SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with ‘A’ Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

COURSE NAME : 19EC513 – IMAGE PROCESSING AND COMPUTER VISION

III YEAR / V SEMESTER

Unit IV- MORPHOLOGICAL IMAGE PROCESSING

Topic : Hit or miss Transformation

Hit-or-Miss Morphological Operation

In this session, you will learn how to find a given configuration or pattern in a binary image by using the Hit-or-Miss transform (also known as Hit-and-Miss transform).

This transform is also the basis of more advanced morphological operations such as thinning or pruning.

Morphological operators process images based on their shape. These operators apply one or more *structuring elements* to an input image to obtain the output image.

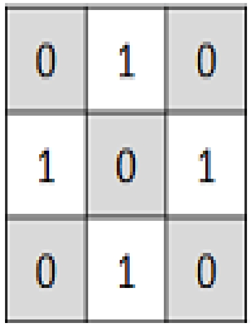
The two basic morphological operations are the *erosion* and the *dilation*. The combination of these two operations generate advanced morphological transformations such as *opening*, *closing*, or *top-hat* transform. To know more about these and other basic morphological operations refer to previous demos.

The Hit-or-Miss transformation is useful to find patterns in binary images. In particular, it finds those pixels whose neighbourhood matches the shape of a first structuring element B1
while not matching the shape of a second structuring element B2 at the same time. Mathematically, the operation applied to an image A can be expressed as follows:

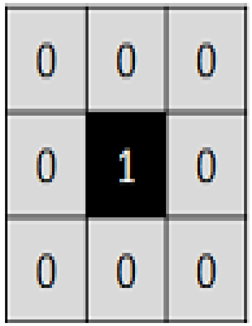
$$A \otimes B = (A \ominus B_1) \cap (A^c \ominus B_2)$$

Therefore, the hit-or-miss operation comprises three steps:
*# Erode image A
with structuring element B1. *# Erode the complement of image A (Ac) with structuring element B2. *# AND results from step 1 and step 2.

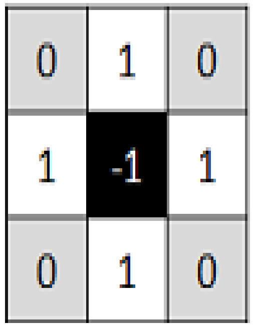
The structuring elements B1
and B2 can be combined into a single element B. Let's see an example:



0	1	0
1	0	1
0	1	0



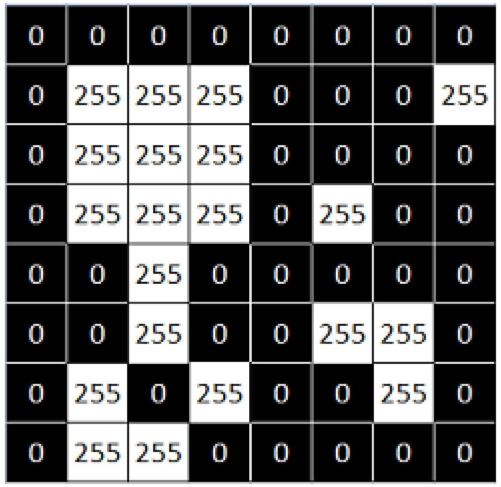
0	0	0
0	1	0
0	0	0



0	1	0
1	-1	1
0	1	0

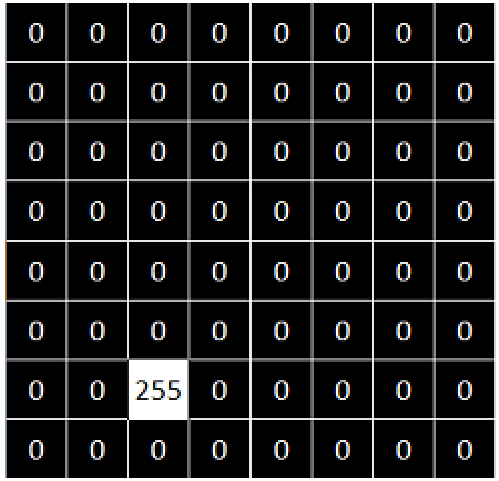
Structuring elements (kernels). Left: kernel to 'hit'. Middle: kernel to 'miss'. Right: final combined kernel

In this case, we are looking for a pattern in which the central pixel belongs to the background while the north, south, east, and west pixels belong to the foreground. The rest of pixels in the neighbourhood can be of any kind, we don't care about them. Now, let's apply this kernel to an input image:



0	0	0	0	0	0	0	0
0	255	255	255	0	0	0	255
0	255	255	255	0	0	0	0
0	255	255	255	0	255	0	0
0	0	255	0	0	0	0	0
0	0	255	0	0	255	255	0
0	255	0	255	0	0	255	0
0	255	255	0	0	0	0	0

You can see that the pattern is found in just one location within the image.



0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	255	0	0	0	0	0
0	0	0	0	0	0	0	0

THANK YOU !!!