IC250 Laboratory Assignment Template

Aditya Nigam

August 12, 2016

IC250 Programming and Data Structure Practicum

Lab Assignment No. 01, Date : {16,17,18} Aug , 2016

Introduction/Problem Context

This laboratory assignment focuses on the use of histogram and cumulative histogram data structure in the context of solving iterative thresholding problem. Both histogram as well as cumulative histogram are representation of the frequency of elements and stored as a 1D array.

It assumes that you are familiar with static and dynamic data representation and C language features related to them. You may refer to the references given, if you required to refresh these topics.

Problem : Optimum Threshold Selection

Any image can be seen as a 2D array. Any element in such an array is an integer within a range of $\{0\ to\ 255\}$ which is also called as its gray value. Thresholding image data (using th as threshold) divide all elements (*i.e.* pixel) into two clusters/groups:

1. Pixels with their gray value more than th
2. Pixels with their gray value less than or equal to th.

Practically there can be 256 different possible thresholds (as $th \in \{0\ to\ 255\}$). The optimum threshold is the one that can generate two maximally different clusters that has to be evaluated efficiently.

Task Description

You are required to write a C program which take a text file (*.txt*) file as an input containing the image data (*tab* separated). The row and column information will also be written into the file. Parse the file and store data into an 2D dense array (dynamically allocated). Now iteratively threshold the 2D array and for each threshold:

1. Generate two clusters A and B.
2. Compute number of points n_A^p and n_B^p, mean μ_A and μ_B and standard deviation σ_A and σ_B.
3. Compute the ratio of points in group A and B, $n_{ratio} = \frac{n_A^p}{n_B^p}$
4. Find out the discriminative index $d' = \frac{|\mu_A - \mu_B|}{\sqrt{\sigma_A^2 + \sigma_B^2}}$.

Input Data and Format

Input file is a normal text file containing image pixels gray value data. First line contain *row*, *column* information followed by the data, as shown below:

```
198 200
71 43 123 12  ...............
34 34 22 245 ...............
...........................................
...........................................
```

Computation Involved

The problem is divided into three parts :

Part A : Read the input text file and parse the file. Store the data into a dynamic allocation 2D array.

Part B : Write a C code that can threshold iteratively 256 time and for each threshold compute n_p, d'. Create a file *result.dat* that have the value of n_p, d' for each threshold th, as follows:

```
th d‘n_p
1 2.1123 1.234
2 2.1123 1.2342
3 2.1123 1.2343
4 2.1123 1.2344
5 2.1123 1.2345
```

Now plot three graphs using gnuplot :

1. th Vs d', save it as *a.png*.
2. th Vs n_p, save it as *b.png*.
3. d' Vs n_p, save it as *c.png*.

Finally, compute the optimum threshold and justify your answer.

Part C : Repeat the above exercise but efficiently using cumulative histogram based solution. Compare the timing statistics of iterative and efficient version as well as the correctness of this solution.

Pseudocode

The basic Pseudocode looks like :

1. FUNCTION OPT_Thresh()
2. Parse the file and store data into an 2D dense array (dynamically allocated).
3. Now iteratively threshold the 2D array and for each threshold:

(a) Generate two clusters A and B. //(Hint: you have to use linked list to store A and B)

(b) Compute number of points n_A^p and n_B^p, mean μ_A and μ_B and standard deviation σ_A and σ_B.//(These variables can be reused)

(c) Compute the ratio of points in group A and B, $n_{ratio} = \frac{n_A^p}{n_B^p}$

(d) Find out the discriminative index $d' = \frac{|\mu_A - \mu_B|}{\sqrt{\sigma_A^2 + \sigma_B^2}}$.

4. Save 3-tuple tab separated data $< th\ d'\ n_p >$ for each threshold in *result.dat* file.

5. Draw the required plots using *gnuplot*.

Expected Output for Correct and Incorrect Inputs

Three data files are given along with this assignment in the above mentioned format *viz.* 1.*txt* (small size file), 2.*txt* (medium size file), 3.*txt* (very big size file). For any given input file theses things are required to be done:

- Above mentioned three graphs.
- Optimum Threshold *th* and its justification in *res.txt* file.
- Time taken in part A. (save in *res.txt*)
- Time taken in part B. (save in *res.txt*)
- Time taken in part C. (save in *res.txt*)
- Also show that part B and C generate same *result.txt* file.

Sample Output

For Part B

```
$ ./iterative 1.txt

Reading the input data .........
Data read in 0.22111  ms

Performing iterative thesholding ...........
Thresholding done in 1.2342 ms

result.dat is generated at <location>.

Graphs are generated at <location>.

Final res.txt is generated at <location>
```

For Part C

```
$ ./efficient 1.txt

Reading the input data .........
Data read in 0.22111  ms

Performing thesholding using cumulative histogram ...........
Thresholding done in 1.2342 ms

result.dat is generated at <location>.

Graphs are generated at <location>.

Final res.txt is generated at <location>
```

GNUPLOT

Read the gnuplot documentation. You can go through the following links
http://people.duke.edu/~hpgavin/gnuplot.html
http://www.gnuplot.info/docs/tutorial.pdf

In this section you will see how to call gnuplot from a C program. Gnuplot is used for plotting graphs. The data to be plotted is written in a file. For example to plot the impedance as a function of frequency you need to create a data file with two columns separated by a space. The first column contains frequency and second the impedance values.

The commands for gnuplot are stored in a character array. The data generated is written into a file which gnuplot will access and plot. The code segment is given below (taken from http://stackoverflow.com/questions/3521209/making-c-code-plot-a-graph-automatically).

```
char * commandsForGnuplot1[] = {"set term x11 1","set title \"Impedance vs
    frequency\"", "set key outside", "plot 'data.temp' with line"};


char * commandsForGnuplot[] = {"set term x11 0", "set title \"Voltage waveforms\"",
    "set key outside", "plot for [col=2:4] 'data.temp' using 0:col with line"};


FILE * temp = fopen("data.temp", "w");
```

The commands for gnuplot are written in double quotes with each command separated by a comma. The "set term" command is used to open a new figure window, plot command for plotting the data in data.temp file and the with command to indicate that points should be connected by a line. Notice that in the second set of commands *for* is used inside plot - this is because the file has multiple columns and the *for* command will plot multiple curves on the same graph. The command *using* fixes the variable on the $x-$axis. The "set key outside" command puts the legend of the graph outside the plot area.

The following code segment writes the data into the file and the second for loop starts gnuplot and generates the graphs. Remember to close the data file and gnuplot stream using fclose and pclose, respectively.

```
FILE * gnuplotPipe = popen ("gnuplot -persistent", "w");
   for (ind=0;ind<i;ind++)
     fprintf(temp, "%lf %lf %lf %lf\n", time[ind],
          cimag(VR[ind]),cimag(VL[ind]),cimag(VC[ind]) );
   for (ind=0;ind<4;ind++)
     fprintf(gnuplotPipe, "%s \n", commandsForGnuplot[ind]);
   fclose(temp);
   pclose(gnuplotPipe);
```