Rubik’s Cube Recognition and Identification using Computer Vision

Saichand Bandarupalli and Rahul Dev Appapogu

Course No: CSCI507
Colorado School of Mines

Introduction

- Named after Erno Rubik, its inventor (in 1947)
- Has always been an interest for mathematicians and engineers

Quick Facts:

- Official Rubik's cube (3x3) consists of 6 solid colors.
- Each small cube is called a cubie
- Has 8 corner cubies,12 edge cubies and 6 center cubies (immovable)
- Possible number of configurations: **43,252,003,274,489,856,000**
- God's Number: 20

Key Objectives

- Recognition and Identification of Rubik’s cube using Computer Vision techniques/algorithms
- 3D model reconstruction of the Rubik’s cube
- Solving the identified Rubik’s cube using Matlab

Existing Work

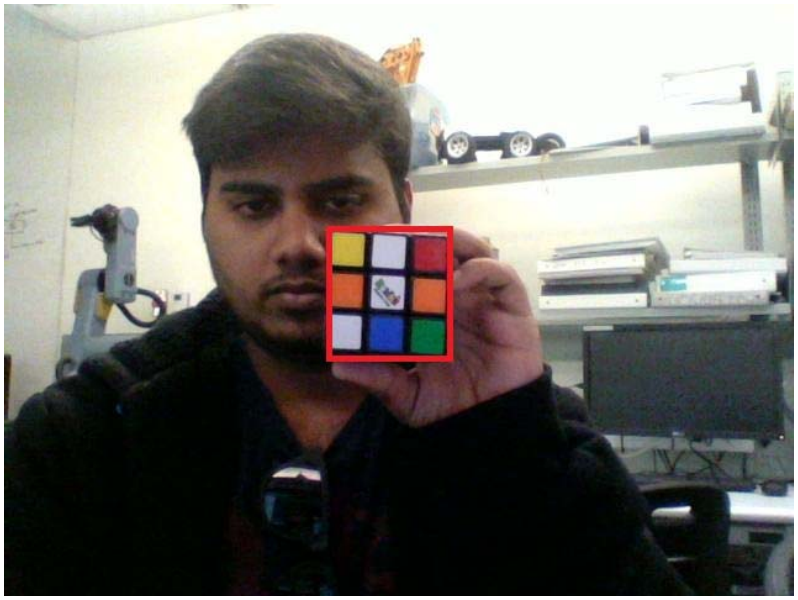
- Rubik's Cube® Solver using an Arduino and MATLAB
- Rubik's Cube Simulator and Solver
- Not open source
- Mostly used feature extraction

Our Approach

- Recognition:
  - Each face has to be identified separately
  - 9 cubie colors in a given face have to be detected
  - Repeat for all 6 faces

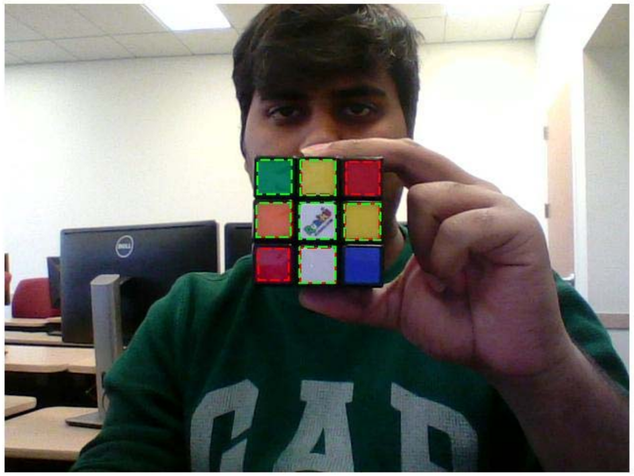
1. Fixed Location in the FOV of a Camera
   - Place one face of the cube in a given location
   - Crop the face of the cube part
   - Divide into 9 regions for identifying color

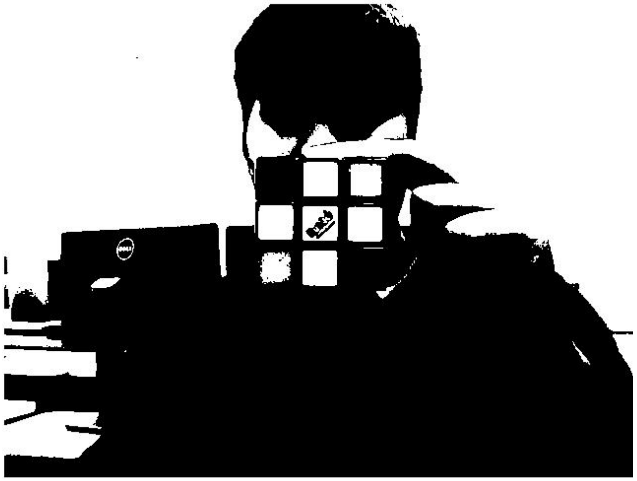
   Non - Robust

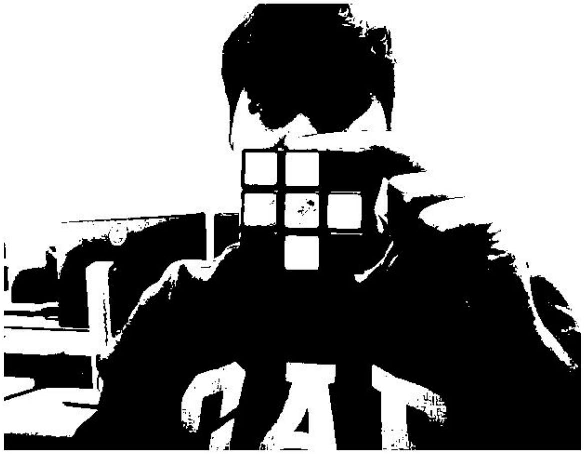


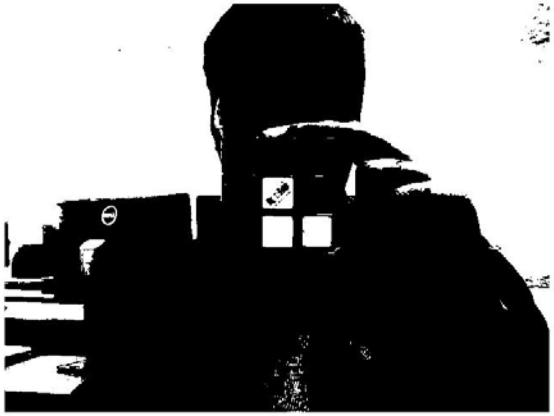
Our Approach

- Recognition of cube in any random position
  - Extract R, G, B images from a given image
  - Transform all 3 images into black and white images adjusting threshold
  - Extract blob features (Area, Centroid, MajorAxis, MinorAxis)
  - Set threshold range for area, and identify blobs in that range (t1<Area<t2)
  - Calculate difference between major and minor axes. (For ideal square, it is 0) If the difference is less than a given threshold, identify it as a square. Identify all squares
  - Extract minimum and maximum centroid coordinate values. Draw a bounding box using the coordinate, thus identifying the face. Crop the face.









Our Approach

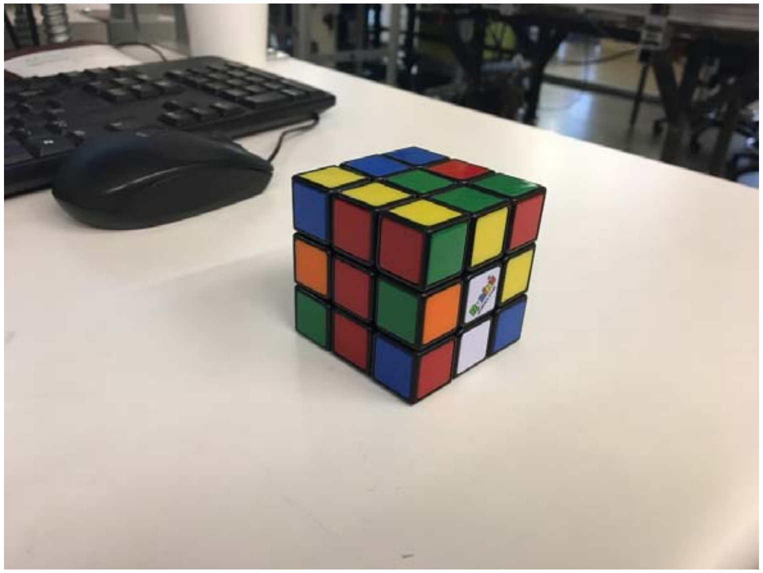
- Color Identification
  - Divide the face of the cube into 9 regions
  - Convert the image from RGB space to HSV space
  - Take the median HSV value of a given region
  - Identify the color using the Hue
  - Hue values account for both dark and bright images

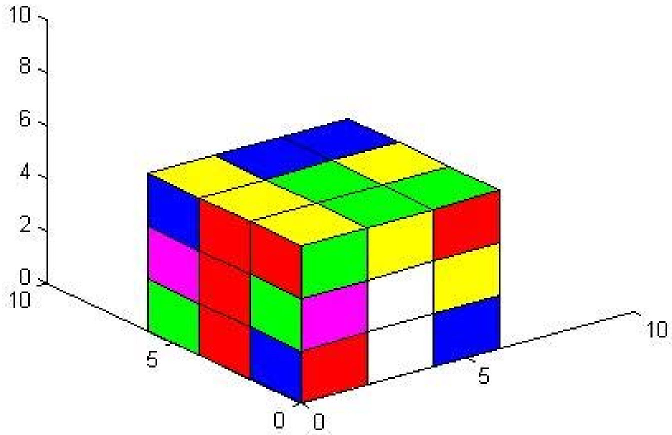
Color	Hue Min	Hue Max
Orange	0.01	0.08
Yellow	0.1	0.28
Green	0.32	0.49
Blue	0.56	0.68
White	0.61	0.73
Red	0.81	1.00

Repeat for all 6 faces and store colors in a 3x3x6 matrix

Use the matrix and fill3 function from Matlab in order to plot the cube

Results Obtained

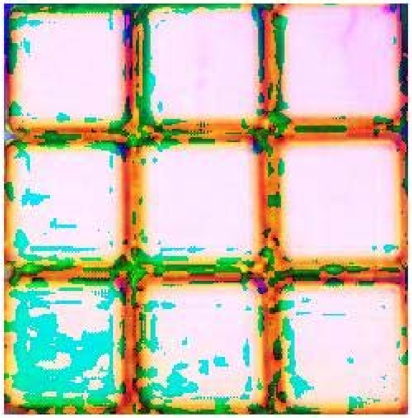




Limitations

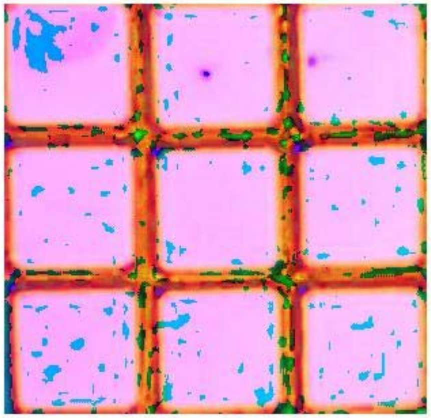
- Reflective Surface – Incident Light is reflected back to the camera. Leads to Bad Color Identification
- External squares in the background – Other Squares may lead to bad detections. Solution: Median Distance Estimate
- Bad Lighting Condition – Solution: May adjust intensity
- Extreme distance between cube and camera may lead to poor identification
- Squares very close to the cube squares

Limitations – Lighting Conditions



Red Face

Bad Lighting



Red Face

Good lighting

Solving the cube

- Does not involve computer vision
- Brute force approach – All possible permutations
- Standard 7-step algorithm
- Thistlethwaite's Algorithm – Almost impossible for a human to execute
- Called, the first genuine math attempt to solve the Rubik's Cube
- Can solve under 45 moves

Thistlethwaite’s Algorithm

- Permits only fixed number of move set / states - (G0,G1,G2,G3,G4)
- A cube in any configuration can be classified in one of the above 5 categories
- Any move sequence can be constructed using G0,if cube is in G0
- Idea is to bring cube to the final state from a given state using the fixed move sets. (G0 to G4,G4 being the final state, which is the solved state)
- Generate Pruning Tables

Thistlethwaite's Algorithm Continued

- Pruning table holds possible Gi -> the current configuration and all possible G(i+1)s, the next possible states.
- In order to generate pruning table, one has to back iterate from the solved state, applying the possible fixed move sets
- Once we have a fully populated Pruning Table, we fixed the orientation of the edge and corner cubies.
- Look through the pruning table and apply possible moves for a given fixed orientation until the cube is solved

References

- https://www.rubiks.com/about/cube-facts/
- https://www.rubiks.com/about/the-history-of-the-rubiks-cube/
- http://www.doc.ic.ac.uk/teaching/distinguished-projects/2015/l.hoang.pdf
- https://www.mathworks.com/matlabcentral/fileexchange/31672-rubik-s-cube-simulator-and-solver