fingered Hand and a High-Speed Vision System," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, 2018, pp. 6609-6614.

  