INTRODUCTION

Segmentation - a process of grouping parts of the image into units (classes, regions, subsets) that are homogeneous with respect to one or more characteristics (or features)

Examples: (1) when we segment a picture by thresholding its gray level, we are classifying the pixels into "dark" and "light" regions
(2) in edge detection, we are classifying pixels into "edge" and "no edge" classes
(3) in text recognition, we classify pixels into different character classes: "A", "B", "a", "b", etc.
(4) when analyzing aerial photo, we classify pixels into terrain regions: "forests", "urban areas", "bodies of water", "roads", etc.
(5) when analyzing medical images, we classify pixels into anatomical regions, such as "bone", "muscle", "blood vessel", etc.

There is no single standard approach to segmentation. The definition of the goal of segmentation varies according to the type of the data and the type of the applications. Different assumptions about the nature of the images being analyzed lead to use of different algorithms.

There are many methods for segmenting an image into regions, which, subsequently, can be analyzed based on their shapes, sizes, relative positions, and other characteristics. The most commonly used segmentation techniques can be classified into two broad categories (1) region extraction techniques that look for maximal regions satisfying some homogeneity criterion, (2) edge extraction techniques that look for edges occurring between regions with different characteristics.

Thresholding is a common region extraction method. It is based on the assumption that the image has a bimodal histogram and, therefore, contains an
object or objects of interest that can be extracted from the background by a simple operation that compares image values with a threshold level. There are several thresholding methods: global methods based on gray level histograms, global methods based on local properties, local threshold selection, and dynamic thresholding.

**Clustering** is a name of another class of algorithms for image segmentation. Clustering segments the image in terms of sets or clusters of pixels that have strong similarity in the feature space. The basic operation is to examine each pixel and assign it to the cluster that best represents the value of its characteristic vector of features of interest.

Thresholding and clustering operate at the pixel level. Other methods, such as region growing operate on groups of pixels. Region growing is a process by which two adjacent regions are assigned to the same segment if their image values are close enough according to some pre-selected criterion of closeness.

The strategy of edge-based segmentation algorithms is to find object boundaries and segment regions enclosed by the boundaries. These algorithms usually operate on edge magnitude and/or phase images produced by an edge operator suited to the expected characteristics of the image. For example, most gradient operators such as the Prewitt operator, the Kirsch operator, or the Roberts operator are based on the existence of an ideal step edge. Another example is the Hueckel operator, which fits the image function values in a region to an ideal edge element using a least-squares minimization criterion. Other edge-based segmentation techniques are graph searching, and contour following.

Often there are problems associated with edge detection. First of all, it is difficult to define an adequate model of an edge that will hold over the whole image (since most images are not homogeneous) or over all images of a given application. Also, edge detection does not always provide a complete segmentation and further procedures for edge linking and edge cleaning are required to organize the resulting edges.
THRESHOLDING

Thresholding is based on the assumption that a histogram has a bimodal histogram and, therefore, the object can be extracted from the background by a simple operation that compares image values with a threshold level $T$. Suppose that we have an image $F[i,j]$ with the histogram shown below:

The object and background pixels have gray levels grouped into two dominant modes. One obvious way to extract the object from the background is to select a threshold $T$ that separates these modes. Thresholded image $F_T[i,j]$ is defined as:

$$F_T[i,j] = \begin{cases} 
1 & \text{if } F[i,j] \geq T \\
0 & \text{if } F[i,j] < T 
\end{cases}$$ (1)

Thus pixels labeled 1 (or any other convenient intensity level) correspond to objects, whereas pixels labeled 0 correspond to the background.

When $T$ depends only on $F[i,j]$, the threshold is called **global**. If $T$ depends on $F[i,j]$ and some local properties of the neighborhood of pixel $[i,j]$ - for example, average gray level of the neighborhood - the threshold is called **local**. If, in addition, $T$ depends on the spatial coordinates $i$ and $j$, the threshold is called **dynamic** or **adaptive**.

Figure below shows an example of simple thresholding applied to the "Blobs" image. Please note that the edges of the blobs were obtained by a 3x3 Laplacian applied to the thresholded image.
**Multilevel thresholding**

If an image contains more than two types of regions, it may still be possible to segment it by applying several thresholds. Figure below shows a histogram of an image containing two types of light objects on a dark background:
A pixel \([i,j]\) belongs to "object 1" class if \(T_1 < F[i,j] \leq T_2\); to "object 2" class if \(F[i,j] > T_2\), and to the background if \(F[i,j] \leq T_1\).

As the number of region types increases, the peaks become harder to distinguish, and segmentation by thresholding becomes more difficult.

**Smoothing and thresholding**

The gray level subpopulations corresponding to the different types of regions in an image will often overlap and segmentation by thresholding becomes difficult. **Smoothing** an image (by using **average** or **median** filter) before thresholding sometimes helps to alleviate that problem. Figures below illustrate how a 7x7 median filter sharpened the peaks on an image histogram.
Thresholding based on the local properties

- Local properties can be used to aid in the selection of global thresholds; for example: extracting two samples of pixels from an inside and outside regions of the object and selecting a threshold based on a minimum number of misclassified pixels

- For example, table below shows the pixel distributions for sampled regions of background and object. The number of misclassified pixels for different threshold levels $T$ is calculated and plotted for $T$ -- for example, a number of misclassified pixels for threshold level $T=11$ is 20 (sum of the shadowed areas of object and background distributions). the least number of misclassified pixels (18) is obtained for $T=12$. 
<table>
<thead>
<tr>
<th>background</th>
<th>object</th>
<th>Number of misclassified pixels</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>35</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>25</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>15</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>25</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>16</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Semi-automated thresholding based on local properties technique applied to the lymph node images; (a) original lymph node image with two selected pixels: x - inside and y - outside the node, (b) circular regions (circle X and Y) around pixels x and y, (c) segmented lymph node.
**Dynamic (adaptive) thresholding**

In many cases, no single threshold gives good segmentation results over an entire picture. Suppose, for example, that the image shows dark objects on a light background, but it was obtained under conditions of uneven illumination. The objects will still contrast with the background, but both the background and objects may be lighter on one side of the image than on the other. Thus, a single threshold will not separate objects from background. Similar situation occurs when an image contains shadows, or when the picture was obtained by a sensor whose sensitivity varies from point to point.

If the uneven illumination is described by some known function of position in the picture, one could attempt to correct for it using gray level correction techniques, after which a single threshold should work for the entire image. If this information is not available, one can divide the image into blocks and apply threshold selection techniques to each block. If a block contains both objects and background, its histogram should be bimodal, and the valley bottom should yield a local threshold. If a block contains objects only, or background only, it will not have a bimodal histogram, and no threshold will be found for it; but a threshold can still be assigned to it by interpolation from the local thresholds found for nearby bimodal blocks (some smoothing of the resulting thresholds may be necessary, since if a threshold changes abruptly from one block to another, artifacts may result).

Another situation in which local thresholding may be useful is where the objects to be segmented are very small and sparse (bubble tracks, stars, etc.), so that the image consists almost entirely of background, and the objects produce no detectable peak on a histogram.

Example of dynamic thresholding (from Rosenfeld and Kak, Digital Picture Processing, p. 72, Academic Press, 1982):
Fig. 9 Local thresholding. (a) Picture of mechanical parts, with grid superimposed defining blocks. (b) Result of applying a single threshold to (a), obtained by fitting a mixture of two Gaussians to the histogram of (a). (c) Histograms for the individual blocks in (a); the lines provide an indication of the relative scale of these histograms. A threshold was obtained for each block by fitting a mixture of two Gaussians to its histogram. (d) Display of the array of thresholds obtained by interpolation from these block thresholds. (e) Result of applying these thresholds to every point of (a).
The watershed algorithm

A relative of adaptive thresholding is the watershed algorithm. We assume that the objects in the image are of low gray level, on a high gray level background. Figure below shows the gray levels along one scan line that cuts through two objects that are close together.

The image is initially thresholded at a low gray level, one that segments the image into the proper number of objects, but with boundaries that are too small. Then the threshold is raised gradually, one gray level at a time. The objects’ boundaries will expand as the threshold increases. When they touch, however, the objects are not allowed to merge. Thus, these points of first contact become the final boundaries between adjacent objects. The process is terminated before the threshold reaches that gray level of the background - that is, a point when the boundaries of well-isolated objects are set.

Both the initial and final threshold gray levels must be well chosen. If the initial threshold is too low, then low-contrast objects will be missed at first and then merged with nearby objects as the threshold increases. If the initial threshold is too high, objects will be merged from the start. The final threshold value determines how well the final boundaries fit the objects.

In some implementations of the watershed algorithm both the initial and final threshold gray levels are chosen locally (based, for example, on the local properties of an image) - and can vary from object to object.
**REGION GROWING**

While thresholding focuses on the difference of pixel intensities, the region growing method looks for groups of pixels with similar intensities. Region growing starts from the bottom, or individual pixel level, and works upwards. Starting at some seed location (usually provided by an operator), neighboring pixels are examined one at a time and added to the growing region if they are sufficiently similar (based on a uniformity test). The procedure continues until no more pixels can be added.

One example of the uniformity test is comparing the difference between the value of a pixel and the average over a region. If the difference is less than some predefined value (for example, two standard deviations of the mean over the region), the pixel is included in the region; otherwise it is defined as an edge pixel. Depending on the uniformity tests employed, different starting points may not grow into identical regions. Also, if too small a test region is used, a common result is that regions leak out into adjoining areas or merge with different regions.
REGION GROWING

Region growing [24–27] is an approach to image segmentation that has received considerable attention in the computer vision segment of the artificial intelligence community. With this approach, one begins by dividing an image into many tiny regions. These initial regions may be small neighborhoods or even single pixels. In each region, suitably defined properties that reflect membership in an object are computed. The properties that distinguish the pixels inside the different objects might include average gray level, texture, or color information. Thus, the first step assigns to each region a set of parameters whose values reflect the object to which they belong.

Next, all boundaries between adjacent regions are examined. A measure of boundary strength is computed utilizing the differences of the averaged properties of the adjacent regions. A given boundary is strong if the properties differ significantly on either side of that boundary, and it is weak if they do not. Strong boundaries are allowed to stand, while weak boundaries are dissolved and the adjacent regions merged.

The process is iterated by alternately recomputing the object membership properties for the enlarged regions and then dissolving weak boundaries. The region-merging process is continued until a point is reached where no boundaries are weak enough to be dissolved. Then, image segmentation is complete. Monitoring this procedure gives one the impression of regions in the interior of objects growing until their boundaries correspond with the edges of the object.

Region-growing algorithms are computationally more expensive than the simpler techniques, but region growing is able to utilize several image properties directly and simultaneously in determining the final boundary location. Perhaps it shows greatest promise in the segmentation of natural scenes, where strong a priori knowledge is not available.

Figure 18–19 shows four stages in the region growing of one muscle fiber viewed on a microscope slide. In this example, low gradient was the sole region membership property. The lower right quadrant shows the final boundary.
5.1.2 Towards good segmentation

Segmentation is a critical component of a computer vision system because errors in this process will be propagated to the higher level analysis processes and increase the complexity of the subsequent tasks. Ideally the segmented regions within an image should have the following characteristics [3]:

(a) regions should be uniform and homogeneous with respect to some particular characteristic;
(b) region interiors should be simple and without many small holes;
(c) adjacent regions should have significantly different values with respect to the characteristic on which they are uniform;
(d) boundaries of each segment should be simple, not ragged, and must be spatially accurate.

Most image segmentation techniques are ad hoc and domain-dependent. It is difficult to obtain quantitative data on the quality of segmentation as the results are open to subjective interpretation, the only simple criterion available being a measure of the percentage of pixels mis-classified. Good segmentation is somewhat analogous to the definition of ‘real-time’, the segmentation is good if it provides appropriate output for the solution of the problem under investigation.

Achieving all the above desired properties in practice is extremely difficult. Insisting that adjacent regions have large differences in value can cause adjacent regions to merge and boundaries to be lost. Over-dividing or under-dividing regions may cause them to correspond to more than one surface, or conversely surface variability may split a single real surface into several regions.

The problem of segmentation of natural images is basically one of emulating psychological perception and therefore does not lend itself to a purely analytical solution. Any mathematical algorithms must be supplemented by heuristics, usually involving semantics or descriptions about the class of images under consideration.

Sometimes it is appropriate to go beyond simple heuristics and introduce a priori knowledge about the image. In such cases image segmentation proceeds simultaneously with image understanding. In segmentation, a priori knowledge refers to implicit or explicit constraints on the likelihood of a given pixel grouping. Such assumptions often arise from restrictions placed on the image as a consequence of domain-dependent considerations.

Even in a back-lit binary image, for example, where segmentation into regions of background and foreground is a trivial task, it is still necessary to label holes that occur within objects. They have the same pixel intensity values as the general background, but have a quite different significance. Holes can be uniquely identified by observing the hypothesis that they consist of regions of background intensity which is entirely enclosed by foreground intensity.