Introduction to Computer Vision
Optical Character Recognition

Mayukh Sattiraju
msattir@clemson.edu

ECE 631
Lab 3 Report

Thursday 5th October, 2017

1 Introduction

This lab deals with counting the number of occurrences of a particular template (in this case the letter 'e') in an image. Here we employ a Matched Spatial Filter to perform a Template Matching and fine tune the result by thinning the image and classifying it more specifically.

2 Implementation

2.1 Match Spatial Filter

The implementation of the matched spatial filter remains unchanged from lab2.

2.2 Process Flow

For lab2, the output of the MSF image was the only criteria on which classification was performed. Here, we add an additional check, before we conclude that the letter is indeed an 'e'.

The additional check, is a check on the number of branch points and end points the letter 'e' has. It can be uniquely classified to a certain degree that letters with exactly one branch point and one end point are most probably an 'e'. While other characters that don't satisfy this criteria are most likely not an 'e'.

This branch point and end point estimation is performed by first thinning the image then counting the branch and end points.

Flowchart of the process is shown in Fig. 1

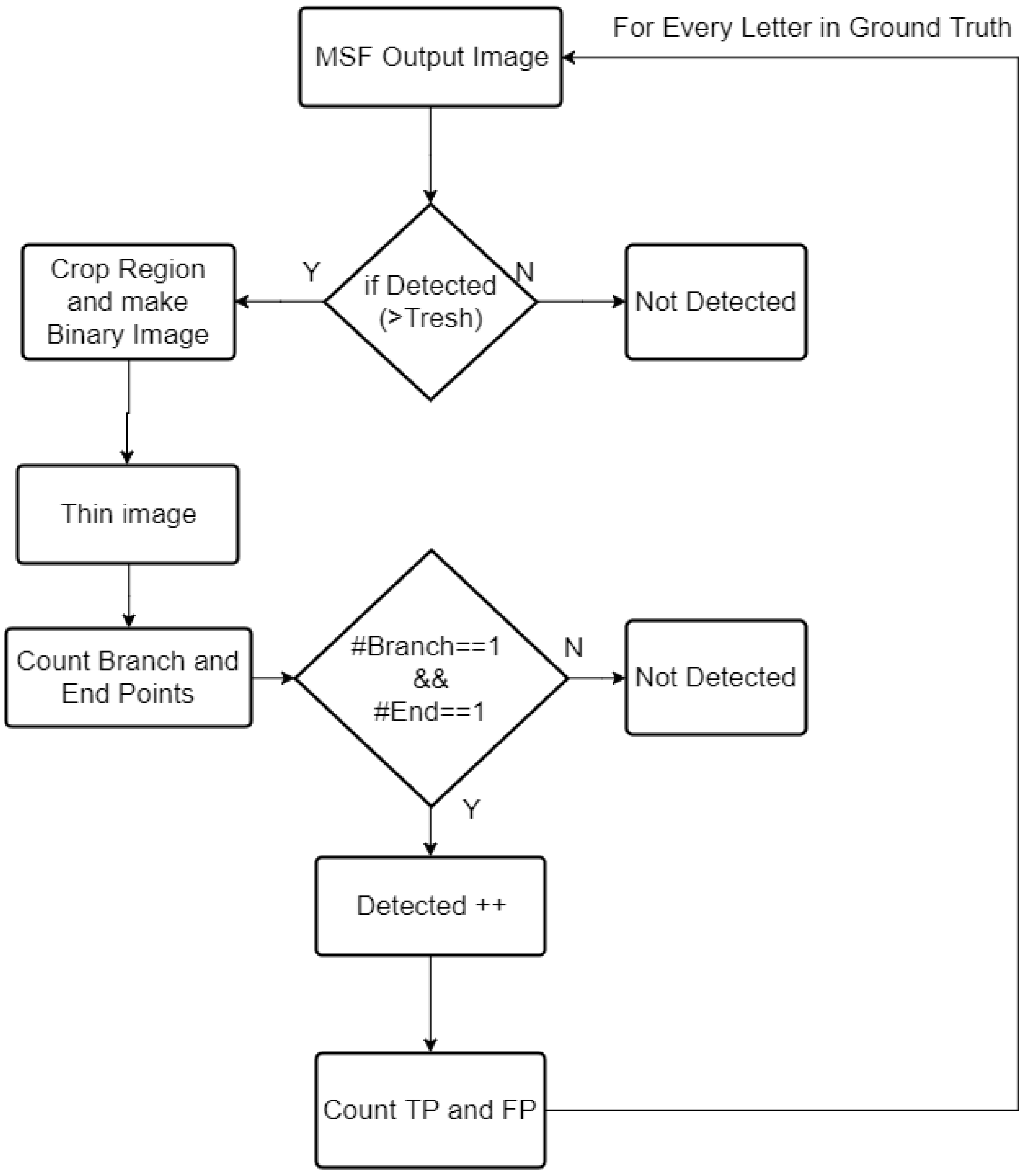


Figure 1: Algorithm Process Flow

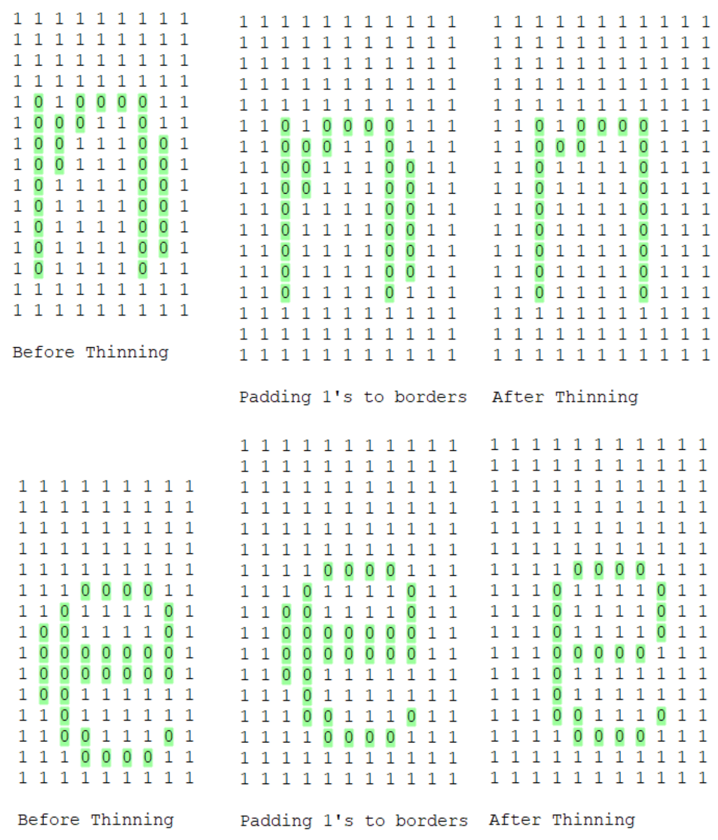
2.3 Thinning Algorithm

The thinning algorithm implemented here uses there criteria:

- Number of Edge to non Edge Transitions
- Total number of neighbours (8-way neighbours)
- Neighbours in specific locations

Once the template is thinned, we iterate over each point and check whether it is a branch point or end point. If a template has exactly one of each branch and end point, then it is classified as an 'e'.

Fig 2 shows the results of the thinning algorithm.



```
1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1
1 0 1 0 0 0 0 1 1
1 0 0 0 1 1 0 1 1
1 0 0 1 1 1 0 0 1
1 0 0 1 1 1 0 0 1
1 0 1 1 1 1 0 0 1
1 0 1 1 1 1 0 0 1
1 0 1 1 1 1 0 0 1
1 0 1 1 1 1 0 0 1
1 0 1 1 1 1 0 1 1
1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1

Before Thinning
```

```
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 0 1 0 0 0 0 1 1 1
1 1 0 0 0 1 1 0 1 1 1
1 1 0 0 1 1 1 0 0 1 1
1 1 0 0 1 1 1 0 0 1 1
1 1 0 1 1 1 1 0 0 1 1
1 1 0 1 1 1 1 0 0 1 1
1 1 0 1 1 1 1 0 0 1 1
1 1 0 1 1 1 1 0 0 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1

Padding 1's to borders
```

```
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 0 1 0 0 0 0 1 1 1
1 1 0 0 0 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1

After Thinning
```

```
1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1
1 1 1 0 0 0 0 1 1
1 1 0 1 1 1 1 0 1
1 0 0 1 1 1 1 0 1
1 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 1
1 0 0 1 1 1 1 1 1
1 1 0 1 1 1 1 1 1
1 1 0 0 1 1 1 0 1
1 1 1 0 0 0 0 1 1
1 1 1 1 1 1 1 1 1

Before Thinning
```

```
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 0 0 0 0 1 1 1
1 1 1 0 1 1 1 1 0 1 1
1 1 0 0 1 1 1 1 0 1 1
1 1 0 0 0 0 0 0 0 1 1
1 1 0 0 0 0 0 0 0 1 1
1 1 0 0 1 1 1 1 1 1 1
1 1 1 0 1 1 1 1 1 1 1
1 1 1 0 0 1 1 1 0 1 1
1 1 1 1 0 0 0 0 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1

Padding 1's to borders
```

```
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 0 0 0 0 1 1 1
1 1 1 0 1 1 1 1 0 1 1
1 1 1 0 1 1 1 1 0 1 1
1 1 1 0 1 1 1 1 0 1 1
1 1 1 0 0 0 0 0 1 1 1
1 1 1 0 1 1 1 1 1 1 1
1 1 1 0 1 1 1 1 1 1 1
1 1 1 0 0 1 1 1 0 1 1
1 1 1 1 0 0 0 0 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1

After Thinning
```

Figure 2: Thinning Result Matrices

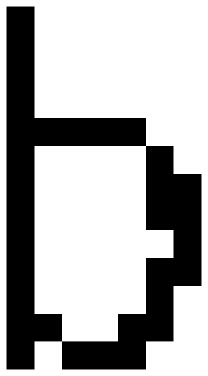


Figure 3: Before Thinning

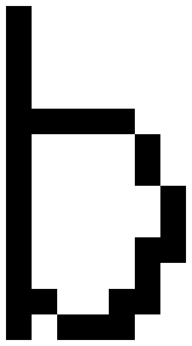


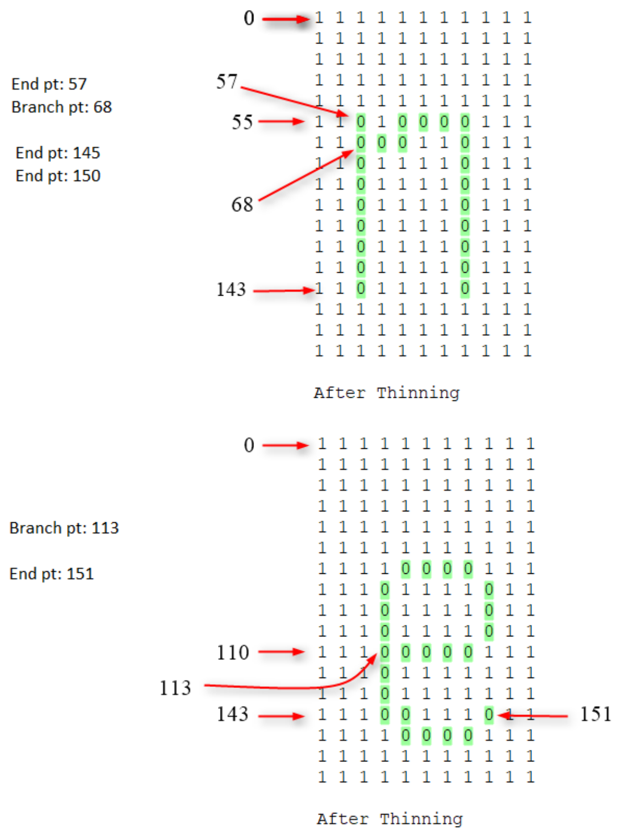
Figure 4: After Thinning

2.4 Branch Point and End Point Calculation

After thinning the image, we iterate through all points on the letter, and determine the number of transitions each point has. If it has exactly 1 transition, its a end point. If it has more than two transitions, it is a branch point.

To detect an 'e' one can look for exactly one branch point and one end point.

Here is an example figure,



```
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 0 1 0 0 0 0 1 1 1
1 1 0 0 0 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 0 1 1 1 1 0 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1

After Thinning
```

```
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 0 0 0 0 1 1 1
1 1 1 0 1 1 1 1 0 1 1
1 1 1 0 1 1 1 1 0 1 1
1 1 1 0 1 1 1 1 0 1 1
1 1 1 0 0 0 0 0 1 1 1
1 1 1 0 1 1 1 1 1 1 1
1 1 1 0 1 1 1 1 1 1 1
1 1 1 0 0 1 1 1 0 1 1
1 1 1 1 0 0 0 0 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1

After Thinning
```

Figure 5: Branch and End Point calculation

2.5 Tresholding

The process of Tresholding and generating the ROC curve also remain unchanged.

Here is the ROC curve generated for Threshold values from 0-255

The FPR for the MSF+Thinning algorithm, starts only at 0.2 and reaches 0 fairly quickly.

The highest TP achieved is 136/151 or 90.06%. Minimum FP achieved is 0.

A number of 'e' go into true negatives, this is because of stray pixels from adjacent letters and sometimes, the input 'e' is broken, thus yeilding more than 1 branch point.

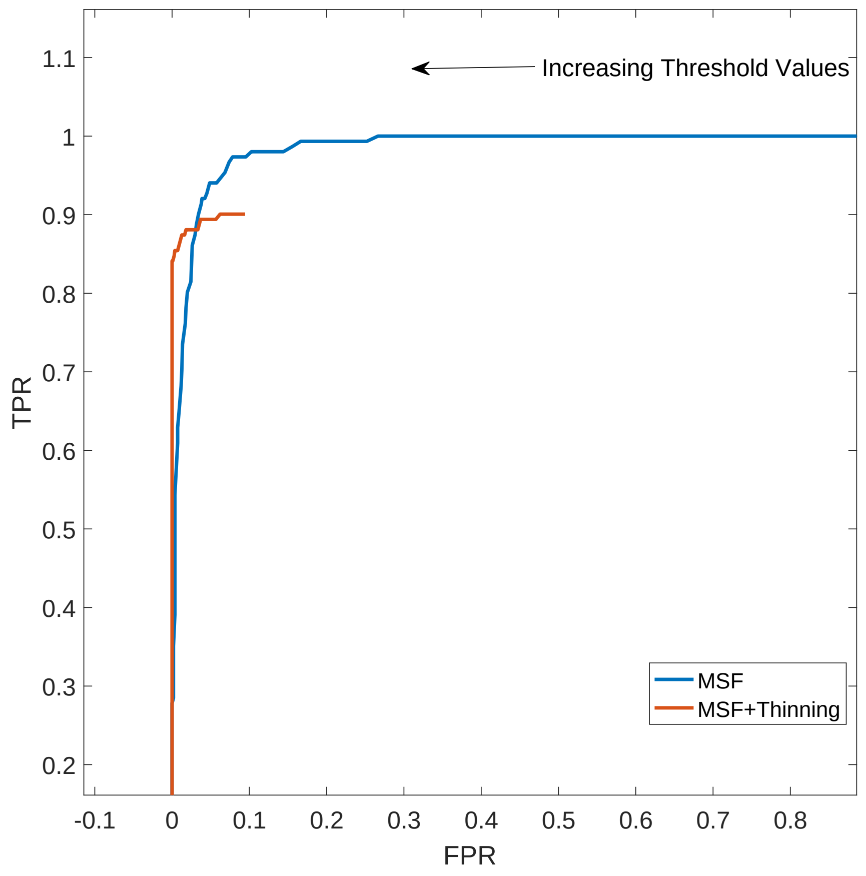


Figure 6: ROC Curve for Threshold range 0-255 Compared with just MSF Tresholding

3 Results

To chose the right threshold value, we ideally chose a point with a low enough False Positive Rate and a high enough True Positive Rate. This is generally found near the 'Knee'.

Here is a comparison of the True Positives and False Positives around the 'Knee' with that of lab2.

From this result, choosing a Threshold of **206** seems appropriate. It has a high enough TPR of 84.77% (TP:128) and a low enough FPR of 0.27% (FP:3). Thresholds above 208 have a lower TPR value, and thresholds below have a higher FPR value.

As compared with the previous results, for data at around the same threshold value, we see a significant plunge, in FPR to as low as 0.2% or virtually non-existent. Although, this has compromised on the TPR rate as well, a decline of about 8% is noted here.

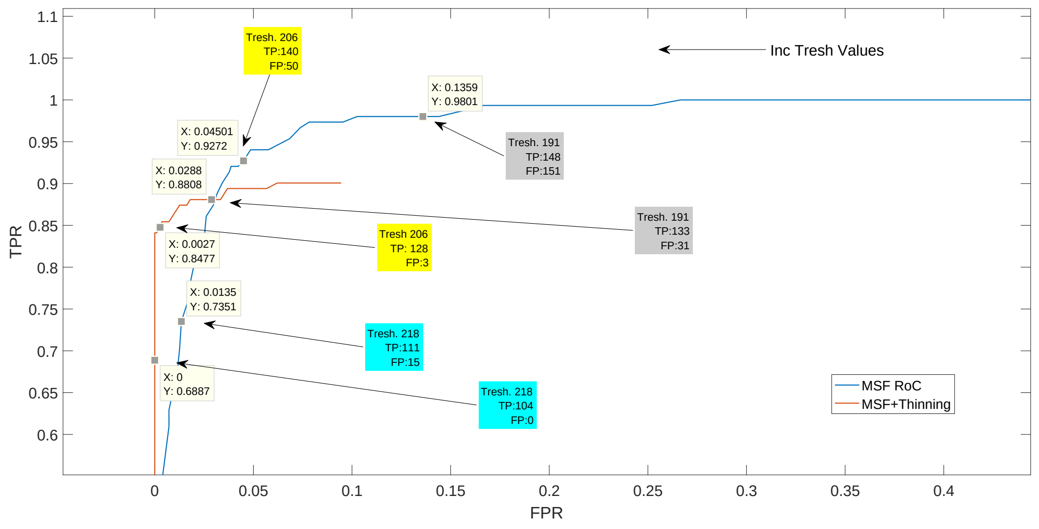


Figure 7: TP and FP values near the 'Knee'

4 Code

Listing 1: Entire Code Lab3

```
/* Optical Character Recognition
     ** This program reads parenthood.ppm, parenthood_gt.txt, smoothed1.ppm (MSF
        Image from Lab2) and a command line Treshold Value
     **
     ** The program also demonstrates how to time a piece of code.
     **
     ** Mayukh Sattiraju (10/5/17)
     */

#include <stdio.h>
#include <stdlib.h>

#define TOTAL_DATA  1262


void calc_c(int c, int n, int arr[8])
{
   arr[0]=c-1;
   arr[1]=c-n-1;
   arr[2]=c-n;
   arr[3]=c-n+1;
   arr[4]=c+1;
   arr[5]=c+n+1;
```