

**NATIONAL UNIVERSITY OF SCIENCE AND TECHNOLOGY**

**School of Electrical Engineering and Computer Sciences**

**Digital Image Processing (EE-433)**

**ASSIGNMENT # 4**

**SUBMITTED TO:**

Sir Imran Malik

**SUBMITTED BY:**

Mujtaba Shahid Faizi

BSCS-5A

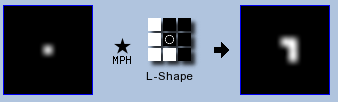
#131818

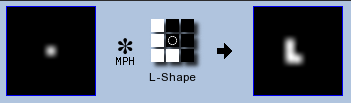
**Date of Submission:** 25-05-2018

Q1: Why do we flip the mask in Convolution. What are the properties we get by flipping it?

Mathematically speaking, a convolve operation is commutative and associative while a correlation operation is neither. So, convolve acts more like a mathematical multiply while correlate does not e.g. if kernel=F, image=G, convolve says F \* G == G \* F (commutative).

So, basically correlate method applies the kernel as is, which results in the single pixel **expanding into a rotated form,** while convolve uses an 180 degree rotated form of the kernel to **expand the same pixel into non-rotated shape**. This is why we flip the mask in convolution, so as to get the same shape as the mask. If the mask is symmetric, the it doesn’t matter if we flip it or not, because the result is same in both cases.

 No flipping

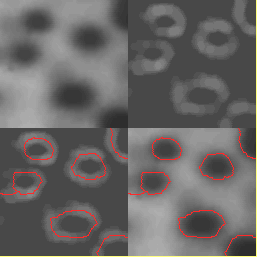
 Flipping

Here, no flipping expanded the single pixel to a rotated ‘L’ shape while flipping expanded correctly to a ‘L’ shape.

Q#2: Why don’t we directly apply watershed segmentation on the picture itself rather than on its gradient?

The gradient image is often used in the watershed transformation, because the main criterion of the segmentation is the homogeneity of the grey values of the objects present in the image. We can use the transitions between different intensity of pixels (obtained through the gradient), to identify these objects and finally, use watershed transformation to separate the objects.

e.g the following figure represents original image (top-left), gradient image (top-right), watershed of the gradient image (bottom left), and final contours (bottom right).

Here, applying the gradient has caused the catchment basins to be formed, corresponding to the homogeneous grey level regions of the image. 

Q#3: Simplified Otsu formula. One page summary in plain explanation.

Otsu’s thresholding is an algorithm for reducing a grayscale image to a binary image. Firstly, the histogram of the image is analyzed (considering the image and its histogram is bimodal) by seeing all the pixel probabilities. Using those, we use a formula for finding out the optimal threshold to use.

The formula states that we divide the pixels into 2 classes say C1, C2 by choosing a particular threshold ‘t’. Then, the inter-class variance of the 2 classes is calculated by the following function considering μ1, μ2 as class means & q1, q2 as class probabilities.

F^2(t) = q1(t) \* [1 − q1( t)] \* [μ1( t ) − μ2( t)] ^2

So, the idea is to iteratively go over all thresholds (t=1,..max intensity) and choose that particular threshold ‘t’ to classify between 2 classes, that maximizes the inter-class variance, which is the same as minimizing the intra-class variance, but less expensive. We can also find the intra-class variance by using the following function

F^2(t) = q1(t) σ1^2(t) + q2(t) σ2^2(t)

where σ1, σ2 are the variances of the 2 classes

Similarly, Otsu’s algorithm can be used for multi-thresholding by assuming more thresholds for more separating classes. This has a limitation though, that as the number of classes to separate increases, the less likely it becomes that the thresholds are selected correctly.

The range of its applications is not restricted only to the thresholding of the grayscale image, but it may also cover other cases of unsupervised classification in which a histogram of some characteristic (or feature) discriminative for classifying the objects is available.

