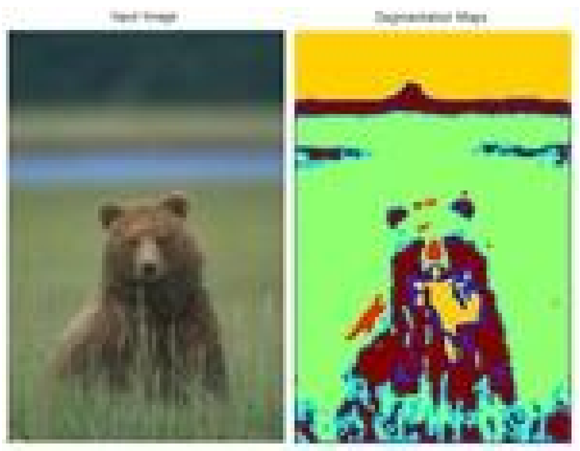
8.Segmentation

Image segmentation is important in many computer visions and image processing application .The goal of image segmentation is to find regions that represent objects or meaningful parts of objects.



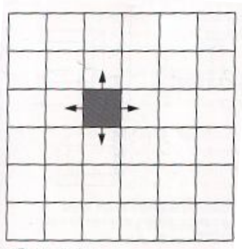
Division of the image into regions corresponding to objects of interest is necessary before any processing can be done at a level higher than that the pixel. Identifying real objects, pseudo—objects, and shadows or actually finding anything of interest within the image requires some form of segmentation.

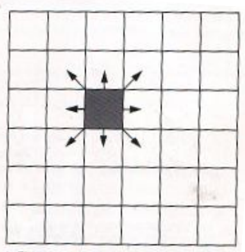
Image Segmentation Methods

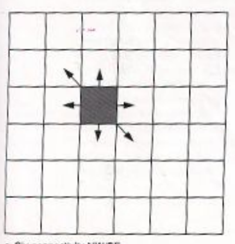
Image segmentation methods will look for objects that either have some measure of homogeneity within themselves or have some measure of contrast with the objects on their border. Most image segmentation algorithms are modifications, extensions، or combinations of these two basic concepts. The homogeneity and contrast measures can include features such as gray level, color, and texture. After we have performed some preliminary segmentation, we may incorporate higher-level object properties, such as perimeter and shape, into the segmentation process.

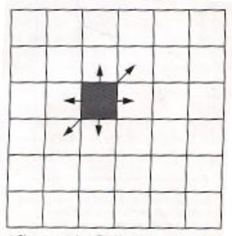
Before we look at the different segmentation methods, we need to consider some of the problems associated with image segmentation. The major problems are a result of noise in the image and digitization of a continuous image. Noise is typically caused by the camera, the lenses, the lighting, or the signal path and can be reduced by the use of the preprocessing methods previously discussed. Spatial digitization can cause problems regarding connectivity of objects. These problems can be resolved with careful connectivity definitions and heuristics applicable to the specific domain.

Connectivity refers to the way in which we define an object. After we have segmented an image, which segments should be connected to form an object? Or, at a lower level, when searching the image for homogeneous regions, how do we define which pixels are connected? We must define which of the surrounding pixels are considered to be neighboring pixels. A pixel has eight possible neighbors: two horizontal neighbors, two vertical neighbors, and four diagonal neighbors. We can define connectivity in three different ways: 1) four-connectivity, 2) eight-connectivity, and 3) six-connectivity. Figure below illustrates these three definitions.

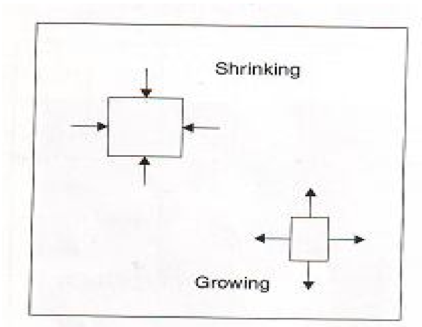

a. Four-connectivity.


b. Eight-connectivity.

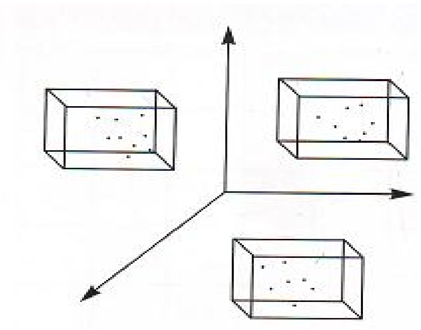

c. Six-connectivity NW/SE.


d. Six-connectivity NE/SW.

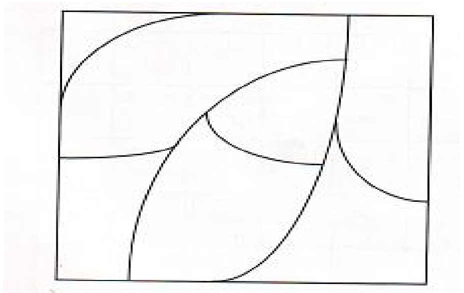
We can divide image segmentation techniques into three main categories (see Figure below): 1) region growing and shrinking, 2) clustering methods, and 3) boundary detection. The region growing and shrinking methods use the row and column (rc) based image space, whereas the clustering techniques can be applied to any domain spatial domain, color space, feature space, etc.



a. Region growing/shrinking is performed by finding homogeneous regions and changing them until they no longer meet the homogeneity criteria.



b. Clustering looks for data that can be grouped in domains other than the spatial domain.



c. Boundary detection is often achieved using a differentiation operator to find lines or edges, followed by postprocessing to connect the points into borders.

1.Region Growing and Shrinking

Region growing and shrinking methods segment the image into regions by operating principally in the RC—based image space. Some of the techniques used are local, in which small areas of the image are processed at a time; others are global, with the entire considered during processing.

Methods that can combine local and global techniques, such as split and merge, are referred to as state space techniques and use graph structures to represent the regions and their boundaries.

In general, the split and merge technique proceeds as follows:

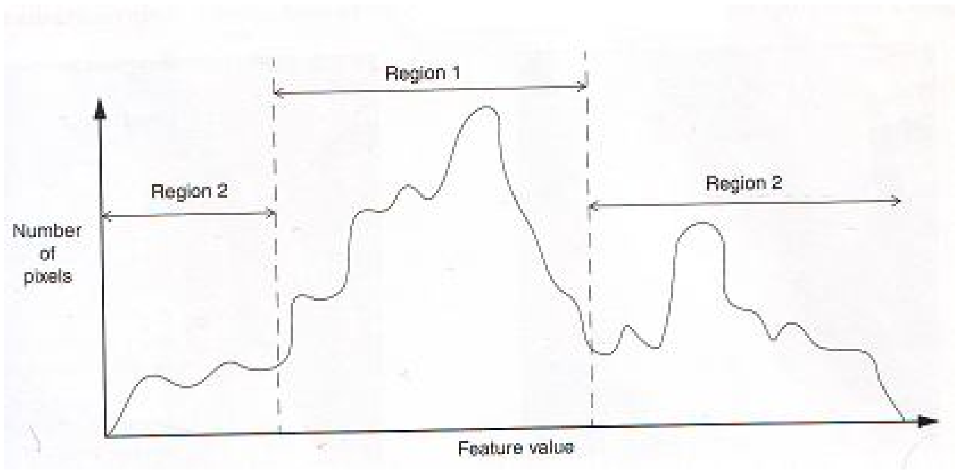
1. Define a homogeneity test. This involves defining a homogeneity measure, which may incorporate brightness, color, texture, or other application-specific information, and determining a criterion the region must meet to pass the homogeneity test.
2. Split the image into equally sized regions.
3. Calculate the homogeneity measure for each region.
4. If the homogeneity test is passed for a region, then a merge is attempted with its neighbor(s). If the criterion is not met, the region is split.
5. Continue this process until all regions pass the homogeneity test.

2.Clustering Techniques

Clustering techniques are image segmentation methods by which individual elements are placed into groups; these groups are based on some measure of similarity within the group. The major difference between these techniques and the region growing techniques is that domains other than the rc-based image space (the spatial domain) may be considered as the primary domain for clustering. Some of these other domains include color spaces, histogram spaces, or complex feature spaces. The simplest method is to divide the space of interest into regions by selecting the center or median along each dimension and splitting it there; this can be done iteratively until the space is divided into the specific number of regions needed. This method is used in the SCT Center and PCT/Median segmentation algorithms.

Recursive region splitting is a clustering method that has become a standard technique. This method uses a thresholding of histograms technique to segment the image. A set of histograms is calculated for a specific set of features, and then each of these histograms is searched for distinct peaks.

Histogram peak finding



The best peak is selected and the image is split into regions based on this thresholding of the histogram. One of the first algorithms based on these concepts proceeds as follows:

1. Consider the entire image as one region and compute histograms for each component of interest (for example, red, green, and blue for a color image).
2. Apply a peak finding test to each histogram. Select the best peak and put thresholds on either side of the peak. Segment the image into two regions based on this peak.
3. Smooth the binary thresholded image so that only a single connected sub region is left.
4. Repeat steps 1-3 for each region until no new sub regions can be created, that is, no histograms have significant peaks.

Two thresholds are selected, one on each side of the best peak. The image is then split into two regions. Region 1 corresponds to those pixels with feature values between the selected thresh olds, known as those in the peak. Region 2 consists of those pixels with feature values outside the threshold.

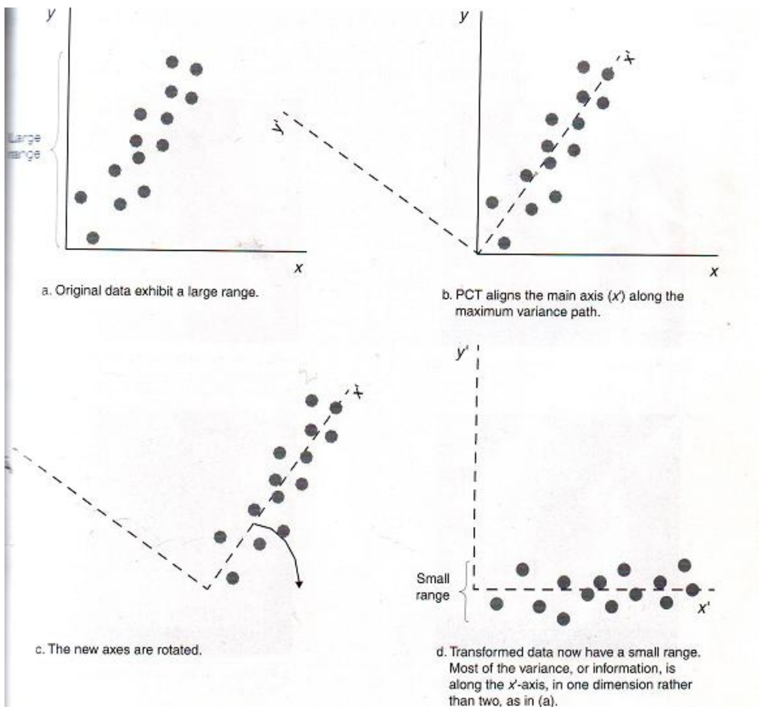
PCT/Median color segmentation algorithm:

This algorithm is based around the principal components transform (PCT). The median split part of the algorithm is based on an algorithm developed for color compression to map 24 bits/pixel color images into images requiring an average of 2 bits/pixel.

The PCT is based on statistical properties of the image and can be applied to any K-dimensional mathematical space. In this case, the PCT is applied to the three-dimensional color space. It was believed that the PCT used in conjunction with the median split algorithm would provide satisfactory color image segmentation because the PCT aligns the main axis along the maximum variance path in the data set.

The PCT/Median segmentation algorithm proceeds as follows:

1. Find the PCT for the RGB image. Transform the RGB data using the PCT
2. Perform the median split algorithm: find the axis that has the maximal range (initially it will be the PCT axis). Divide the data along this axis so that there are equal numbers of points on either side of the split the median point. Continue until the desired number of colors is reached.
3. Calculate averages for all the pixels falling within a single box.
4. Map each pixel to the closest average color values, based on a Euclidean distance measure.


y
Large range
x
a. Original data exhibit a large range.
y
x
b. PCT aligns the main axis (x') along the maximum variance path.
c. The new axes are rotated.
y'
Small range
x'
d. Transformed data now have a small range. Most of the variance, or information, is along the x'-axis, in one dimension rather than two, as in (a).

Principal Components Transform

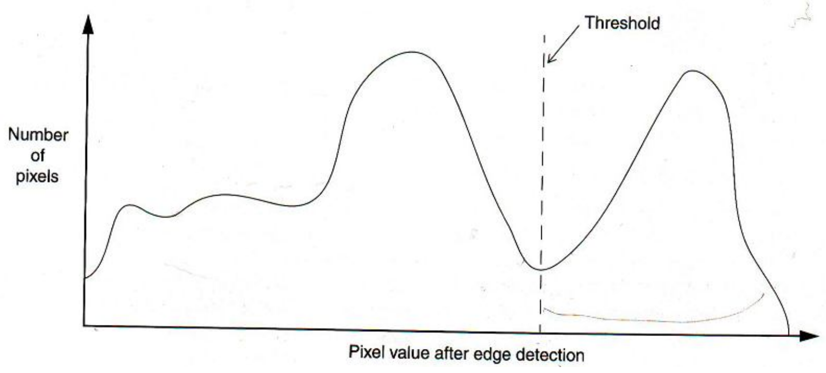
Results of this segmentation algorithm are shown in Figure below.

It is interesting to note that the PCT is also used in image compression (coding) since this transform is optimal in the least-square-error sense.

3.Boundary Detection

Boundary detection, as a method of image segmentation, is usually begun by marking points that may be a part of an edge. These points are then merged into line segments, and the line segments are then merged into object boundaries. The edge detectors are used to mark points, thus indicating the possibility of an edge. After the edge detection operation has been performed, the next step is to threshold the results. One method to do this is to consider the histogram of the edge detection results, looking for the best valley.

Edge Detection Threshold



Often, the histogram of an image that has been operated on by an edge detector is unimodal (one peak), so it may be difficult to find a good valley. This method works best with a bimodal histogram. After we have determined a threshold for the edge detection, we need to merge the existing edge segments into boundaries. This is done by edge linking.

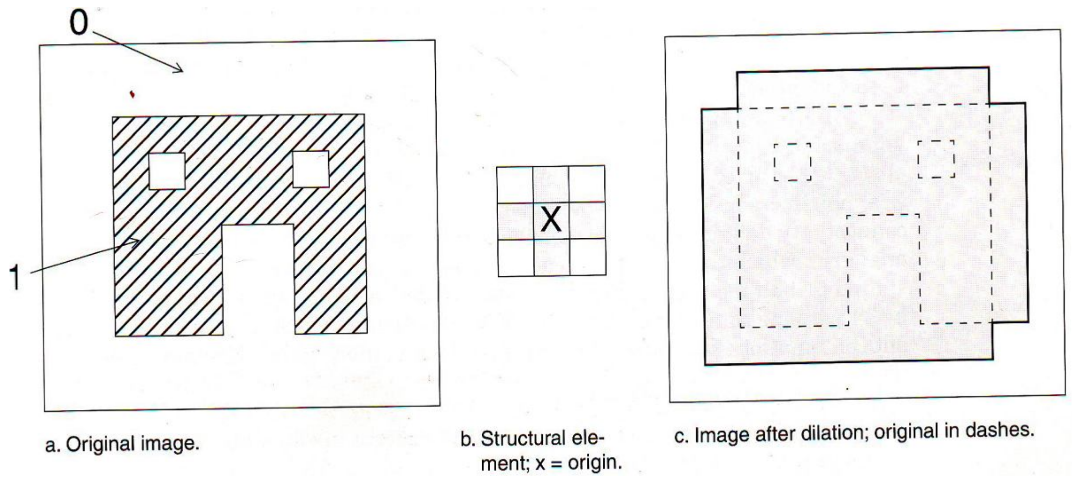
Morphological Filtering

Morphology relates to the structure or form of objects. Morphological filtering simplifies a segmented image to facilitate the search for objects of interest. This is done by smoothing out object outlines, filling small holes, eliminating small projections, and using other similar techniques. Even though this section will focus on applications to binary images, the extension of the concepts to gray-level images will also be discussed. We will look at the different types of operations available and at some examples of their use. The two principal morphological operations are dilation and erosion. Dilation allows objects to expand, thus potentially filling in small holes and connecting disjoint objects. Erosion shrinks objects by etching away (eroding) their boundaries. These operations can be customized for an application by the proper selection of the structuring element, which determines exactly how the objects will be dilated or eroded. The dilation process is performed by laying the structuring element on the image and sliding it across the image in a manner similar to convolution. The difference is in the operation performed. It is best described in a sequence of steps:

1. If the origin of the structuring element coincides with a ‘0’ in the image, there is no change; move to the next pixel.
2. If the origin of the structuring element coincides with a ‘1’ in the image, perform the OR logic operation on all pixels within the structuring element.

An example is shown in Figure below.

Dilation



a. Original image. b. Structural element; x = origin. c. Image after dilation; original in dashes.

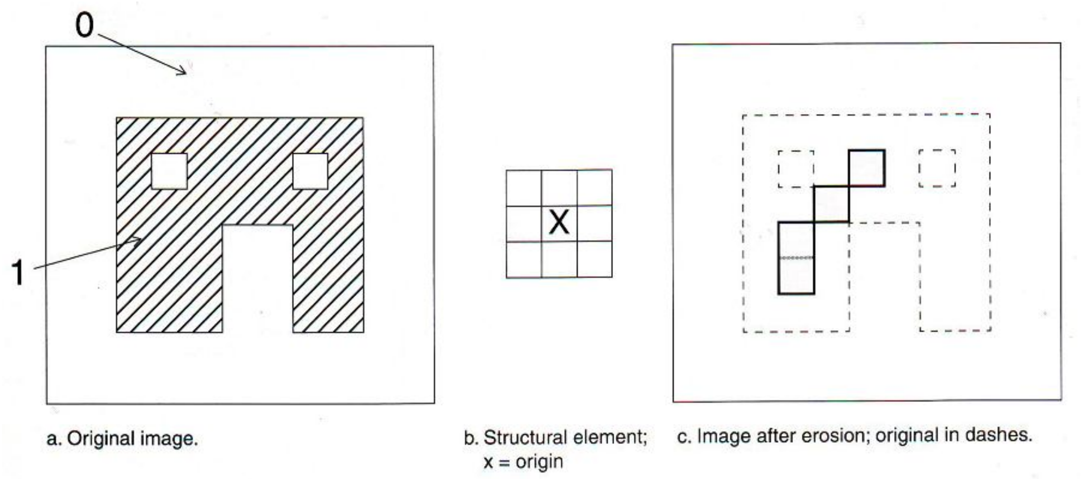
Note that with a dilation operation, all the ‘1’ pixels in the original image will be retained, any boundaries will be expanded, and small holes will be filled.

The erosion process is similar to dilation, but we turn pixels to ‘O’, not ‘1’. As before, slide the structuring element across the image and then follow these steps:

1. If the origin of the structuring element coincides with a ‘0’ in the image, there is no change; move to the next pixel.
2. If the origin of the structuring element coincides with a ‘1’ in the image, and any of the ‘1’ pixels in the structuring element extend beyond the object (‘1’ pixels) in the image, then change the ‘1’ pixel in the image to a ‘0’.

In Figure below, the only remaining pixels are those that coincide to the origin of the structuring element where the entire structuring element was contained in the existing object.

Erosion

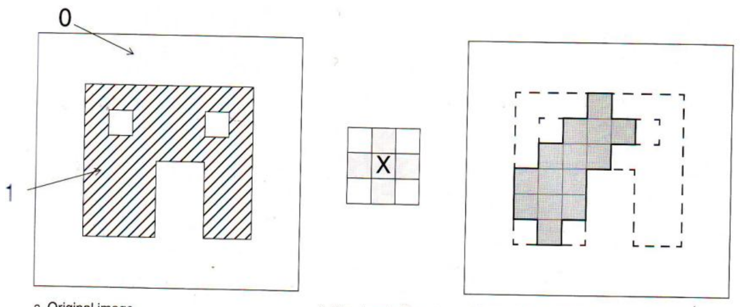


a. Original image.
b. Structural element; x = origin
c. Image after erosion; original in dashes.

Because the structuring element is 3 pixels wide, the 2-pixel-wide right leg of the image object was eroded away, but the 3-pixel-wide left leg retained some of its center pixels.

These two basic operations, dilation and erosion, can be combined into more complex sequences. The most useful of these for morphological filtering are called opening and closing. Opening consists of an erosion followed by a dilation and can be used to eliminate all pixels in regions that are too small to contain the structuring element. See Figure below for an example of opening.

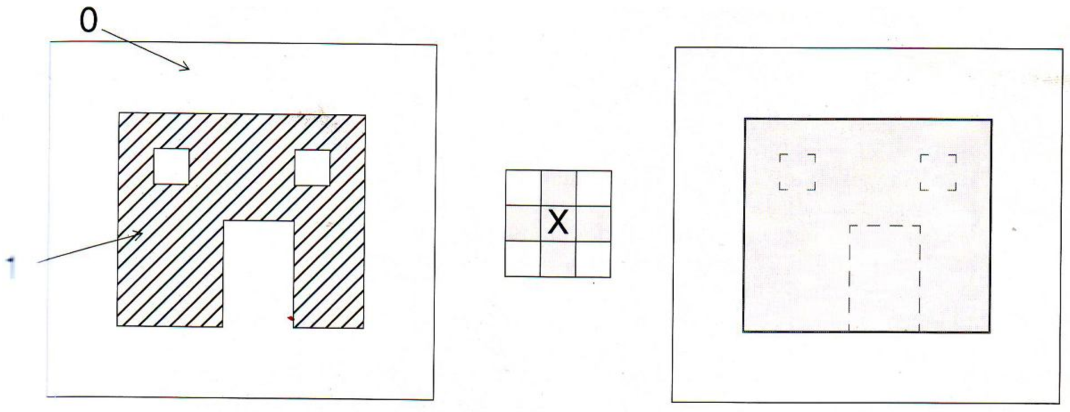
Opening



a. Original image. b. Structural element; x = origin. c. Image after opening; erosion followed by dilation.

Closing consists of a dilation followed by erosion and can be used to fill in holes and small gaps. In Figure below we see that the closing operation has the effect of filling in holes and closing gaps.

Closing

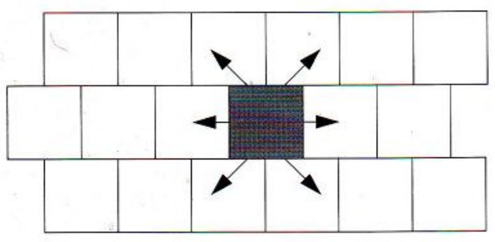


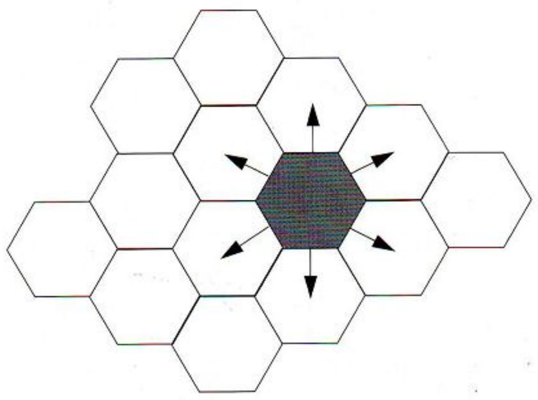
a. Original image. b. Structural element; x = origin. c. Image after closing; dilation followed by erosion; original in dashes.

Comparing Figure closing to Figure opening, we see that the order of operation is important. Closing and opening will have different results even though both consist of an erosion and a dilation.

Another approach to binary morphological filtering is based on an iterative approach. The usefulness of this approach lies in its flexibility. It is based on a definition of six-connectivity; in which each pixel is considered connected to its horizontal and vertical neighbors but to only two diagonal neighbors (the two on the same diagonal). This connectivity definition is equivalent to assuming that the pixels are laid out on a hexagonal grid, which can be simulated on a rectangular grid by assuming that each row is shifted by half a pixel (see Figure below).

Hexagonal Grid


a. Rectangular image grid with every other row shifted by one-half pixel.


b. Hexagonal grid.