Volume 2, Issue 2, February 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Locating an Object of Interest in an Image Using its Color Feature

M.Tirupathamma
Assistant Professor in ECE Department
JNTUHCEJ
Kondagattu, Jagtial, Karimnagar, India.

M.Ravinder
Associate Professor in CSE Department
Sree Chaitanya College of Engineering
Karimnagar, Andhra Pradesh, India.

Abstract: Nowadays image understanding is the great challenge in the image processing. Object location is an important but challenging research. The object location can be done using the colour segmentation, shape signature. Colour is of interest to those working in computer vision largely because it is assumed to be helpful for recognition. Any single object can project infinity of image configurations to the retina. The orientation of the object to the viewer can vary continuously, each giving rise to a different two-dimensional projection. The object can be occluded by other objects or texture fields, as when viewed behind foliage [1]. In this paper we implemented an algorithm for finding and locating an object of interest with the help of colour features. We first implement the algorithm on an image containing the object of interest. In our algorithm we collect valuable image metadata, such as which pixels belong to which object in the image. We first try to find out the area in which our object of interest presents in the image which involves a number of operations. Then we try to locating the object with a marker by finding the centroid of the object.

Keywords: object, centroid, threshold, locating, pixel.

I. INTRODAUCTION

Object location involves the task of identifying a correspondence between a 3-D object and some part of a 2-D image taken from an arbitrary viewpoint in a cluttered real-world scene. Object detection and tracking have applications in robotics, surveillance, and other fields. As the number of consumers who own computers equipped with video cameras increases, so do the possibilities of using the GPU for tracking objects for games and to enable new user-interaction models. Object Recognition is important in many Computer Vision problems. The problem is defined as classifying new images given a set of training images. The output of the classification can be a label for the whole image, a bounding box for the object, fine segmented boundary or the object with all of its parts. There are two different types of Object Recognition in terms of labels assigned to images. Some Object Recognition systems recognize specific objects and some others detect the object's category. For example, the output can be whether the image is an specific car or if it has any car in it. In this project we focus on fine-grained visual object categories in which the categories are somewhat similar to each other and focus is on the little differences between the categories.

In this paper we propose and implement an algorithm for locating an object of interest with the help of colour features of the object in an image. In our algorithm we collect valuable image metadata, such as which pixels belong to which object in the image. We first try to find out the area in which our object of interest presents in the image which involves a number of operations. Then we try to pointing the object with a marker by finding the centroid of the object. In section II we discuss about object detection followed by section III discusses about our algorithm and in section IV we discuss our experiment and results, in section V we conclude and discuss about future work.

II. OBJECT DETECTION

The object location can be done in two ways: colour segmentation and shape analysis. Colour image segmentation allows identifying the objects in the image. Recent work includes variety of techniques: stochastic based approaches [1], energy diffusion [2] and graph partitioning [3]. In this paper we have used colour image segmentation using the thresholding.

A. *Thresholding*

Thresholding is the simplest method of image segmentation for a gray image, thresholding can be used to create binary images. During the thresholding process, individual pixels in an image are marked as "object" pixels if their value is greater than some threshold value (assuming an object to be brighter than the background) and as "background" pixels otherwise. This convention is known as *threshold above*. Variants include *threshold below*, which is opposite of threshold above; *threshold inside*, where a pixel is labeled "object" if its value is between two thresholds; and *threshold outside*, which is the opposite of threshold inside. Typically, an object pixel is given a value of "1" while a background pixel is given a value of "0." Finally, a binary image is created by coloring each pixel white or black, depending on a pixel's labels [4].

B. *Threshold selection*

The key parameter in the thresholding process is the choice of the threshold value (or *values*, as mentioned earlier). Several different methods for choosing a threshold exist. Users can manually choose a threshold value, or a thresholding

algorithm can compute a value automatically, which is known as *automatic thresholding*. A simple method would be to choose the mean or median value, the rationale being that if the object pixels are brighter than the background, they should also be brighter than the average. In a noiseless image with uniform background and object values, the mean or median will work well as the threshold, however, this will generally not be the case. A more sophisticated approach might be to create a histogram of the image pixel intensities and use the valley point as the threshold. The histogram approach assumes that there is some average value for the background and object pixels, but that the actual pixel values have some variation around these average values. However, this may be computationally expensive, and image histograms may not have clearly defined valley points, often making the selection of an accurate threshold difficult. One method that is relatively simple, does not require much specific knowledge of the image and is robust against image noise is the following iterative method:

1. An initial threshold (T) is chosen, this can be done randomly or according to any other method desired.
2. The image is segmented into object and background pixels as described above, creating two sets:
   a. G_1 = {f(m,n):f(m,n)>T} (object pixels)
   b. G_2 = {f(m,n):f(m,n) $\leq$ T} (background pixels) (note, f(m,n) is the value of the pixel located in the m^{th} column, n^{th} row)
3. The average of each set is computed.
   a. m_1 = average value of G_1
   b. m_2 = average value of G_2
4. A new threshold is created that is the average of m_1 and m_2
   a. T' = $(m_1 + m_2)/2$
5. Go back to step two, now using the new threshold computed in step four, keep repeating until the new threshold matches the one before it (i.e. until convergence has been reached).

This iterative algorithm is a special one-dimensional case of the K-means clustering algorithm, which has been proven to converge at a *local* minimum—meaning that a different initial threshold *may* give a different final result.

C. *Thresholding methods*

Thresholding methods are divided into the following six groups based on the information the algorithm manipulates:

1. Histogram shape-based methods, where, for example, the peaks, valleys and curvatures of the smoothed histogram are analyzed
2. Clustering-based methods, where the gray-level samples are clustered in two parts as background and foreground (object), or alternately are modelled as a mixture of two Gaussians
3. Entropy -based methods result in algorithms that use the entropy of the foreground and background regions, the cross-entropy between the original and binarized image, etc.
4. Object attribute-based methods search a measure of similarity between the gray-level and the binarized images, such as fuzzy shape similarity, edge coincidence, etc.
5. Spatial methods that use higher-order probability distribution and/or correlation between pixels
6. Local methods adapt the threshold value on each pixel to the local image characteristics."

D. *Creating the mask*

We create the mask using a Mask from Colour filter that has two input parameters: an input colour and an input threshold. The input colour defines the colour to track. The input threshold defines the range of colour values around the input colour that are included in the calculations for tracking, choosing the threshold is a bit of an art. You'll want to make sure it is small enough to avoid extraneous noise but large enough to allow for lighting variations in the colour of the object that you want to track. Recall that a Core Image filter (in this case, Mask from Colour) performs the high-level tasks of setting up samplers and applying a kernel routine [5].

E. *Finding the centroid*

We compute the centroid of the masked area as a weighted average of all the coordinates in the mask image. If m(x, y) is the value of the mask image at coordinate (x, y), the equation to compute the centroid is

$$\begin{pmatrix} c_x \\ c_y \end{pmatrix} = \frac{\sum_{x,y} m(x,y) \begin{pmatrix} x \\ y \end{pmatrix}}{\sum_{x,y} m(x,y)}.$$

After calculating the centroid of the object, the object with chosen colour is successfully identified and marked with a symbol [6].

III. PROPOSED ALGORITHM

The proposed algorithm is:

Step1: Take the input image. From the input image extract the red (R), green (G), blue (B) parts separately.

Step2: The colour of the object which we are interested can be selected using the thresholding and normalization of the colour.

Step3: Create a masking image by comparing each pixel with a target colour value. Convert pixels that fall within the range to white, and convert those that fall outside the range to black.

Step4: Find the centroid of the target colour. The centroid of the tracked colour defines the centre position for the overlay image. We use a multi pass pixel-processing kernel to compute a location. The output of this phase is a 1x1-pixel image, containing the coordinate of the centroid in the first two components and the area in the third component. We use the area later to estimate the distance of the object from the camera.

Step5: Composite an image over the detected object. Assuming the shape of the object does not change with respect to the frame, and then the change in area of the tracked colour is proportional to the square of the distance of the object from the viewer. We use this information to scale the overlay image, so that the overlay image increases or decreases in size appropriately.

IV. EXPERIMENT AND RESULTS

We implement our algorithm using MATLAB on a sample image where three primary colour images are extracted and then green colour space is taken as a reference for applying the algorithm. In the selected colour map green colour chosen for thresholding and identification of the object. The object with

chosen colour is successfully identified and marked with a symbol.

Fig.1 Actual image on which we implement our algorithm

We implement our algorithm on the image shown in Fig. 1, we locate the green object present in the image with following steps.

A. RGB component extraction

At first we extract the Red component, Green component and Blue component as shown in below figures.

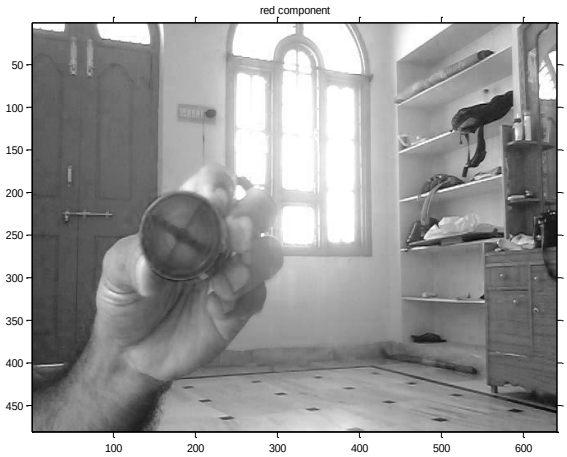

Fig.2 Red component of the image

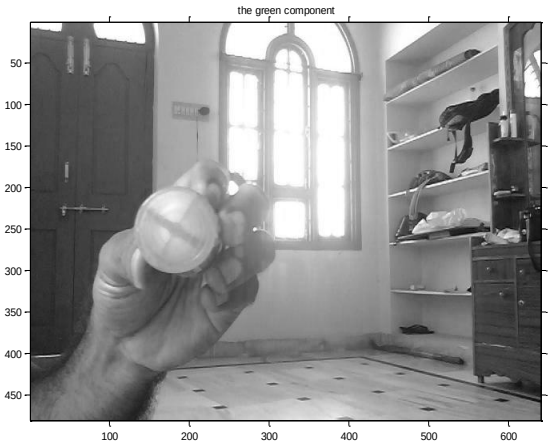

Fig.3 Green component of the image

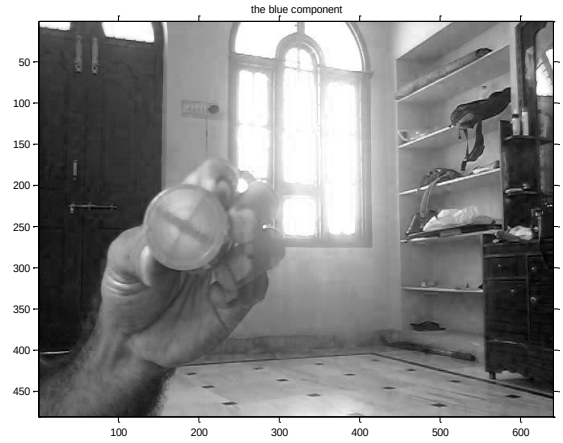

Fig.4 Blue component of the image

B. Thresholding

We implement thresholding operation on green component which is extracted from the actual image for finding the region in which our Green coloured object present. The results after performing thresholding are as shown in below figures.

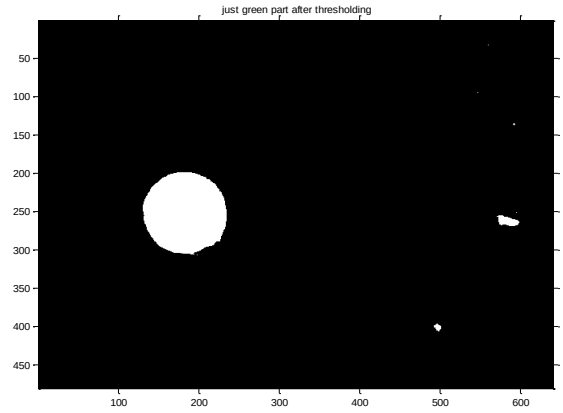

Fig.5 Level 1 result of Thresholding

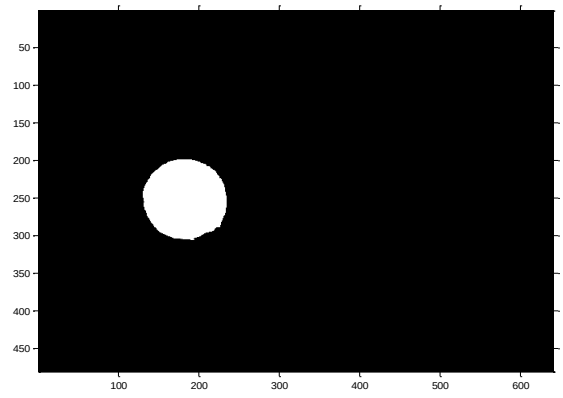

Fig.6 Level 2 result of Thresholding

C. Locating the Object

The next step is to find the centroid and locate the object whose result is as shown in below figure.

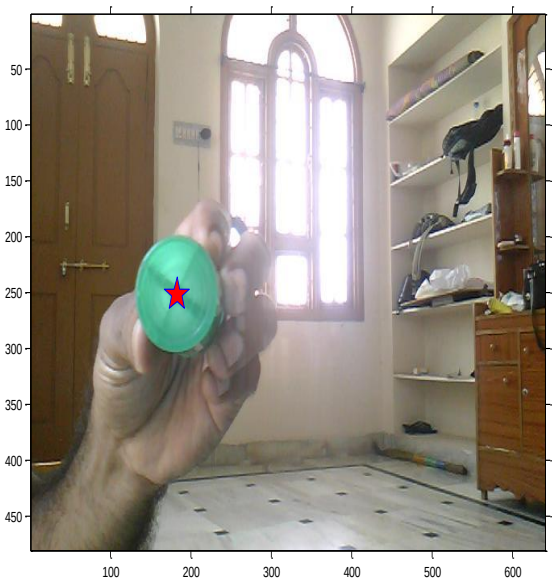

Fig.4 Locating the green object in image

V. CONCLUSION AND FUTURE WORK

In this paper we presented a technique for locating and tracking objects using colour. We used the Core image-processing framework because, as a high-level framework it allows the technique presented here to work on a variety of graphics hardware without the need to make hardware-specific changes. The experimental results showed using colour information significantly increases the recognition rate of the object having same shape.

In future we can also extend the same algorithm for the videos. And we can also use the shape analysis for the detection of the object.

VI. REFERENCES

[1] S. Balongie, Chad carson, Hayit greenspan, and Jitendra Mailik, "color and texture based image segmentation using EM and it's applications to content based image retrieval" Proc. Of ICCV PP 675-682, 1998.

[2] W.Y Ma and B.S Manjulath, "Edge flow: A frame work of boundary detection and image segmentation", Proc of CVPR PP, 744-749, 1997.

[3] J.Shi and J. Malik "Normalized cuts and image segmentation" Proc of CVPR PP 731-737, 1997.

[4] Object detection by color, "proceedings, ICPR'00 Proceedings of international conference on pattern recognition, Volume 1.

[5] http://http.developer.nvidia.com/gpugems3/gpugems3-ch26.html

[6] http://www.cs.columbia.edu/cave/reserch/softlib/coil-100.html