Report on Computer Vision Project
Automatic Rubik's Cube Detection and Recognition

Jeyaprakash Rajagopal*
Matrikel Nr.: 9019231

B-IT Master Studies in Autonomous Systems

Bonn-Rhein-Sieg University of Applied Sciences

Advisors: Prof. Dr. -Ing Rainer Herpers†

Jan. 15, 2015

*jeyaprakash.rajagopal@smail.inf.h-brs.de
†rainer.herpers@h-brs.de

Abstract

This project aims to design and implement an algorithm that will identify the current state of the Rubik's cube by making use of standard techniques in image processing. It attempts to locate three faces of Rubik's cube in a video stream and to extract color of all three faces automatically. The edges of the cube is identified using it's standard shape and then perspective transformation is applied. The color information is extracted by finding mean value of a region of pixels and comparing it against the range of color values defined based on experimentation.

Keywords: Contour detection, feature detection, pixel processing, shape detection, perspective transformation and HSV colorspace.

1 Introduction

The Rubik's Cube is a well-known puzzle that has been invented in 1974 and is considered to be the worlds best-selling toy. For this project, a system will be developed based on computer vision methods. The main focus will be on the visual detection and recognition of the cube's color parts, and it does not solve the puzzle since there are several algorithms available on the Internet for finding the solution based on its face information. In this paper, three faces of the Rubik's cube is to be identified and the transformation of each face is performed using perspective transform. Finally colors are extracted from each face by averaging pixels of a ROI(Region Of Interest) and these mean values are compared against different color ranges in HSV colorspace. A few assumptions have been made about setup. The camera that is used can be in different frame rates, brightness, hue and saturation. So, it is assumed that the cube is situated against a white background which is not positioned directly under the light. This assumption is made, because lighting conditions could make the detection and recognition process harder due to the reflection from the shiny stickers and the possibility of shadows in a higher brightness, edge detection technique misses outer edges of the cube when it is placed against a black background.

1.1 Objective

There is a large community of people in the world trying to find faster and better ways to solve the puzzle. The common need that arises is that people want to get hints on the fastest way to complete the puzzle. However, the only way to do that is to manually enter the each face of the cubes configuration that is the color information and the location of a 3x3 cube into the computer, which is a time consuming task [1]. The motivation of this project is to design an algorithm using computer vision techniques to detect and recognize each face of the three visible sides of the Rubik's cube without manually entering face information into the computer.

1.2 Problem Analysis

To address the problem of detecting Rubik's cube from a 3-D view, the following components will be needed

* Rubik's cube edge detection: Detecting all the three faces of the Rubik's cube using contour shape information for which the pre-knowledge of the cube is used.
* Surface transformation: It is mandatory to transform each face of three visible sides in order to process it for color recognition.
* Color detection: To recognize colors of each cell for the three sides requires pixel processing to find mean of each component(H, S, V) in HSV colorspace.

1.3 Challenges

The challenges of this project are:

* Given three points of a quadrangle finding the fourth point in a 2-D image when analysing the contour information.
* Varying color schemes of the cube in the edges due to the position of the stickers.
* As the colors of the grid is time dependent on the illumination settings, angle at which the camera receives the data.
* Different illumination conditions when recognizing color information.

1.4 Settings

1.4.1 Software requirements

* C++ language.
* OpenCV Library.

1.4.2 Hardware requirements

* RGB web camera.
* Tripod.

1.5 Architecture diagram

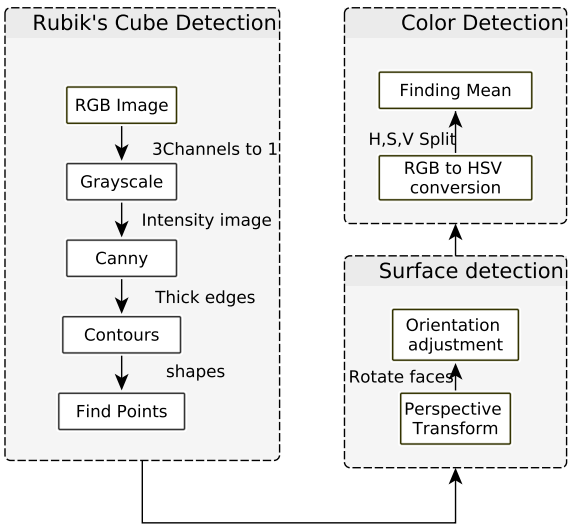

Figure 1: Architecture Diagram

2 Related work

This project's goal is to detect the state of the cube reliably and robustly. There are many techniques specialize in this kind of tasks. Though there is no specific solution for the robust detection, feature extraction techniques has grown significantly over the years. So, the standard image processing techniques are used to detect left, right and top faces of the cube which is oriented in different directions. Template matching techniques lead to large computations since it takes a lot of processing to train the classifier to recognize the faces with different orientation [1]. Shape based techniques suffer in detecting the 3x3 grid of squares since it looses pixel information during transformation.

Andrej[1] combine the hough transformation with HSV colorspace to detect the state of the Rubik's cube. This work is not a published work but it is close to provide good results as Rubik's cube state detector. Though it provides significant results, it can process only one face at a time. Rubik's cube is showed in front of the web camera where it gets sufficient light from the laptop monitor. Rubik's cube reconstruction[2] is accomplished from a single view and the topological map is created to locate the 3x3 grid of the cube.

3 Scenarios

3.1 Rubik's cube detection and recognition against black background

* Detection process against black background eliminates reflection due to shiny stickers but it needs higher brightness in order to locate edges of the cube.

3.2 Rubik's cube detection and recognition against white background

* It is considered that Rubik's cube is tested against a white background. There is a possibility of shadows in higher brightness and recognition of colors is performed by keeping the cube away from the light instead of positioning it under the light.

3.2.1 Rubik's cube detection and recognition with flash light

* Detection process is carried with flash light without any ambient lights. In this scenario, algorithm might produce wrong results during color recognition due to insufficient light on the faces.

3.2.2 Task description

Rubik's cube detection and recognition is intended to extract the state of the cube by recognizing its color information in real-time without manually entering the state. This work can be used to

* Acquire the state in ease.
* No need for a manual entry.
* Assistance in detecting states to solve the puzzle.
* Save time in reading the state of the cube.

4 Approach

To identify the current state of Rubik's cube involves three steps as follows.

1. The three faces of the Rubik's cube to be detected.
2. Process each face and perspectively transform it to the view which is 90% perpendicular to the camera view.
3. Color of each cell of the cube is extracted by calculating mean value of a particular ROI.

4.1 Rubik's cube detection

This module is to identify the possible edges of the Rubik's cube in a 3-D view from the input. The process for each image is composed of a series image manipulations (such as Canny, dilate and finding contours).

* It starts with converting the RGB image into grayscale. There are so many edge detection techniques available such as (Canny, Prewitt, Sobel [7] and etc.,), but Canny edge detection is chosen due to the robustness in noisy conditions.

* Dilate receives the input image as edges and performs a dilation process using a particular structuring element. Dilation results in returning a frame with thicker edges and it is used in finding the contours in the image.

* Finding contours extracts possible shape information from the dilated image and to extract the six vertices of the cube in a 3-D view. All the above mentioned process is depicted in figure 2.

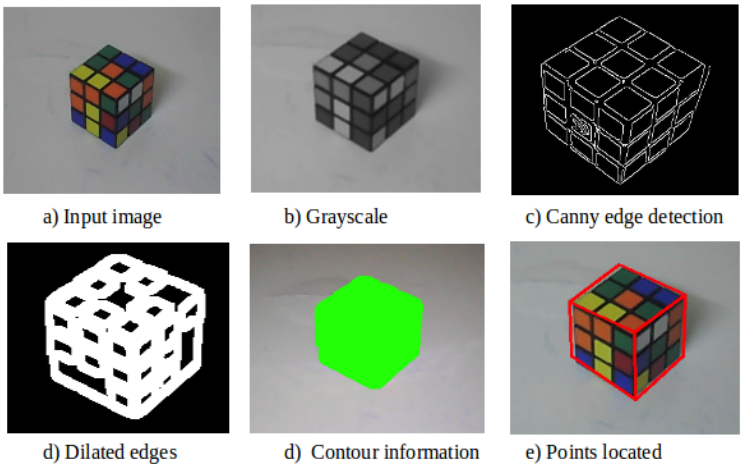

Figure 2: Processing a)input b)intensity c)edge detection d)dilation e)contour shape f)finding points

The challenge rises in identifying the center point(P4) of the cube as depicted in the figure 3. The straight line equation is taken into consideration for this purpose

$$y = (m \times x) + c \tag{1}$$

The following equation calculate the slopes of two lines between three points to find out the fourth missing point, since contour provides the output as depicted [3,4] in figure 2(b).
The slope can be calculated to know how inclined the line is by using formula (2)

$$M = \frac{(P1.y - P2.y)}{(P1.x - P2.x)} \tag{2}$$

After knowing three (x,y) coordinates of a parallelogram, the fourth point is an intersection of two projections of the two lines with respect to opposite directions.

$$P4.x = \frac{(P2.y - M2 \times P2.y) - (P3.y - M1 \times P3.y)}{M1 - M2} \tag{3}$$

$$P4.y = P2.y - (M2 \times P2.x) + (M2 \times P4.x) \tag{4}$$

where P1, P2, P3, P4 are the points Point1, Point2, Point3, Point4 respectively. M1, M2 represents slopes of lines .

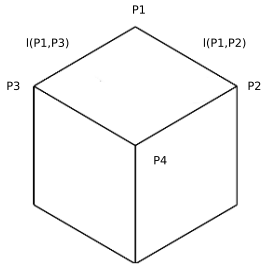

Figure 3: Finding the last point of the quadrangle

4.2 Surface transformation

Rubik's cube location is calculated based on above approximations, to be used in this module to transform the faces of the cube. Perspective transformation [5] is applied for the four points that belongs to a particular face (left or right or top face). Perspective projection is to create 2D images of 3D scenes and this projection is considered to be a realistic approach. The straight lines remain the same after the transformation and as a result projection points are rotated.

Rotation of projected points is achieved by

$$R = \begin{pmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{pmatrix} [5] \tag{5}$$

4.3 Color cell detection and recognition

The input RGB color space is converted to HSV color space to compromise different lighting conditions[1]. A 3x3 grid is created in the reference frame to detect the Region Of Interest(ROI). This grid is constant in the frame and it finds the color information of a particular cell by calculating the average of hue, saturation and value. The ranges of color is experimented and identified in the table below

Colors/Ranges	Hue	Saturation	Value
Red	160-179	200-255	50-255
Orange	0-17	179-255	179-255
Green	50-75	200-255	50-255
Yellow	20-30	100-255	100-255
White	0-180	0-170	100-185
Blue	110-120	150-255	50-255

The saturation and value differs based on the lighting conditions set. The mean value of the hue component is calculated as in formula (6).

$$HueMean = \frac{\sum_{P_i \in (no.of pixels)} Hue(P_i)}{no.of pixels} \tag{6}$$

The mean value of the saturation component is calculated as in formula (7).

$$SatMean = \frac{\sum_{P_i \in (no.of pixels)} Saturation(P_i)}{no.of pixels} \tag{7}$$

The mean value of the value component is calculated as in formula (8).

$$ValMean = \frac{\sum_{P_i \in (no.of pixels)} Value(P_i)}{no.of pixels} \tag{8}$$

5 Experimentation

To test our algorithm, it is assumed that three faces of the cube is visible to the camera. The test is carried out in different set of conditions such as an object placed against a black and white backgrounds. Also tests were conducted to make sure that this algorithm works fine when there is an occlusion in the foreground of the series of input images without hiding the cube's edges. In our experiments, it was possible to extract the edges of the cube in different illumination conditions. This algorithm suffered in detecting the target object with noisy backgrounds and due to the errors in approximating vertices of the cube.

After localizing the cube, the detection accuracy significantly improves in reading the pixel information from each face of the cube. The colour detection suffers in classifying the similarity coloured pieces such as red or orange and blue or green colors and it is also affected due to the loss of pixel information when applying perspective transformation on faces of the cube.

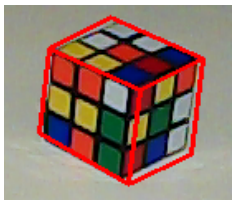

Figure 4: Edges of the cube located

5.1 Rubik' cube detection and recognition against black background

Rubik's cube is placed against a black background, the edge detection process fails in detecting the edges of a Rubik's cube present in the scene. In figure 5 (b,c), the cube is located against black background. Edge detection technique that is used in this work, completely misses the cube's outer layer against a black background. To solve this problem, perfect lighting conditions has to be set for every face that is detected.

Figure 5: Edges of the cube located against black(a,b,c), white background(d,e)

5.2 Rubik' cube detection and recognition against white background

The cube is placed against a white background with brighter lighting conditions lead to shadow and reflection problems due to non-standard shiny stickers. Left, top and right face of the cube is tested for extracting color cells of the grid against a white background as mentioned in the

figure 5 (d,e). Edges of three faces have been detected as in figure 6(a). Figure 6(b,c,d) shows that color cells of top, left and right side faces respectively.

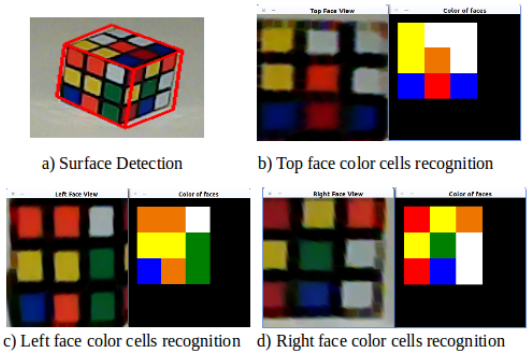

Figure 6: Recognizing color cells of top face against white background

Though the faces of the cube has been extracted successfully, the color cells of the cube can't be successfully extracted in varying lighting conditions. The following experiment is conducted under extreme lighting conditions where the algorithm fails to provide results due to the difference in brightness in the faces of the cube as depicted in figure 7. This could be improved by using color based training to detect color cells of the Rubik's cube.

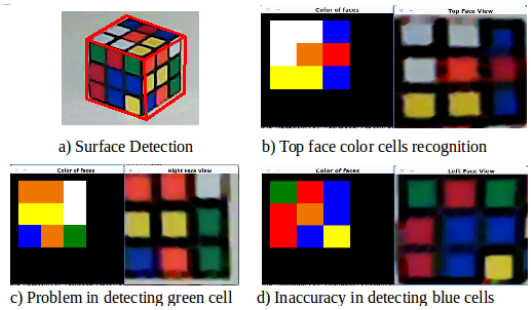

Figure 7: Problem in recognizing color cells in different lighting conditions

5.2.1 Rubik' cube detection and recognition with flash light

Experiments conducted with the flash light of the camera. Rubik's cube edge detection has responded really well with 82% accuracy out of 15 tests conducted and surface orientation, color recognition modules responded with 75% accuracy in recognition. Eventually, this algorithm lacks in accuracy when recognizing colors from each face in extreme lighting conditions and it tends to miss few grids during recognition. Blue, red and green grid cells are not getting detected consistently due to insufficient intensity on its face.

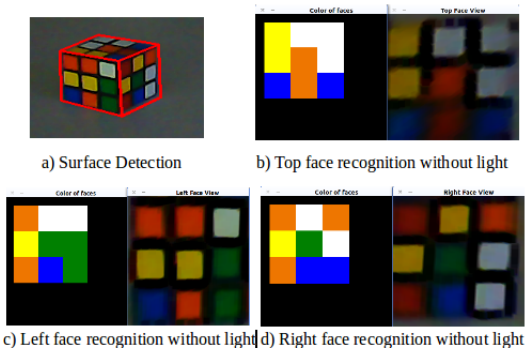

Figure 8: Recognizing color cells with flash light

The following table explains evaluation results of Rubik's cube detection and recognition in varying lighting conditions such as shadows present or not, recognition based only on flash light. It provides better results when the object is placed under perfect lighting conditions without any shadow information present in the detection range.

Conditions/Accuracy	Detection accuracy	Recognition accuracy
Intensity without shadows	88%	85%
Intensity with shadows	82%	75%
Flash light	75%	70%

6 Conclusion and Future work

The results of the algorithm are very promising. It was possible to detect the Rubik's cube fast in real-time and it was possible to attain maximum accuracy in identifying the colors of three faces of the Rubik's cube. There are some deviation in recognizing colour cells due to the fact that the vertices of the cube are extracted based on approximation in real-time. Though this work provides prominent results, there is always a room for improvement. This algorithm can be improved from normal edge detection to multi-level edge detection and it can be extended by removing shadows when the object is located against a white background. Colour extraction can be done efficiently by using color-based particle filter or histogram based recognition[6]. It is possible that this work could be used along with puzzle solving algorithms to acquire its current state without manually feeding cube's current state.

7 References

[1] Andrej Karpathy, "Extracting sticker colors on Rubiks cube, Semester project", github.com/kopernicky/rubik/tree/master/materials

[2] Kasprzak W, lodzimierz and Szynkiewicz, Wojciech and Czajka, Lukasz, "Rubiks Cube Reconstruction from Single View for Service Robots", Machine Graphics and Vision International Journal, Feb 2006.

[3] O Faugeraus, "Three dimensional computer vision, a geometric viewpoint", The MIT press, Nov 1993.

[4] L. Venturino et al. "Improving 3D scene reconstruction through the application of geometric constraints". Workshop on Systems, Signals and Image Processing , Poznan University of Technology Press, Sept 2004

[5] Richard Szeliski, "Computer Vision: Algorithms and Applications", Springer Sept 2010.

[6] Peng Chang, Krumm J, "Object recognition with color concurrence histograms", Computer Vision and Pattern Recognition, IEEE Computer Society Conference on June 1999.

[7] R Maini, H Agarwal, "Study and comparison of various image edge detection techniques", International Journal of Image processing, 2009.

[8] Katja Nummiaro, Esther Koller-Meier, Luc Van Goo, "An adaptive color-based particle filter", Image and Vision Computing, Jan 2003.