Assignment 2.1: Distance Transform

In [1]:
```
import numpy as np
import math
```

Intensity Matrix

In [2]:
```
intensity = np.array([[0,1,1,0,1,0,0],
                      [0,1,1,1,1,1,0],
                      [0,1,1,1,1,1,0],
                      [0,1,1,1,1,1,0],
                      [0,1,1,1,1,1,0],
                      [0,1,1,1,1,0,0]])
```

Function for distance transform

In [3]:
```
def distanceTransform(intensity):
    # add boundary padding to image pixel intensity matrix
    transformedMatrix = np.zeros([len(intensity)+2, len(intensity[0])+2], dtype=int)

    for i in range(len(transformedMatrix)):
        transformedMatrix[i][0] = 0
        transformedMatrix[i][len(intensity[0])] = 0

    for j in range(len(transformedMatrix[0])):
        transformedMatrix[0][j] = 0
        transformedMatrix[len(intensity)] = 0

    for i in range(0, len(intensity)):
        for j in range(0, len(intensity[0])):
            transformedMatrix[i+1][j+1] = intensity[i][j]

    for i in range(1, len(transformedMatrix)-1):
        for j in range(1, len(transformedMatrix[0])-1):
            if intensity[i-1][j-1] == 1:
                left = transformedMatrix[i][j-1]
                right = transformedMatrix[i][j+1]
                top = transformedMatrix[i-1][j]
                down = transformedMatrix[i+1][j]
                distFromBoundary = min(left, min(right, min(top, down))) + 1
                transformedMatrix[i][j] = distFromBoundary
    return transformedMatrix
```

Transformed image after distance transform

In [4]:
```
tranformedMatrix = distanceTransform(intensity)
print("Image after distance transform:\n\n{}".format(tranformedMatrix))
```

```
Image after distance transform:

[[0 0 0 0 0 0 0 0 0]
 [0 0 1 1 0 1 0 0 0]
 [0 0 1 2 1 2 1 0 0]
 [0 0 1 2 2 2 1 0 0]
 [0 0 1 2 2 2 1 0 0]
 [0 0 1 2 2 2 1 0 0]
 [0 0 1 1 1 1 0 0 0]
 [0 0 0 0 0 0 0 0 0]]
```

In [ ]:
```

```