## Assignment No 5

### Aim : To implement a program for feature extraction in 2D colour images (any features like colour, texture etc.)

**AIM :** To implement a program for feature extraction in 2D color images (any features likecolor, texture etc.)

**Objective:** To study a program for feature extraction in 2D colour images (any features likecolour, texture etc.)

**Scope** **:** Take a color image and extract color features.

### THEORY

· Multimedia IR

· Techniques to represent audio and visual document

· Two dimensional color images

· Automatic feature extraction

**INPUT**: Color Image

**OUTPUT**: Extraction of features like color and texture

### Feature Extraction

The feature is defined as a function of one or more measurements, each of which specifies some quantifiable property of an object, and is computed such that it quantifies some significant characteristics of the object.

Various features currently employed as follows:

* General features: Application independent features such as color, texture, and shape. According to the abstraction level, they can be further divided into:

* + Pixel-level features: Features calculated at each pixel, e.g. color, location.

* + Local features: Features calculated over the results of subdivision of the image band on image segmentation or edge detection.
  + Global features: Features calculated over the entire image or just regular sub-area of an image.

* Domain-specific features: Application dependent features such as human faces, fingerprints, and conceptual features. These features are often a synthesis of low level features for a specific domain.

### CBIR

In various computer vision applications widely used is the process of retrieving desired images from a large collection on the basis of features that can be automatically extracted from the images themselves. These systems called CBIR (ContentBased Image Retrieval) have received intensive attention in the literature of image information retrieval since this area was started years ago, and consequently a broad range of techniques has been proposed. The algorithms used in these systems are commonly divided into three tasks:

* extraction,

* selection, and

* classification.

The extraction task transforms rich content of images into various content features. Feature extraction is the process of generating features to be used in the selection and classification tasks. Feature selection reduces the number of features provided to the classification task. Those features which are likely to assist in discrimination are selected and used in the classification task.

#### Color Feature Extraction

The color feature is one of the most widely used visual features in image retrieval. Images characterized by color features have many advantages:

* Robustness. The color histogram is invariant to rotation of the image on the view axis, and changes in small steps when rotated otherwise or scaled. It is also insensitive to changes in image and histogram resolution and occlusion.

* Effectiveness. There is high percentage of relevance between the query image and the extracted matching images.

* Implementation simplicity. The construction of the color histogram is a straightforward process, including scanning the image, assigning color values to the resolution of the histogram, and building the histogram using color components as indices.



* Computational simplicity. The histogram computation has O(X, Y ) complexity for images of size X × Y . The complexity for a single image match is linear, O(n), where n represents the number of different colors, or resolution of the histogram.

* Low storage requirements. The color histogram size is significantly smaller than the image itself, assuming color quantization.

### Histogram

A histogram uses a bar graph to pro le the occurrences of each gray level present in an image. The horizontal axis is the gray-level values. It begins at zero and goes to the number of gray levels (256 in this example). Each vertical bar represents the number of times the corresponding gray level occurred in the image.

Histograms also help select thresholds for object detection (an object being a house, road, or person). Objects in an image tend to have similar gray levels. Histogram Equalization Equalization causes a histogram with a mountain grouped closely together to spread out" into a flat or equalized histogram. Spreading or fattening the histogram makes the dark pixels appear darker and the light pixels appear lighter.

Histogram Equalization Algorithm

1. calculate histogram

loop over i ROWS of input image

loop over j COLS of input image

k = input\_image[i][j]

hist[k] = hist[k] + 1

end loop over j

end loop over i

1. calculate the sum of hist

loop over i gray levels

sum = sum + hist[i]

sum\_of\_hist[i] = sum

end loop over i

1. transform input image to output image

area = area of image (ROWS x COLS)

Dm = number of gray levels in output image

loop over i ROWS

loop over j COLS

k = input\_image[i][j]

out\_image[i][j] = (Dm/area) x sum\_of\_hist[k]

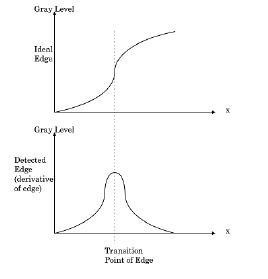
end loop over j

end loop over i

### Edge Detection

Detecting edges is a basic operation in image processing. The edges of items in an image hold much of the information in the image. The edges tell you where items are, their size, shape, and something about their texture.

The top part of shows the side view of an ideal edge. An edge is where the gray level of the image moves from an area of low values to high values or vice versa. The edge itself is at the center of this transition. An edge detector should return an image with gray levels like those shown in the lower part of Figure.



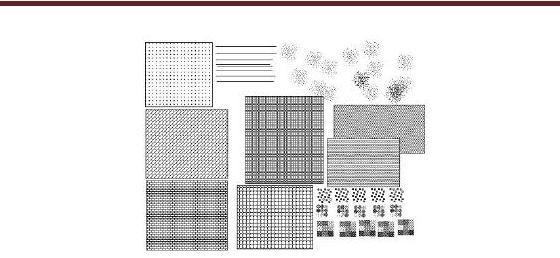
The detected edge gives a bright spot at the edge and dark areas everywhere else. Calculus fans will note the detected edge is the derivative of the edge. This means it is the slope or rate of change of the gray levels in the edge. The slope of the edge is always positive or zero and it reaches its maximum at the edge. For this reason, edge detection is often



### Color Image and Its Edge Detection Textures

One way of describing texture is \things" arranged in a \pattern." Textures are a function of things and patterns | mathematically, texture = f(thing, pattern). The thing is a grouping of pixels such as a dot or line. The pattern is the arrangement of the things such as a random or regular cluster of dots and horizontal or vertical lines. Regular patterns are usually man made while random patterns are natural. These ideas about texture all make sense, but a commonly used model of texture is lacking.

What we want is an operator that could characterize the things and pat-terns that comprise texture. Such an operator would represent a texture by a number just as gray levels represent the lightness or darkness of objects. We could then use the gray level and edge-based segmentation techniques.



#### Edge Detectors as Texture Operators

Edge detectors can be used as texture operators. This is because a textured area has many edges compared with a smooth area. Applying an edge detector to a texture produces many strong, bright edges while edge detecting a smooth area yields nothing. Smoothing the edge detector result gives a bright area that can be separated from the dark area that lacked edges

**INPUT**: Color Image

**OUTPUT**: Histogram of color image

**PROGRAMMING**: C/C++/ Java