COMP ENG 3SK3 Project Phase 2
Liuyin Shi 1427387

This is part is for calculate the feature of template for matching. I wrote a function named feature(). It has 3 arguments:

Feature(picture, start,size)

The *pic* is the template matrix comes from reading, start is the start point of template matching (top left point of the matrix), and *size* is the size of a size x size matrix for calculating features. The return values are:

[mean, variance, gradient magnitude, gradient angle]

1. The mean of the patch comes from the table L1.we use builtin function sum to find the sum and divide it by square of size
2. The variance can be measured from the E[(X-mean)^2]. For calculating it, we can expand the formula inside:

   (X-mean)^2=X^2-2*X*mean+mean^2

   The X^2 can be obtained from the table L2, and mean comes from L1. the variance can be obtained from the table L2 and and sum the each variance element multiplied by 1/(size^2).

3. The gradient comes from the table LX and LY. At first we initialized an array of EP, as the EP operator for EPX and EPY are the same. The only difference is the order of multiplication. As the size is always odd for convenience, the size it should be size x 1 and the absolute value of max/ min of it is (size-1)/2. Then we can multiply the elements in L1 with the operators to get LX and LY . when we have LX and LY, we can sum up the tables and get gradient.

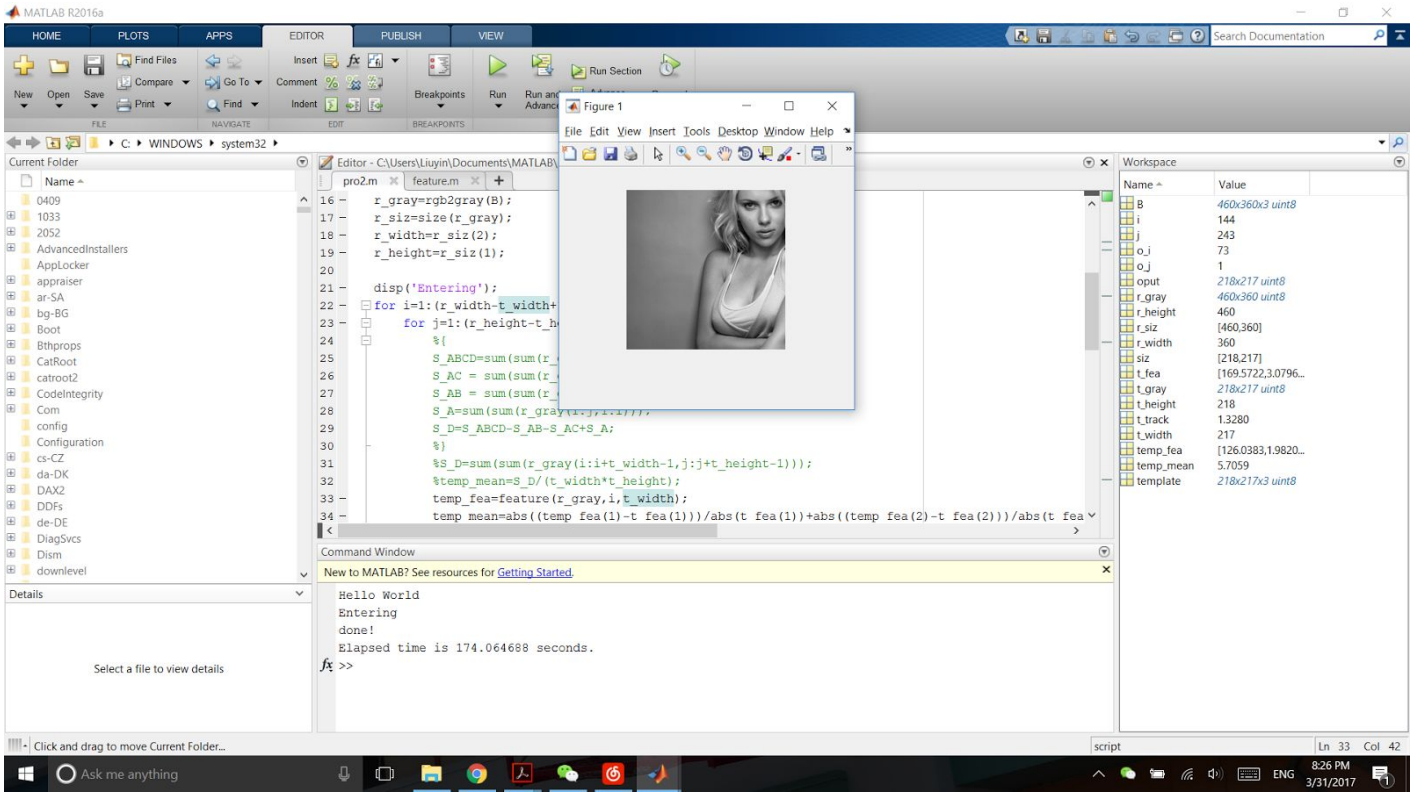
For obtaining the feature and get the most likely patch, we need all the attributes. When we have the mean, variance and gradient, we compare it to the features of the template. We sum up the distance/template value to see which patch has the smallest ratio.

abs(patch_mean-template_mean)/template_mean+abs(patch_var-temp_var)/temp_var+abs(patch_gra-temp_gra)/temp_gra

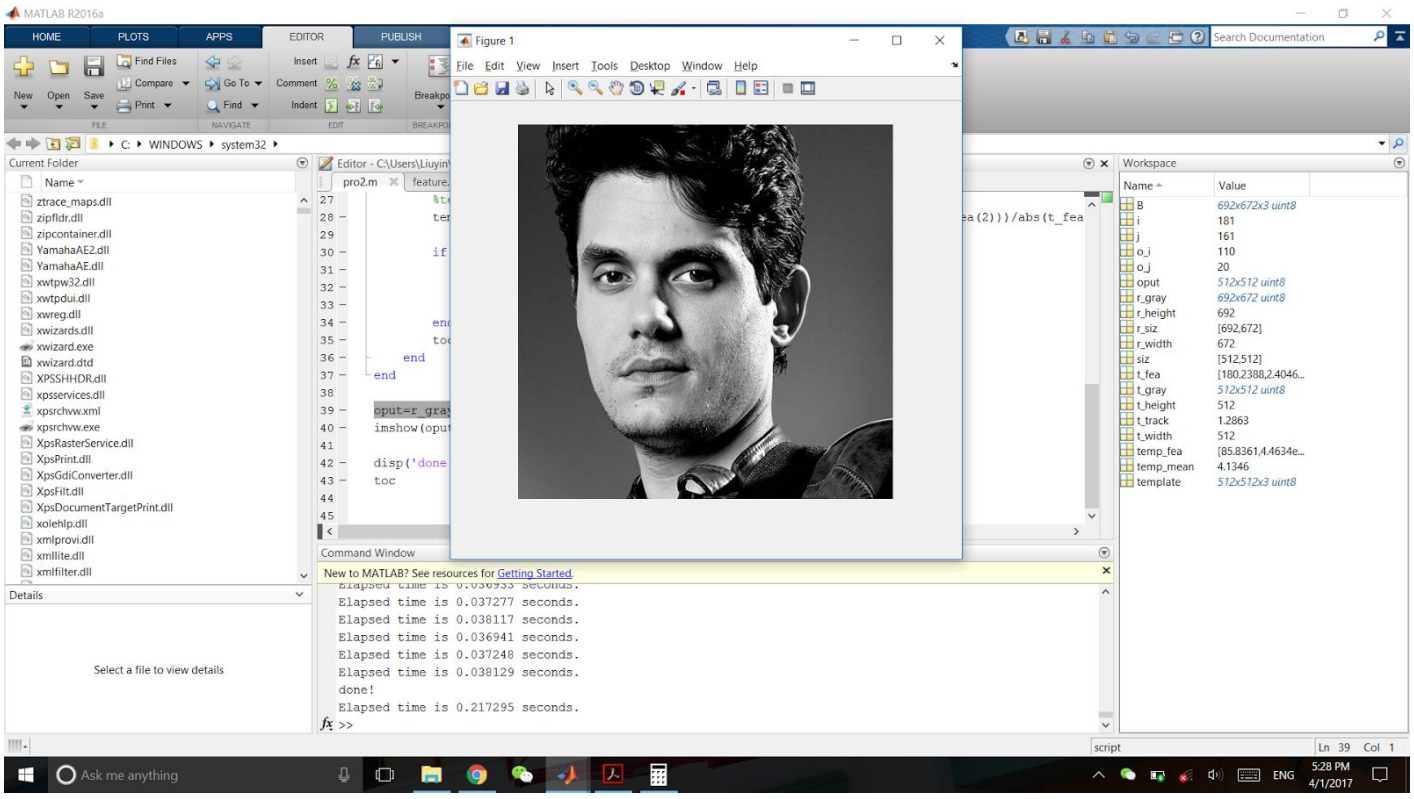
Complexity: as we have table size nxn, each table needs n^2 times of operations. As for the nested for loop, we have n^2 times, the overall would be n^4 times.

Question:
As we are using 32-bit unsigned integer for storage, we have the limit for some size so that there is no overflow when summing all the elements in the table. We have 256 as the max of each element and this takes 8 bit. After that we have 24 bit available. Assume we have template size of n x n. n/2*n*n would have a max of 2^24. n^3=2^25. So the max template should have 256*256 pixels for safety concern. For the ratio of the width and height. Neither width and height can exceed 1024 as the reference's size is 1024x1024

Reference Tests



LT as template, down as reference.



Lenna as template, test2 as reference