Proceeding of the IEEE
International Conference on Information and Automation
Shenzhen, China June 2011

Design of an Embedded Vision System for the Rubik's Cube Robot

Xin Hu, Shuanshuan Chen, Lei Nie, Zhan Song
Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences,
The Chinese University of Hong Kong
NO.1068,XueYuanRoad,NanShan District, Shenzhen, China
{xin.hu;ss.chen;lei.nie;zhan.song}@siat.ac.cn

***Abstract* – This paper presents a novel vision system design for the Rubik's cube robot. The vision system is implemented on an ARM9 platform. The s3c2440 development board was cut to meet hardware system, including the core-board, the bottom board, touch screen, USB camera and other interfaces. The principle of the vision system can be described as follows: Six image of the Rubik's cube are captured by the USB camera. Each block on the cube is segmented and determined via a proposed image segmentation algorithm and labeled. And then, an artificial intelligence search reduction algorithm is introduced to restore the Rubik's cube.**

Index Terms - Rubik's cube restore, image segmentation, ARM9.

I. INTRODUCTION

With the development of automatics, computer vision technologies have been widely applied in more and more applications. In the robotics research domain, as a principle perception technique, computer or machine vision has attracted more attentions. Compared with PC-based image processing and vision systems, embedded platform has the advantage of better real-time performance [1], high level of integration and supports multi-tasks.

Embedded platform based on image processing system is the future image processing development trend. The research on how to combine the embedded platform and image processing would have the important guiding significance on rapid developing the image applications.

In this paper, we present a novel embedded vision system design for the for the Rubik's cube robot. The vision algorithms are implemented with an ARM9 platform [3]. The processing results are used in the field of artificial intelligence search area. Each cube face's color is captured by a USB camera, and then an image segmentation algorithm is conducted to label the cube faces accordingly [5]. Finally, the Rubik's cube reduction algorithm is applied to restore the Rubik's cube [9].

II. IMAGE ACQUISITION

Image preprocessing is the first step to image processing. We apply ARM9 development board to obtain the images. S3c2440 development board is composed by the core board and the bottom board. Bottom board is the expansion of the core board. The system takes full advantage of the configurability of the embedded board. NandFlash, touch screen and a USB camera with 30fps are our needs.

A. Core Structure Diagram

As can see from the Fig.1, the core board includes Samsung's s3c2440A microprocessors, 256M NandFlash, 64M SDRAM, 12M passive crystals and related interfaces.

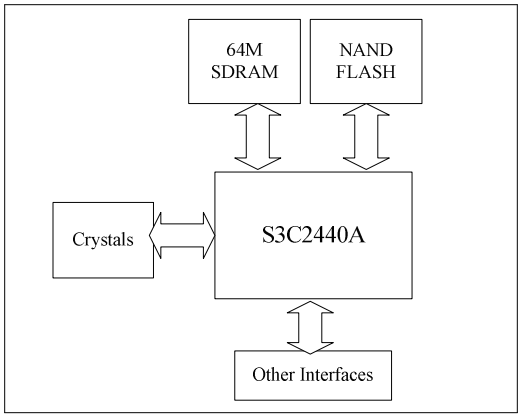

Fig. 1 The external structure of core board

B. *Bottom Board Structure Diagram*

According to our need, we cut off some unwanted interfaces so the outside circuit can be configured, Touch Screen, Network Interface, USB camera and 3D cube are added in. The structure diagram is as shown in Fig.2.

978-1-61284-4577-0270-9/11/$26.00 ©2011 IEEE

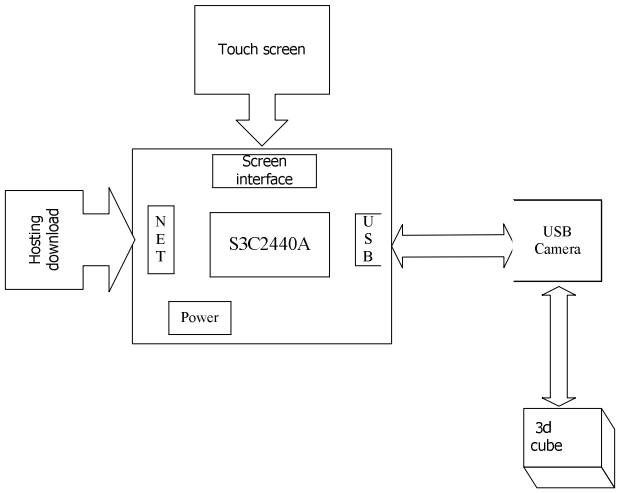

Fig. 2 Rubik's cube robot hardware experiment platform

C. V4L2 USB Camera Driver

Video4-Linux2 is a kernel driver under Linux for video acquiring. It gives us a video capture application program interface, and then with the help of image acquisition card, we could realize the image capture and compression. It has been widely used in Linux embedded system. This system is Linux-based application developed for video capture. Further advanced application could be applied based on this embedded system. V4l2 device acquisition flow chart is shown in Fig.3:

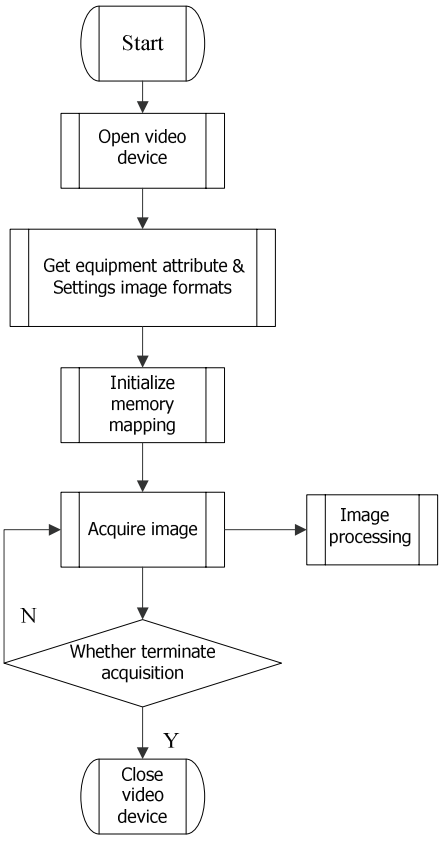

Fig. 3 Video acquisition flow chart

(a) Open video device:

The camera device descriptor fd could be returned by using open(dev_name, O_RDWR, 0) function.

(b) Obtain Settings image attribute format:

V4l2_format is a struct which can be used to set image property, size and format, such as: YUYV format or MJPEG format, with size of 320×240 pixels.

(c) Memory mapping:

V4l_resquestbuffers is a struct to set up memory mapping model and the quantities of cache.V4l_buffer sets the memory mapping between the device driver and application.

(d) Image capture:

The image information which captured by using ioctrl and memcpy functions is stored in the memory buffer, leaving the algorithm to handle this information.

(e) Close device:

Close function is called when finishing video data capture.

III. IMAGE RECOGNITION

Image segmentation and recognition is a crucial step in this system. The classification results are the key to determine whether the artificial intelligence search solution is right or not. Improved fuzzy c-means algorithm (FCM) is used in the system for image recognition.

A. Algorithm Description

Fuzzy c-means clustering algorithm put n vectors into c fuzzy groups, and find out each group's clustering center, minimize the similar function. Among FCM, each given data is set in 0 to 1, and then the fuzzy partition is introduced. For membership degree matrix, all matrix elements are normalized according to the sum of the corresponding column [10].

$$\sum_{i=1}^{c} w_{ij} = 1, \forall j = 1, ..., n \tag{1}$$

So, the general objective function is as follow:

$$J(W, c_1, ..., c_c) = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \sum_{j}^{n} w_{ij}^m d_{ij}^2 \tag{2}$$

Among (2), w_{ij} is between 0 to 1. c_i denotes the clustering center. $d_{ij} = \| c_i - x_i \|$ denotes the Euclidean distance between c_i and x_i; $m \in [1, \infty)$ denotes a weighted index.

In order to calculate the minimum, a new target function is constructed as (3). Then we simplify

$$\overline{J}(W, c_1, ..., c_c, \lambda_1, ..., \lambda_n) = J(W, c_1, ..., c_c) + \sum_{j=1}^{n} \lambda_j (\sum_{i=1}^{c} w_{ij} - 1) \tag{3}$$

to the formula below:

$$\sum_{i=1}^{c} \sum_{j}^{n} w_{ij}^m d_{ij}^2 + \sum_{j=1}^{n} \lambda_j (\sum_{i=1}^{c} w_{ij} - 1) \tag{4}$$

λ_j is $j \in [1,n]$ Lagrange constraint factor. For every input, we derive them. So the necessary conditions for (4) to obtain minimum are:

$$c_i = \frac{\sum_{j=1}^{n} w_{ij}^m x_j}{\sum_{j=1}^{n} w_{ij}^m} \tag{5}$$

And

$$w_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d_{ij}}{d_{kj}} \right)^{2/(m-1)}} \tag{6}$$

The two necessary conditions make this algorithm become an iterative process. The specific step as follows:

Step 1: use the random number between 0 and 1 to initialize the membership matrix U, making it satisfy formula (1) the constraint condition

Step 2: use (4) to calculate the clustering center c_i, which i = 1,... , c.

Step 3: according to (2) , calculate the value function. If it's less than a certain threshold, or the value change is relatively less than a threshold compared to the last value function, then the algorithm stops.

Step 4: use (6) to calculate the new W matrix, then return to step 2.

The clustering center can also be initialized first , then executes iterative process.

The result of the fuzzy c-means clustering algorithm is as below:

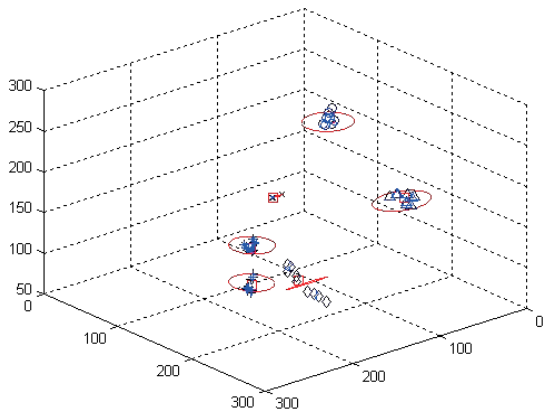

Fig. 4 The results by run FCM

As we can see, 6 red circles shown in the figure are the result of the classification. So the algorithm basically takes effect.

B. Algorithm Improvement

FCM is the improvement to the earliest k-means algorithm, which took some effects, but also has its flaws. As we all know, the fuzzy clustering target is a Non-Convex function, but the calculating process of the FCM is an iterative mountain-climbing process. The process can be fallen into the local extreme value point easily, which makes the global optimal solution unreached. Nevertheless, we should ensure the clustering results are all correct so as to make the next step calculating correctly.

Therefore, the input data is drawn our attention for the improvement. The specific way is:

Firstly, convert the RGB space to HSV. We chose the V component and set a benchmark to V, then convert back to RGB space. After that, the value beyond the benchmark would be set to the benchmark value, and two other components will be normalized correspondingly. Through this change, we can both remove the effects of light-uneven and reduce the disparity of the same class and enhance the disparity of different class.

Secondly, use the average of the neighborhood pixels to eliminate the error. One pixel stands for the RGB could result in the error value. 10x10 neighborhoods (100 pixels) were taken to calculate the real RGB value to avoid the situation mentioned before. The average value of 100 pixels can guarantee the accuracy of the RGB value in general.

The below graph illustrates the result of improved-FCM algorithm:

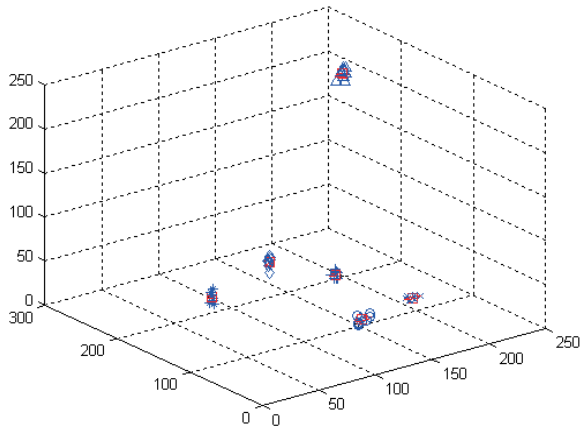

Fig. 5 The results by run the improvement FCM

As shown in Fig. 5, the spaces distance between the 6 centers are obvious. The result is satisfactory.

Some improvement had been made through the above figure by running the improvement FCM compared to the original FCM.

IV. Rubik's Cube Restore

A model must be built if we want to restore Rubik's cube.

A. Establish the Rubik's Cube Model

Two principles should be taken into consideration when establishing the Rubik's cube mode.

The nodes derived in mode operation process should be as few as possible, in order to reduce the computational cost in the node-chosen process of artificial intelligence search. The

color property of each square cube and relative position could be recorded. The Rubik's cube has the following features:

(a) It has 6 faces, denotes 6 different colors. Each side has 9 color pieces of the same size. We treat each color piece as an element, so we get a total of 6x9 = 54 elements, namely, 54 labels.

(b) Rubik's cube has three forms of cube block; they are the center block, the edge block and the corner block. Separately counts 6, 12 and 8.

(c) The color and position of the center blocks in Rubik's are always unchanged, and also have the corresponding relations. White to yellow, orange to red, green to blue.

(d) In addition, there are some universal symbols representing some meanings. We use: T, F, R, L, B and D denoting top, front, right, left, back and bottom sides relatively. Then '+' ,'-' and '*' mean turn clockwise 90 °, turn counterclockwise 90 °, turn clockwise 180 ° respectively. For example: 'T +' means turn top face clockwise 90 °, the others by analogy.

In the practical situation, a struct is used to store the Rubik's cube's information. Struct Square denotes one color piece object, and struct Side denotes one face object. So every time when the Rubik's cube rotates, the position and the color of each face, block can be stored in these structs.

When solving the Rubik's cube, we decompose the whole face's rotation into two rotations: the cube block rotating and the translation of cube block element in the face.

The cube block element's rotation is the rotation around the center axis, which divided into X, Y, Z three axis. As shown in Fig. 6.

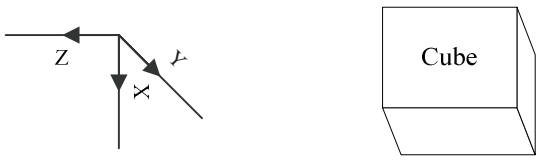

Fig. 6 Cube block

The rotation of cube block, which we need not consider the process, only need to know the result of the rotation. For example: 'F+', front face rotates 90°clockwise. Then result is: Originally left color becomes top surface color, originally top surface color becomes the right color, originally the right color becomes underside color, originally underside color becomes left color. The color of front and back face do not change. Using this method, the 3D cube's rotating can be considered into 4 sides color's transformation.

Translation of the cube block element: actually, it is a loop movement along the Rubik's cube rotating plane. The color of center blocks are unchanged, the other 8 blocks go to clockwise (counterclockwise) twice as shown in Fig. 7:

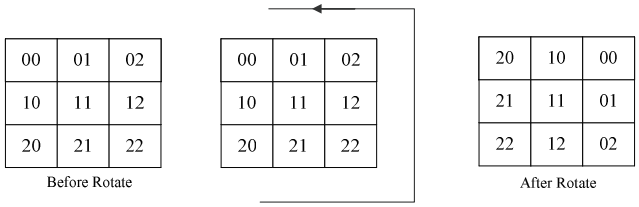

Fig. 7 Loop movement

At this point, the 3D matrix mathematical mode of Rubik's cube has been built.

B. *Solve Rubik's Cube*

Rubik's cube solution adopted the foundation of face first, layer first, corner first.

Step 1): Align the edge of T blocks, then align the angle pieces, till now, the top face gets ready.

Step 2): Align the middle layer's edge blocks.

Step 3): Align bottom edge blocks.

After completion the above steps, the top face and the two-tier layer of the 4 sides faces have been aligned.

Step 4, the last step is to align the 4 bottom corner blocks so as to restore the Rubik's cube.

The figures of the Rubik's cube before and after restoring are as follows:

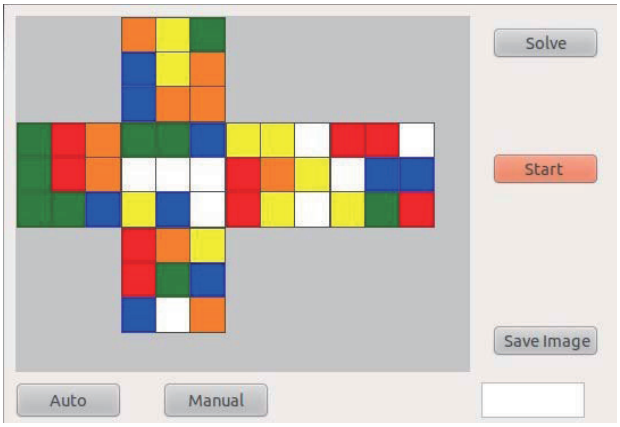

Fig. 8. Before restoration

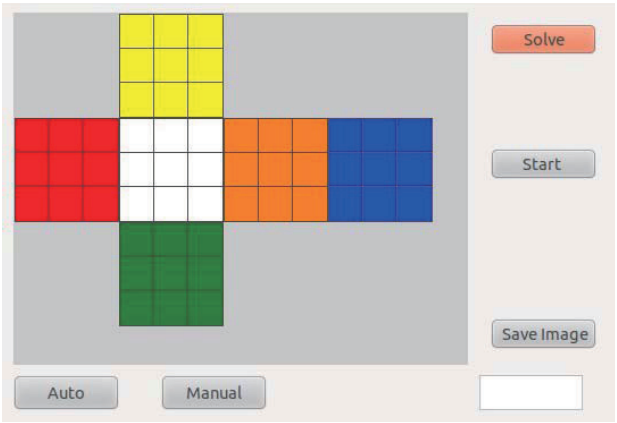

Fig. 9. After restoration

. CONCLUSION

We proposed the automatic method for the restoration of the Rubik's cube. The experimental results validate the practicability of our system. This method is reasonable and effective. From the figures above, this conclusion could be reached: based on s3c2440 hardware platform, we can restore the Rubik's cube automatically by using image recognition algorithm and Rubik's cube reduction algorithm in a short time.

Nevertheless, because of the time and the algorithm accuracy, some limits still exist in some occasions, which needs further improvement and exploration.

ACKNOWLEDGMENT

The work described in this article was supported partially by the grants from the National Natural Science Foundation of China (NSFC, grant no. 61002040, 60903115) and partially by NSFC-Guangdong (grant no. 10171782619-2000007).

REFERENCES

[1] S. D Wei, "The Application Development of Embedded Linux ," Post and Telecom Press,2008

[2] MONGA V. Perceptually based methods for robust image hashing, Ph.D. Dissertation, *University of Texas, Austin,* 2005.

[3] B. H Song, "Linux Device Driver Detailed Development," Post and Telecom Press, 2008

[4] H. Chan, and W. W Bledsoe. "A Man-Machine Facial Recognition System: Some Preliminary Results," *Panoramic Research Inc. PaloAlto,* CA, 1965.

[5] Y. Fukuyama. M. Sugeno. "A new method of choosing the number clusters for the fuzzy C-means method". *In: Proceeding of Fifth Fuzzy System Symposium.* 1987:247 – 250.

[6] W. H. H, K. Y, and W. Z, "An embedded control system for mobile robots", *High Technology Letters,* vol. 17, no. 6, 2007, pp. 595-599

[7] T. Yang. "Using computational verbs to cluster trajectories and curves," *International Journal of Computational Cognition, December 2006, VOL.4(4): 78-87.*

[8] B. N Li. "Weighted fuzzy c-means clustering," *Fuzzy System with Mathematics,* 2007, (01)

[9] X. J Peng, The Artificial Intelligence Model Program based on Turbo C 2.0, *Guangdong University of Technology,* 2005

[10] X. Y Sha, Y. He, "The weighted fuzzy c-means clustering image segmentation algorithm," *Journal of Naval Aeronautical Engineering Institute,* Vol. 2, no. 3, May. 2007