**ADVANCED ALGORITHMS**

**Final Project**

**Image Comparison**

**By**

**Sukesh Reddy (sg121)**

**Shashank Reddy (sa123)**

**Vandana Rao(vk34)**

**Rishi Reddy (rc81)**

**Professor**

**Dr. Zhong-Hui Duan**

**Description:**

We live in an age where images are a daily part of our lives, there are wide ranges of comparing images, someone might want to compare both images and check to see whether the shapes of the object are similar or not, while others might want to check for subtle changes. Some people might just want to know simply whether the two images are equal (1) or unequal (0).

The Algorithm we are trying to implement is a naïve similarity algorithm where the inner regions (image is divided into smaller blocks) of the image A is compared with its counterpart regions of image B and we get the distance between the two regions. The more dissimilar the images are, the more the distance between the images. We use **Euclidean distance** to calculate the distance between the two images based on the RGB values of two images. The final images are shown on a JFrame.

The Second part of the project is whenever we click on some portion (pixel) of the **Image-A**, we compare the portion’s clicked block RGB values with all of the remaining portions (blocks) of the **Image-B** and we highlight the selected portion (block) of the **Image-B** with the red colour that matches with the RGB values of the **Image-A.**

**Procedure**

**Part-1:**

* Take the **Image-A** as the first input image.
* For images of irregular sizes we need to rescale them into a single size. So we are rescaling them into N\*N pixels (say 500\*500) image and then we apply the image comparison technique.
* For comparing the image, we divide the image into several blocks. We are considering that each block consists of 25\*25 pixels and since we have rescaled to N\* N pixels image, we will have m\*m number of blocks. For example consider 500\*500 pixels. There will be 20\*20 (400 blocks) since we are considering each block consists of 25\*25pixels.
* We then take the average of all the pixels of the each region (block). Such as for each region we get all the reds, blues, green values and summing up all the red’s, green’s and blue’s separately. Then we take the average of R,G and B values by dividing with the total number of pixels of that region (In our case 25\*25 i.e 625 pixels since each block consists of 25\*25 pixels).
* r1+r2+r3+…r25/total number of pixels.

b1+b2+b3+…b25/total number of pixels.

g1+g2+g3+…g25/total number of pixels.

* These are the pixel values of each of the N regions (blocks) .so each region has a RGB value which is the average of all the pixels of its particular region.
* We then take the second image as the input and therefore we repeat the same process as mentioned above for the image A. So Image B also has the RGB values (average of all the pixels of particular region) for all the regions.
* Then we compare each 25\*25 pixel region in **Image A** with all the 25\*25 pixel regions in **Image B** based on the **Euclidean** distance i.e we are comparing 1:N (1 block in image A with N blocks in image B) and finding out how many blocks are similar in image B to particular block in image A. Similarly we do it for all the blocks in Image A. Then we find the average blocks matched in image B for each block in Image A.
* The blocks are said to be similar if the distance between the 2 blocks is Zero. The distance is nothing but the **Euclidean distance**. If we consider the threshold (distance) to be <50, then the two blocks are said to be similar if the Euclidean distance between two blocks is <50.
* The distance is calculated by **√(r1-r2)2+(g1-g2)2+(b1-b2)2** between two regions where r1,g1,b1 belongs to Image A region and r2,g2,b2 belongs to Image B region.

**Part-2:**

* The second part of the project is where the input is an **Image A**.
* When we click on any portion of the image A, this pixel region of the image A would be compared with all the regions of the Image B.
* The comparison is done based on the **Euclidean distance** of the regions RGB values.
* The pixel portions in Image B with the minimum scaling threshold are selected and then these pixel portions are coloured with red.

**Test Cases:**

**Step1:**

**Input:** Image A **Target:** Image B

**Step2:**

Output  

Indicates the area of mouse click

The Image covered in Red indicates the matched portion with the actual selected area.

From the above Test Case-1, we can see that Image A is the input where the user clicks and Image B is the output where some portion of the image will be highlighted with Red colour. In the step 2, on the Image A we can see the Black colour dot. This dot indicates that user has clicked in that portion. Output image indicates the highlighted regions based on the similarity with the region where the user has clicked.

**Process:** When the user clicks on the portion of Image A, program calculates the average RGB value of that block and compares with all other blocks of Image B and highlights the Image B portions which are matched.

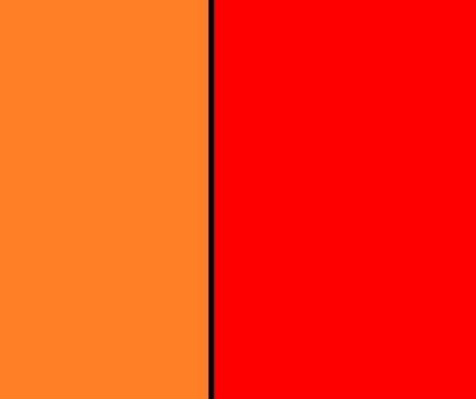
**Testcase2:**

**Step 1:**

**Input:** Image A **Target:** Image B 

**Step 2:**

Output

The Image covered in Red indicates the matched portion with the actual selected area.

Indicates the area of mouse click

From the above Test Case-2, we can see that Image A is the input where the user clicks and Image B is the output where some portion of the image will be highlighted with Red colour. In the step 2, on the Image A we can see the Red colour dot. This dot indicates that user has clicked in that portion. Output image indicates the highlighted regions based on the similarity with the region where the user has clicked.

**Process:** When the user clicks on the portion of Image A, program calculates the average RGB value of that block and compares with all other blocks of Image B and highlights the Image B portions which are matched.

**Contributions:**

**Image comparison: -** Sukesh Reddy and Shashank Reddy

**Report and Documentation :** Shashank Reddy and Sukesh Reddy

**Graph:**

The graph shows the relation between scaling factor and number of highlighted blocks.

The X-axis represents the Scaling Factor.

The Y-axis represents the number of blocks that are matched.

The above graph is plotted between Scaling factor and the number of blocks matched. Scaling factor is nothing but the Euclidean distance between the Image-A block RGB values where the user clicks and the Image B blocks RGB values. If the Euclidean distance between two blocks is zero, then they are said to be perfectly matched. But there may be a case where Euclidean distance is minimum but not considered. So we are considering the Scaling factor. We plotted the graph to show how the number of blocks matched varies with the increase in Scaling factor value.

From the above image, we can see that as the Scaling factor value increases, number of blocks matched also increases as expected.

**TIME COMPLEXITIES:**

The first part of our project is where we compare each of the pixel regions to all of the pixel regions in image B since it compares each of the pixel region in image A to ‘n’ pixel regions in image B and as there are n regions in Image A, the complexity is O(n2).

The second part of the project which is highlighting the matched region .On clicking the mouse pointer on a specified region on the image, the part of the image which matched with the location gets highlighted with the red colour.

Since only a single pixel portion is matched with all the pixel regions of the image B, the Time Complexity is O(n).

**References:**

1. <http://www.oracle.com/technetwork/java/current-142188.html>
2. <http://www.cis.pku.edu.cn/faculty/vision/wangliwei/pdf/IMED.pdf>
3. <http://www.lac.inpe.br/JIPCookbook/6050-howto-compareimages.jsp>
4. Image Recognition and Classification: Algorithms, Systems, and Applications, edited by Bahram Javidi, Marcel Dekker, ISBN 0-8247-0783-4, 2002.