Fuzzy Segmentation for Object Localization from Real Time Video Frames

B Chattopadhyay, A Raychoudhury, A S Chowdhury and A Konar*
Robotics and Vision Laboratory, ETCE Department, Jadavpur University, Kolkata 700 032

Received: 30 October 2001; rev. recd.: 20 February 2003; accepted: 03 March 2003

The paper aims at designing a novel real-time scheme of image segmentation for object localization from dynamic video data stream for applications in mobile robotics. Segmentation of images under non-uniform illumination in real time is not amenable by the conventional algorithms. It employs the classical fuzzy c-means clustering algorithm for image segmentation even in the presence of non-uniform illuminance of dynamic scenes. A novel algorithm for object localization based on the principle of 4-connectivity checking has also been presented in this paper. The proposed algorithm is highly time-efficient and is thus appropriate for applications in interactive multi-agent co-ordination such as, tournament playing, target tracking and bricklaying realized with 2 mobile robots.

Introduction

The problem of real-time image segmentation\(^1\) and object localization\(^17,18\) is an age-old problem in computer vision. With the increasing use of computer vision in industrial automation and robotics, the need for real-time image segmentation is also increasing in pace. The classical approach to image segmentation includes thresholding and edge/region-based segmentation. Thresholding techniques, which can be of 2 basic types: binary and iterative\(^1,2\), have inherent limitations because of their high sensitivity to noise and variation of the lighting conditions during image capture. Consequently, edge-based segmentation results in spurious edge contours, and thus localization of the object from the edges becomes impossible. Region-based segmentation, which is a good choice for images with some beforehand knowledge, is not applicable here for its excessive time requirement in finding the initial seed point.\(^3\) Real time image segmentation, therefore, still remains as an open and challenging area of research. The paper explores the possibility of the well-known fuzzy c-means (FCM) algorithm\(^4,5\) in real-time image segmentation. It may be added here that FCM has already proved itself successful in soft pattern classification\(^6\), and consequently, its significance as a pattern classifier over the hard k-means algorithm\(^7\) or the Bayesian clustering algorithm\(^8\) is beyond any question.

A digital image, represented by a 2-D array of pixels, may be regarded as a pattern of varying gray levels. Therefore, segmentation, which is concerned with segregation of modules in an image, may be treated as a pattern classification problem. Thus the choice of FCM in image segmentation has a logical basis. Prior to using FCM for image segmentation the features of the image based on which it needs to be segmented into components of interest are to be determined. In this paper we presumed that the users have no prior knowledge about the type of the image. Thus the gray level value of the pixels has been treated directly as the feature of the image. Further, in order to keep the feature vector compatible with the requirement of the FCM algorithm, a mapping of the pixel intensities from the 2-D image to a 1-D feature vector is needed. A linear addressing function\(^9\) has been employed in this paper to map the \((i, j)\)th pixel element from the given image of \((m \times n)\) pixels to the \(k\)-th element of a \((1 \times m \times n)\) array, where

\[ k = (i - 1) \times n + j , \quad \ldots \quad (1) \]

where \(1 \leq i \leq m\) and \(1 \leq j \leq n\).

Once an image is segregated into objects of interest, a template-matching algorithm is invoked to identify the objects of interest in the segmented
image. In our application, we used a fuzzy template-matching algorithm that partitions a given image into equal sized grids and matches the statistical features of the selected grid with the same of the template object using a fuzzy norm \(^{10,11}\). There also exist a vast literature on template matching \(^{12}\), and thus we do not explore the same in this paper. Consequently, to make sense of the capability of the FCM and the localization algorithm without using shape matching, let us consider the problem of locating a black moving robot in its dynamic scene. We may presume that the images grabbed by the camera may also include other dark objects like the robot, but the dimension of the robot among all the dark objects in the scene is the largest. Such an assumption is needed in most vision-based target tracking applications of mobile robots \(^{13}\). In fact the present paper highlights the vision-based localization of moving objects and their direct applications in target tracking.

The localization scheme presented in the paper is considerably different with respect to the existing approaches \(^{14,15}\). The proposed scheme checks the connectivity of the neighboring windows based on some criteria (here pixel intensities). Time-complexity wise this algorithm is very efficient and is thus useful for real-time applications.

**The Experimental Setup**

The experimental test-up includes 2 Radio Local Area Networks (LAN). Each LAN comprises a server and a client. The server, which is realized on a desktop Pentium system, includes an Ethernet card connected to the radio-modem through a hub by Ethernet cables (Figure 1). An omni-directional

![Diagram](image_url)

**Figure 1**— Schematic diagram of (a) the server and (b) the client robot
antenna radiating at a center frequency of 8.2 GHz has been mounted on the radio modem for communication with the client. A Pentium II motherboard housed inside the robot has been configured as the client. The robot also has its built-in radio modem and the antenna system mounted on its top. Prior to our experimentation, we configured the LAN in a manner so that each device including the server, the client and the radio modems have a separate IP address, and a common network address. Both the servers and the clients in the radio-LANs run under Linux operating system.

For testing our programs on image segmentation and localization, we used 2 robots, both of which have been configured as clients in their respective radio-LANs. The first robot takes the role of a target tracker, while the second moving robot is the target (object of interest) in the dynamic scenes captured by the first robot. The first robot has a SONY handy cam type camera and a frame grabber mounted at its top that together acquires real time video data at a sampling rate of 200 ms. The problem in the present context is to segment and localize the target robot from the video data stream received by the tracker. The subsequent task of the tracker is to generate control command\(^{19,20}\) for camera pan-tilt orientation to grab the moving target in the dynamic scene continuously. This part, however, goes beyond the scope of the present paper and thus is not elaborated here.

**Image Segmentation Using the Fuzzy c-Means Clustering Algorithm**

Fuzzy c-means clustering is a well-known unsupervised clustering algorithm. The objective of this clustering algorithm is to classify a given set of \(p\)-dimensional data points \(X = [x_1, x_2, \ldots, x_n]\) into a set of \(c\) fuzzy classes or partitions \(A_i\), represented by clusters, such that the sum of the memberships of any component \(x_k\) in all the \(c\) classes is one. Mathematically, this can be represented by:

\[
\sum_{i=1}^{c} \mu_{A_i}(x_k) = 1, \text{ for all } k = 1 \text{ to } n \quad \cdots (2)
\]

Further, it is necessary that

\[
0 < \sum_{k=1}^{n} \mu_{A_i}(x_k) < n, \text{ for all } i = 1 \text{ to } c \quad \cdots (3)
\]

Given \(c\) classes \(A_1, A_2, \ldots, A_c\) we can determine their cluster centers \(V_i\) for \(i = 1\) to \(c\) by means of the following expression:

\[
V_i = \sum_{k=1}^{n} \left[ \mu_{A_i}(x_k) \right]^{m} x_k / \sum_{k=1}^{n} \left[ \mu_{A_i}(x_k) \right]^{m} \text{ for } 1 < m < \infty \quad \cdots (4)
\]

In order to formulate this, a performance criterion \((J_n)\) is defined as follows: Minimize \(J_n\) over \(V_i\) (for fixed partitions \(U\)) and \(\mu_{A_i}\) (for fixed \(V_i\))

\[
J_n(U, V_1) = \sum_{k=1}^{n} \sum_{i=1}^{c} \left[ \mu_{A_i}(x_k) \right]^{m} \|x_k - V_i\|^2 \quad \cdots (5)
\]

where \(\|\|\) is an inner product induced norm in \(p\) dimension. Differentiating the performance criterion with respect to \(V_i\) treating \(\mu_{A_i}\) as constants and to \(\mu_{A_i}\) treating \(V_i\) as constants and setting them to zero we find\(^{5,6}\):

\[
\mu_{A_i}(x_k) = \left[ \sum_{j=1}^{c} \|x_k - V_j\|^2 / \|x_k - V_i\|^2 \right]^{(m-1)/2} \quad \cdots (6)
\]

The following algorithm for fuzzy c-means clustering has been realized in C for image segmentation.

**Procedure Segmentation**

**Input:** an image in one dimensional vector form \(X\);

**Output:** \(c\) number of segmented images;

**Begin**

submit \(c\) and \(m\);

**Step 1:** Assign \(c\) pseudo-partitions in a manner such that for each element \(x_i\) of the image vector the sum of the membership of \(x_i\) in \(c\) partitions is equal to 1;

**Step 2:** Determine the cluster center \(v_1, v_2, \ldots\), by Eq. (4);

**Step 3:** For each \(x_k \in X\)

If \(\|x_k - v_i\| > 0\) for all \(i\)

Then update membership of \(x_k\) in all classes by Eq. (6);

Else If \(\|x_k - v_i\| = 0\) for some \(i\)

Then set membership of \(x_k\) in class \(j \neq i\) by any non-negative real numbers such that sum of the membership of all classes = 1 and set membership of \(x_k\) belonging to class \(i\) equal to 0.

**Step 4:** Repeat from step 2 until the cluster centers Do not change appreciably in 2 successive iterations;

**End**
The initial pseudo-partitions are needed to initiate the algorithm. As the algorithm iterates the cluster centers gradually move from one position to a new position until no further shift in the cluster centers take place. In each iteration the algorithm computes the membership of each pixel intensity (component of vector X) to occur in c number of classes. Thus when the algorithm ceases, closer pixel intensities fall in one class and thus the algorithm works as an image segmenter.

Experiments with FCM and Results

The fuzzy c-means clustering algorithm for image segmentation has 2 parameters namely, the number of clusters, denoted by 'c', and the exponential weighting factor 'm' over the membership functions. The experiments are performed by gradually varying these two parameters and their effects on clustering are noted. The number of clusters needed is usually determined by the problem in hand. For segregating a dark object from a light background or vice versa, we should select c = 2 and thus obtain 2 clusters, one corresponding to the dark region and the other to the lighter region. Figure 2 shows the results of clustering with c = 2. Further, the value of exponential weighting factor m has been increased in steps from m slightly greater than 1, followed by m = 1.2 and m = 2.5. The variation of m clearly indicates the difference between a hard cluster and a soft cluster. For the purpose of illustration, we have constructed the figures using the membership value of each pixel mapped to gray value levels. The contrasting shades in Figure 2(b) indicate that each pixel belongs to either of the classes with large membership value and with a very small membership value for the other class. This is typical of a hard cluster where the pixels are assigned to either of the two classes. In the subsequent Figures 2(c) and 2(d), the 2 shades become less contrasting. As the shades are representing the membership values, it means that each pixel now have intermediate membership values of belonging to the 2 classes. This is fuzzy clustering where each pixel has finite memberships of belonging to the 2 classes. Thus, we have greater latitude of deciding which pixels to select based on their membership values.

In Figure 3, the number of clusters is 3 and 3 shades- dark, intermediate gray and light represent them. Here, too, we observe that how the value of m alters the final membership values of the pixels. As explained for Figure 2, a value of m close to 1 (m=1.01) generates crisp clusters while m = 1.2 or larger makes the clusters fuzzy.

In all the above experiments, we used gray level of the pixels as the feature of interest and accordingly constructed the feature vector. In Figure
4(b) it is observed that although the clustering is done, we do not get the cluster as expected intuitively. This may be attributed to the fact that the general illumination of the image is so low that the black objects cannot be distinguished from the surrounding floor based on gray level alone. We propose that in such circumstances, some transformation of the gray level values be taken to construct the feature vector. The choice of this transformation is based on some prior knowledge of the object of interest. To illustrate this point, let us consider the case where we are interested in segregating some dark object from a poorly illuminated workspace. The suitable transformation in such a circumstance is to take the logarithm of the gray level values of the pixels. As we know, logarithmic transform expands and emphasizes the lower values and cramps the larger value in a smaller range, thus de-emphasizing them. Figure 4(c) illustrates the process of clustering with gray levels as the elements of the feature vector. It is now observed that this transformation has successfully clustered the 'very' dark objects from the dark environment. Thus, we have a third controlling factor, namely, the transformation on the gray level whereby we may classify the image according to the requirements of the problem in hand.

![Image](a)  ![Image](b)  ![Image](c)

**Figure 4** — Fuzzy C means algorithm applied to isolate very black objects from the image in (a) with (b) grayscale as the feature vector (c) logarithm of the grayscale as the feature vector

### Object Localization

Once the robot and other objects, which have sufficiently low gray scale values, are segmented from the image the task that remains at hand is to localize the robot solely from the other black or near black objects. An intelligent algorithm is required for localization of the object in the dynamic scene. In the present scenario the conventional region-growing technique fails because the seed point, which needs to be initialized in the algorithm, cannot always be located within our object of interest. Moreover, quite high time-complexity of the algorithm renders it automatically vulnerable for real time applications like navigation or target-tracking.

After considering all these constraints, a novel algorithm for object localization has been devised. One useful feature that is used at this point is the size information of the object under consideration. The essences of this algorithm lie in splitting the FCM clustered image into some fixed size windows/blocks and determine whether each window is red. The next task is to identify the connected windows by testing the 4-connectivity between them. Then the 4-connected windows are clubbed together to form the separate geographic regions. The selection of the experimental window size is a pertinent parameter to be determined judiciously. A large sized window may include more than one object. For example, a large window may include the target object with others. A small window size is good for accuracy, but a too small window demands a significant computational time. The size of the window chosen in the present application is of 12 x 16 pixels. In fact, the choice of 4-connectivity testing over that of the 8-connectivity testing for identifying geographically connected regions is also guided by the same consideration that the whole computation process should be amenable to a real-time application. A procedure is presented below, describing the various steps of the object localization process.

**Procedure Localize Object**

**Input:** FCM clustered image (containing the robot and other black objects as one cluster and every other object as another cluster)

**Output:** An image containing localized target object, well separated from noise and other objects having similar type of coloration as that of the target.

**Begin**

**Step1:** partition the input image into m x n number of equal sized blocks;
Step 2: For each block (i, j) Do
    Begin
        For each pixel within a block Do
            Begin
                If intensity of pixel ≤ 20
                Then declare it dark and increase
                dark_pixel_count_block(i, j) by 1;
            End For;
        Step 3: If dark_pixel_count_block(i, j) > 50
        Then mark block (i, j) as dark;
    End For;
    Step 4: k = 1;
    Repeat
        Region[k] = any dark block(i, j);
        Region_count[k] = 0;
        For each dark block (i, j) in Region[k]
            If its neighbors are dark,
            Then do
                Begin
                    Region[k] = Region[k] U (block(i, j));
                    Region_count[k] = Region_count[k] + 1;
                End For;
                Mark Region[k] dark;
                k = k + 1;
        Until no dark blocks remain;
        k = k_max;
    Step 5: For k = 1 to k_max
        Identify the Region with the largest Region_count[k] and
        Call it target;
    End For;
End

The procedure Localize_Object aims at identifying the locations of the target object in a segmented image. In the present context, the target object has been assumed to occupy the largest area. Thus, identifying the largest dark region in the image suffices our purpose.

The procedure comprises of 5 main steps. In step 1 we partition the given segmented image into m.n number of equal sized blocks. Step 2 of the algorithm identifies dark pixel based on their intensity values. For this implementation, we select a threshold of 20. Thus if pixel intensity is less than 20, it is declared dark. The dark pixel count in each block is also determined in this step. Step 3 marks a block dark if its dark pixel count exceeds 50. Step 4 of the procedure assembles neighborhood dark blocks in the segmented image into regions such that each two regions are disjoint. In step 5 of the algorithm we declare the largest region as the target object. The localization of the robot from the segmented image of Figure 2(b) is presented in Figure 5 for convenience.

In case the largest region does not correspond to the target object, the regions need to be sorted in descending order based on their block counts. Now a shape-matching algorithm may be invoked for comparing the boundary of the target object with each region in the list in sequence. The region having the closest resemblance with the reference object shape is declared as the target.

Conclusions

The process of image localization presented in this paper can be used for real-time execution of vision based target tracking and co-operation schemes in mobile robotics. The novelty of the approach lies in a successful merging of the FCM algorithm for image segmentation with the "windowing-and-connectivity-checking" method for image localization. The time-complexity of the proposed localization algorithm is $O(m \times n)$ only where $m \times n$ denotes the occupied distributed dark windows in the entire image. If linear search was employed the complexity could have been very high of $O(n^2)$, the size of the entire image.

References