2022 14th International Conference on Signal Processing Systems (ICSPS) | 979-8-3503-3631-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICSPS58776.2022.00141

Research on State Recognition of Rubik's Cube Based on Monocular Vision

Mihang Wang
School of Media and Design
Hangzhou Dianzi University
Hangzhou, China
201330018@hdu.edu.cn

Yigang Wang
School of Media and Design
Hangzhou Dianzi University
Hangzhou, China
yigang.wang@hdu.edu.cn

Qing Shi
School of Media and Design
Hangzhou Dianzi University
Hangzhou, China
3144662527@qq.com

Shi Li*
School of Media and Design
Hangzhou Dianzi University
Hangzhou, China
brightlishi@hdu.edu.cn

***Abstract*—Object state acquisition is one of the most critical parts of the industrial automated system. To ensure the acquisition accuracy of the object states, the system usually adopts the method of adding enough sensors, which makes the system more complicated. Taking the state recognition of Rubik's Cube as an example, Huawei Smart Cube utilizes six sensors to fetch state information. In this paper, we propose a different method that only uses a single sensor to get the object states. The monocular camera is used to collect the color information and geometric outline information of the target Rubik's Cube. The KNN algorithm is applied to maximally distinguish various colors, especially yellow, orange, and red, to ensure acceptable color recognition accuracy. The geometric relationship between the position coordinates of adjacent blocks is utilized to confirm whether the current Rubik's Cube is in a self-rotation state or a recognition state. That helps to solve the key state recognition problem. In the experiment, an AR tag is utilized to indicate the user to operate the Rubik's Cube based on the Kociemba algorithm. It has been proved by experiments that our method can reach similar results to the multi-sensors obtained. It shows that the deep utilization of the single sensor information can also obtain the same results generated by multi-sensor information joint processing.**

Keywords—states acquisition, Rubik's cube, color recognition, contour features

I. INTRODUCTION

With the development of science and technology, more semi-automatic or fully automated technologies enter people's lives. For example, BMW's fully automatic production line does not require manual operation, and the whole process is controlled by the machine [1]. Driving assistance system can intelligently recognize traffic lights [2] and road conditions [3]. In these semi-automatic or fully automated technologies, the acquisition of object states is very important. Once an error occurs during the state acquisition, the subsequent automatic operation will be stopped, even leading to large property losses.

As the most important part of the system, much research has been done on the acquisition of object states. SCADA systems use the data of sensors as a way to obtain the states of objects [4]. To ensure the accuracy of the object states obtained, the system usually adopts the method of adding enough sensors. Although the states obtained in this way has high accuracy, it will increase the complexity of the system. In a complex system, once one sensor occurs errors, the entire system maybe has a huge problem. The states of the Rubik's Cube have a total of 430 billion changes [5], and it is a very difficult task to accurately identify one of the states from so many states.

Huawei's smart Rubik's Cube obtains the states through 6 sensors built-in in the Rubik's Cube [6]. For this kind of smart cube, once one of the sensors has a problem, the whole system cannot work. It is the instability of the high-complexity system. To make the system more stable, we attempt to use a single camera to reach fully automated recognition similar to the 6 sensors have achieved. The follow-up experiments prove that our method can reach fully automatic recognition.

II. PROBLEM ANALYSIS

Huawei Smart Cube has a single separate sensor on each face. And the state information of the cube can be obtained in real-time through each sensor. Through the sensor information of each face, the system can know whether the user rotates the Rubik's Cube in place. For example, if the user rotates a certain face of the Rubik's Cube by 45 degrees, the built-in sensor in the smart Rubik's Cube can obtain the state information of the rotating face. And then the system can synchronize the states to the virtual Rubik's Cube in real-time. Our goal is to only use a single camera to reach the acquisition results as similar as 6 sensors have achieved. It means that our proposed method can achieve automate states' acquisition of the Rubik's Cube in real-time.

Although we only use a single sensor, the camera can obtain the color information and contour features in each frame. Through color information and initialization information, we can calculate which face has been rotated. And through contour features, we can judge whether the cube rotated in place or not. Under these operations, we synchronize the real Rubik's Cube state to the virtual cube in real-time.

At the same time, to better demonstrate the fully automatic recognition of Rubik's Cube based on monocular vision, we design a real-time guidance system. It can advise the user on how to rotate the Rubik's Cube through the recognized states. We want to use real-time performance to prove the effectiveness and stability of our method.

The whole system flow chart is shown in Figure 1.

979-8-3503-3631-3/22/$31.00 ©2022 IEEE
DOI 10.1109/ICSPS58776.2022.00141

Authorized licensed use limited to: Universitatea Tehnica din Cluj-Napoca. Downloaded on March 13,2024 at 15:15:51 UTC from IEEE Xplore. Restrictions apply.

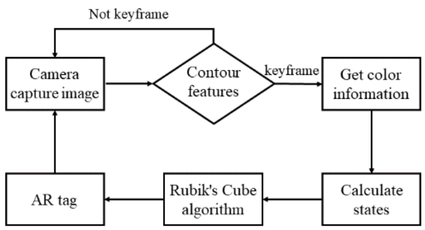


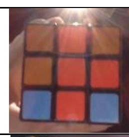
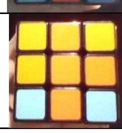
Fig. 1. Flow chart of the whole system.

III. Detail of System

A. Color Recognition of Rubik's Cube under Different Illumination

The color information of the Rubik's Cube is the most important. Color recognition is required by system initialization and state acquisition. If the color of each block cannot be correctly identified, then the state of the cube cannot be accurately obtained. Therefore, the accurate recognition of the color is the key operation of the state acquisition of the Rubik's Cube. The color recognition of the Rubik’s Cube will encounter variable illumination conditions in practical applications, as shown in Table 1.

TABLE I. Rubik's Cube in Variable Illumination

Backlighting	
Side lighting	
Frontal lighting	

To solve this problem, we first need to consider which color space should be used. We use the HSV color space to obtain the color of the block. Because the HSV color space is more suitable for the human eye. It is easier to achieve the display effect expected by the human eye [7].

The Hue values of yellow, orange, and red in the HSV color space are close. The schematic diagram of the three colors of yellow, orange, and red in the HSV color space is shown in Figure 2.

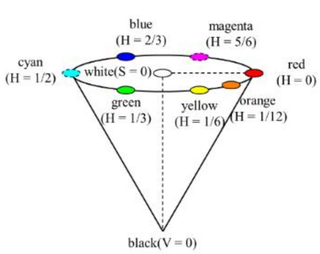


Fig. 2. The position of each color in the HSV color space.

Under bad light conditions, the colors yellow, orange, and red in Rubik’s Cube will be difficult to be distinguished. One example is shown in Figure 3.

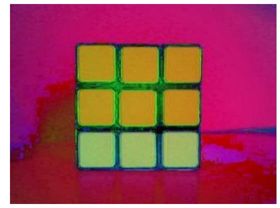


Fig. 3. Visualize the image under HSV color space.

Figure 3 is a visualized result in HSV color space converted from RGB color space. We can see that yellow and orange in Figure 3 are very close and easy to confuse. When the colors are too close to distinguish each other, it will be more unstable if just use a threshold to classify colors in the system. To solve this problem, the KNN algorithm is used to classify the three colors yellow, orange, and red. KNN belongs to instance-based learning. Compared with model-based learning, instance-based learning does not require training and parameter adjustment and directly classifies the data to be tested. It is suitable as a color classification algorithm in this system. The core of the KNN algorithm is to select the K nearest neighbors, and the K nearest neighbors determine which category the current point belongs to [8]. In color classification, the distance formula calculated for each pair of points is:

$$d(a,b) = \sqrt{(a_H - b_H)^2 + (a_S - b_S)^2 + (a_V - b_V)^2} \quad (1)$$

$d(a,b)$ in formula (1) represent the distance between the two points a and b. a_H, a_S, a_V represent the HSV value where the point a is located, b_H, b_S, b_V represent the HSV value where point b is located.

After the system classifies the color by KNN clustering, the visualized result is shown in Figure 4.

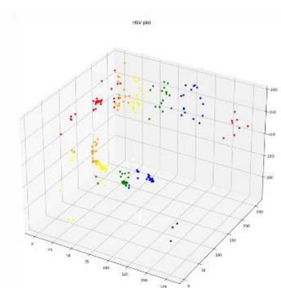


Fig. 4. Visualization of KNN after color clustering.

After the colors are divided into 6 categories, the system will attach a color to each category according to the average HSV value of each one.

B. Rubik's Cube State Recognition in Timeline

As an automatic recognition technology of Rubik’s Cube based on monocular vision, the second problem that needs to be solved is state recognition in the timeline. It is difficult to recognize the Rubik’s Cube state in the timeline. An interval in

Authorized licensed use limited to: Universitatea Tehnica din Cluj-Napoca. Downloaded on March 13,2024 at 15:15:51 UTC from IEEE Xplore. Restrictions apply.

the timeline contains many frames. If the system recognizes the state for each frame, it will become too bloated and cannot be realized in real-time. We define the frame whose cube is rotated in place as a keyframe. Therefore, it is necessary to judge whether the frame should be treated as a keyframe for state recognition. The timeline interval between keyframes is shown in Figure 5.

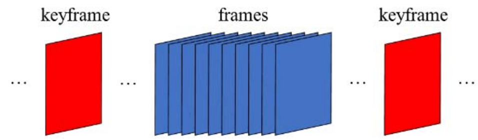
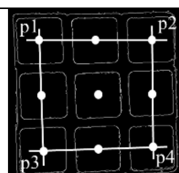
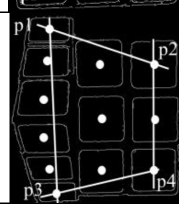


Fig. 5. The timeline interval between keyframes.

The geometric features of the Rubik's Cube are used to judge whether the current frame is the keyframe used for state recognition in our method. When the contours are obtained, the barycenter coordinate of each contour can be calculated. According to the geometric features, the four barycenters are chosen as P1, P2, P3, and P4. P1, P2, P3, and P4 are the points closest to the upper left corner, upper right corner, lower left corner, and lower right corner of the picture respectively. The result linking the adjacent picked points is shown in Table 2.

TABLE II. POSSIBLE STATES IN TIMELINE

Keyframe	
Frame	

From Table 2, we can see that when the Rubik's Cube is in the rotating state, the number of parallel lines is no more than two. That is the recognized feature for distinguishing the rotating state. The parallel lines judging method is as follows. Taking the line p1p2 and the line p3p4 as an example, there is an expression.

$$(y_2 - y_1)(x_4 - x_3) = (x_2 - x_1)(y_4 - y_3) \quad (2)$$

Where x and y are the horizontal and vertical coordinates of the corresponding points. If the left and right sides of the expression (2) are equal, the relation of the line p1p2 and the line p3p4 is parallel.

C. Details of Other Parts of the System

The Rubik's Cube algorithm in this system adopts the Kociemba algorithm. There are two reasons for using the Kociemba algorithm. First, it is more in line with the logical thinking of the computer and has many use cases and research results, which can reduce the program's running time. Second, the steps returned by the algorithm generally do not exceed 20 steps, it is suitable for verification of our method [9].

For the way of indicating to the user how to operate, we use an AR tag to guide the user to operate the Rubik's Cube. Compared with ordinary text guidance, the AR tag is more intuitive. It allows users to quickly understand the next steps that need to be performed [10]. The drawing of the AR tag is shown in Figure 6.

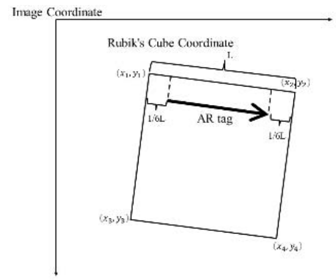


Fig. 6. The relationship between the coordinate system and AR tag.

In Figure 6, the start point coordinates of the AR tag is (x_{start}, y_{start}). Under the Rubik's Cube coordinate system, let its coordinate as (1/6L, 1/6L). Likewise, the endpoint (x_{end}, y_{end}) in the Rubik's Cube coordinate system is (5/6L, 1/6L). According to the relationship in Figure 6, the coordinates in Cube coordinate system can be converted to the coordinates in the image coordinate system. The calculation formulas for the start point and endpoint are as follows.

$$\begin{bmatrix} x_{start} \\ y_{start} \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \begin{bmatrix} 1/6 \\ 1/6 \end{bmatrix} \begin{bmatrix} x_2 - x_1 \\ x_3 - x_1 \end{bmatrix} + \begin{bmatrix} 1/6 \\ 1/6 \end{bmatrix} \begin{bmatrix} y_2 - y_1 \\ y_3 - y_1 \end{bmatrix} (3)$$

$$\begin{bmatrix} x_{end} \\ y_{end} \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \begin{bmatrix} 5/6 \\ 1/6 \end{bmatrix} \begin{bmatrix} x_2 - x_1 \\ x_3 - x_1 \end{bmatrix} + \begin{bmatrix} 5/6 \\ 1/6 \end{bmatrix} \begin{bmatrix} y_2 - y_1 \\ y_3 - y_1 \end{bmatrix} (4)$$

After obtaining the coordinates of the starting point and the endpoint of the AR tag, it can be drawn on the video image to indicate to the user how to operate the Rubik's Cube. To provide users with more intuitive Rubik's Cube guidance, the system also designs a virtual Rubik's Cube to display to users. The virtual cube is filled with 6 faces during initialization, and it will display the corresponding operation animation. It allows users to better understand the current operation steps. The drawing result of the virtual Rubik's Cube can be seen in the experiment results.

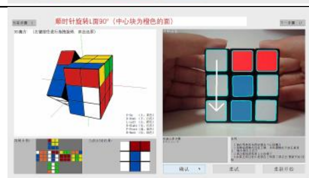
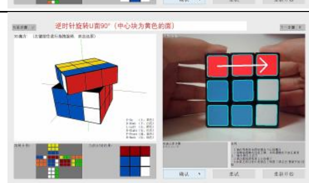
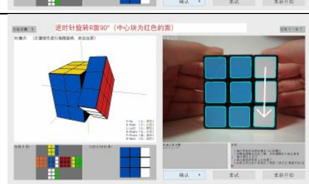
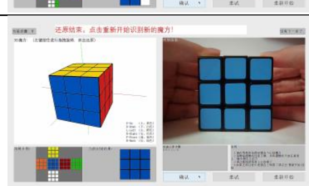
IV. EXPERIMENTS AND RESULTS

To prove the effectiveness of the state recognition of Rubik's Cube based on monocular vision, an AR tag is utilized to indicate the user to operate the Rubik's Cube in the experiment. At first, the Rubik's Cube is randomly scrambled in advance. Then the user without Rubik's Cube experience operates the Rubik's Cube step by step according to the indication of the system.

In the experiment, the window content of the system during initializing, operating, and finishing are shown in Table 3.

TABLE III. SCREENSHOTS OF VARIOUS STATES OF SYSTEM OPERATION

Authorized licensed use limited to: Universitatea Tehnica din Cluj-Napoca. Downloaded on March 13,2024 at 15:15:51 UTC from IEEE Xplore. Restrictions apply.

Initializing	
Step 1	
Step 2	
Step 3	
Step 4	
Step 5	
Step 6	
Finish	

In 20 repeated experiments, users can successfully restore the Rubik's Cube under the indication of our system. The experimental results show that our method for object state acquisition works well.

V. Conclusion

We use a single camera to obtain color information and the contour of the Rubik's Cube, to achieve automatic recognition of Rubik’s Cube. Experiments show that our method can indeed accurately acquire the state of the Rubik's Cube, and the user can quickly and accurately restore the Rubik's Cube under the guidance system designed based on the proposed method. Therefore, the proposed method demonstrates that only using a single sensor can reach the same acquisition results as multi-sensors have achieved. It points out another feasible direction for state acquisition, that is, to make full use of one sensor data.

Acknowledgment

This work was supported in part by the Zhejiang Provincial manufacturing high-quality development industrial chain collaborative innovation project under Grant No. 2110-330108-07-02-205754, and in part by the Zhejiang Provincial Science and Technology Program in China under Grant No. 2021C03137.

References

[1] Qi Cao, “Research and Application of Intelligent Algorithm in Automated Warehouse System” MS thesis. South China university of technology, 2020.

[2] Mun-Kyu Lee, Jeong-Won Pyo, Sang-Hyeon Bae, Sung-Hyeon Joo, and Tae-Yong Kuc, "Traffic Light Recognition for Autonomous Driving Vehicle: Using Mono Camera and ITS," Journal of Image and Graphics, Vol. 10, No. 3, pp. 102-108, September 2022.

[3] Takehito Ogata, "Monocular-Based Drivable Area Segmentation by Fusing 3-D and Texture Information," Journal of Image and Graphics, Vol. 9, No. 4, pp. 140-145, December 2021. doi: 10.18178/joig.9.4.140-145.

[4] Xin Wang, “Design and Implementation of Manufacturing Execution System Based on SCADA for Discrete Manufacturing Industry” MS thesis. East China normal university. 2021.

[5] Xihong Zhao, “Study on the Law of Motion Transformation of Third Order Rubik's Cube Mechanism” MS thesis. Yanshan University. 2021.

[6] Ziwei Chen, “The User Experience Improvements for Smart Cube” MS thesis. South China university of technology, 2020.

[7] Yang Zhao, Shengfeng Min, Dazhou Li, “Research on License Plate Location Algorithm Based on HSV Spatial Color and Texture Features” Journal of Shenyang university of chemical technology. 2022, 36(01): 90-95, doi: 10.3969/j.issn.2095-2198.2022.01.015.

[8] Lei Su, “Algorithm Implementation of a Two-arm Puzzle Cube Robot” MS thesis. Hebei University of Engineering. 2020.

[9] Guiping Lu, Kai Cheng, Zeqi Luo, Rongxin Wu, Yongjian Chen, Shishuo Kang, “Research on Restoration Algorithm of Double Arm Rubik′s Cube Robot Based on Kociemba Algorithm,” Mechanical & Electrical Engineering Technology. 2021, 50(09):100-103, doi: 10.3969/j.issn.1009-9492.2021.09.027.

[10] Zijun Guo, “Application of augmented reality technology in virtual disassembly training system” MS thesis. North University of China. 2022.

Authorized licensed use limited to: Universitatea Tehnica din Cluj-Napoca. Downloaded on March 13,2024 at 15:15:51 UTC from IEEE Xplore. Restrictions apply.