Baylor University

Department

Electrical and Computer Engineering

ELC 5396

Image Quality Comparison in Select Regions

Sarah McCarty

December 8, 2016

Contents

List of Figures

1 Introduction

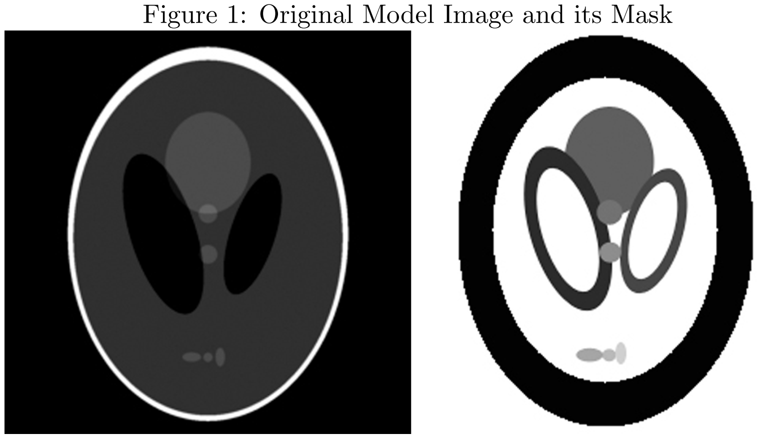
When comparing methods of preprocessing or reconstructing images from data sets in medical imaging, a large number of images are generated and their quality must be compared. While it is possible to send a human subject to compare image after image by eye, this is overwhelming, subjective, and inefficient. Comparing images pixel by pixel using a computer is possible, but the majority of the image space will be empty and insignificant. The goal of this project is to create a program that will take in a large number of images and compare them each to a model image in specified areas. Areas will be specified using a masking image that marks each region for comparison and the program will return an error vector that contains the magnitude of the difference from the model for each image in each region.

2 Implementation

2.1 Mask Prep

The first step in the image comparison process is the creation and prep of the masking image. For the example worked through in this report, the model chosen is a generic head phantom used in particle imaging applications. The mask was created in a photo processing software and consists of simple shapes of various gray tones surrounding areas of interest. In this example, the entirety of smaller shapes were selected as well as the exterior edges of the larger regions. Because the interior of the larger ovals are uniform, comparison in these regions would be superfluous. This model image is a good example of one that has a lot of uniform space both inside the phantom and in the background. Comparing result images with the model in its entirety would consume a lot of computational power to compare large, non-critical portions of an image. The model image and its associated mask image are shown below.

Figure 1: Original Model Image and its Mask



After the mask image was created, it was fed to a function called prepMaski which takes in the mask as well as an estimate of the number of regions in the mask. The function cycles through the image pixel by pixel ensuring that it is not the white of the background. If not, its value is compared to that of the previous pixel in the row. If the result of their subtraction is less than five, it is assumed that this pixel is not at a shape edge and should be assigned to a region. The value of this pixel is compared to a collection of ”region standards” and if it is not within fifteen of an existing region standard is set as a new region standard. If it is within fifteen of an existing region standard, the x and y coordinates of this pixel are recorded in the regions matrix corresponding to the region it matched. This is continued until each pixel has been assigned to a region. The function then returns a three dimensional matrix called regions that contains the x and y coordinates of the points in each region as well as a rcount vector that contains the number of pixels in each region. The example masking image had nine areas of interest so prepMaski returned a three dimensional matrix that contained the x and y coordinates of each point in nine regions. Though this function is relatively slow as a n^2 operation, it only needs to be run once per mask image. This preparation sets up Matlab to perform the comparison of the images.

2.2 Image Registration

Because it cannot be guaranteed that a particular result image will be aligned with the model image in terms of rotation, scaling, or translation, image registration is required. Image registration is accomplished for this project using the Matlab image registration toolbox. Depending on the individual image, the registration may need to be run in a different registration ”mode”. The registration was tested for result images that were rotated, scaled, translated, and a combination of the three.

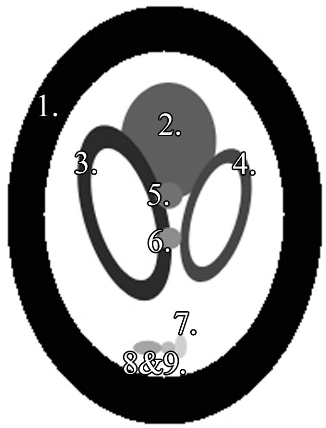
2.3 Image Comparison

Once the mask preparation and image registration steps have been completed, the program is in possession of the locations of important regions, a model image, and a matched comparison image. With this preparation in place, the actual comparison of images is a relatively simple process. The model image, result image, regions matrix, and count vector are passed to imcomp which performs the computation. This function cycles through each region, looping through each point in the selected region. Because the three dimensional region matrix contains the number of points in the largest region, the program first checks to see if both the row and column locations are zero and if not, performs the error computation. For each point, the value of the model image is subtracted from that of the comparison image and the norm of the result taken. The value of these norms are added together for each region and then scaled according to the number of points in the region. This allows us to, at a glance, see the severity of the error in each region. After the Image Comparison step is completed, we have the desired results.

3 Results

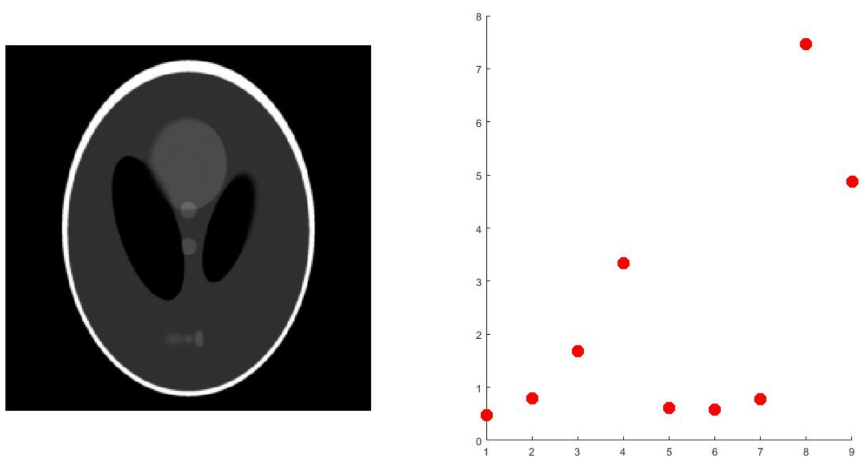
The program returns a vector with a length equal to the number of regions in the mask image, containing the scaled error in each region. This allows location of errors and gives an idea of their severity. By comparing these numbers, it can easily be determined which image is the closest to the model overall or in a specific region. For this example, the resulting errors can be compared to their associated regions as Matlab assigned them. These regions are assigned from top to bottom, left to right and for this example, are assigned as in the image below.

Figure 2: Matlab's Numbering of the Nine Mask Regions



A number of test images were created with known errors in order to test the program's functionality. The first test image had a blur applied to the area between regions two and three, the edge of region four, and the entire regions eight and nine. The result image along with a graph of the associated error is shown below.

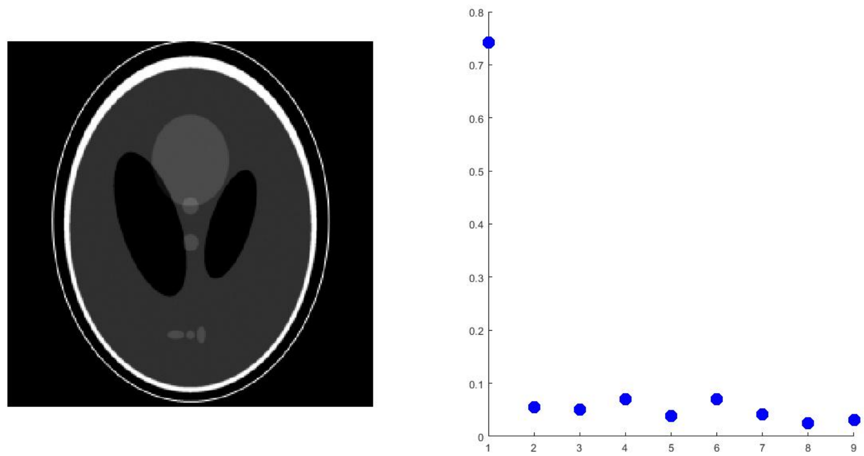
Figure 3: Selected Blurring Test



It can be seen from the associated plot that the error values correspond with the regions in which the blurring is taking place. The greatest blur affects region eight which is blurred almost to the point of non-existence.

Another interesting test image was created by adding a ring surround region one to simulate the residuals that sometimes appear when performing image reconstruction. The results show that the error affects only region one as expected.

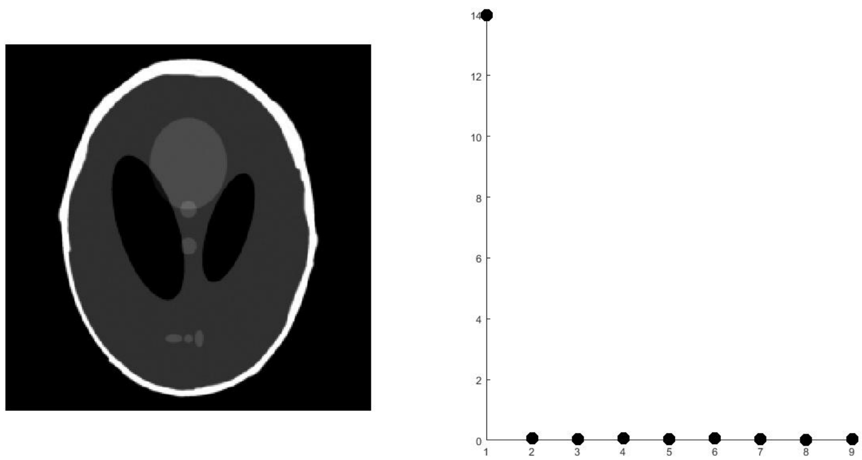
Figure 4: Ring Test



Tests were also run for images containing universal blur, region additives, "dimpling" regions, and removing portions of a region. For each, the program acurately picked out the regions affected and gave a severity value to the imposed error.

Next, an additive result image was taken and rotated, translated, and scaled to compare the error with registration to the same result image without. The original result image and its error is compared with the result image after various methods of image registration and their errors.

Figure 5: Orginal Additive Test Image



As can be seen from the above graph, the error should be located only in region one. Additional white space has been added to the image in the region one area. This error can be compared with that of the same image after going through an alteration in rotation angle, scaling, translation, and a combination of the three.

Figure 6: Rotated Image Test

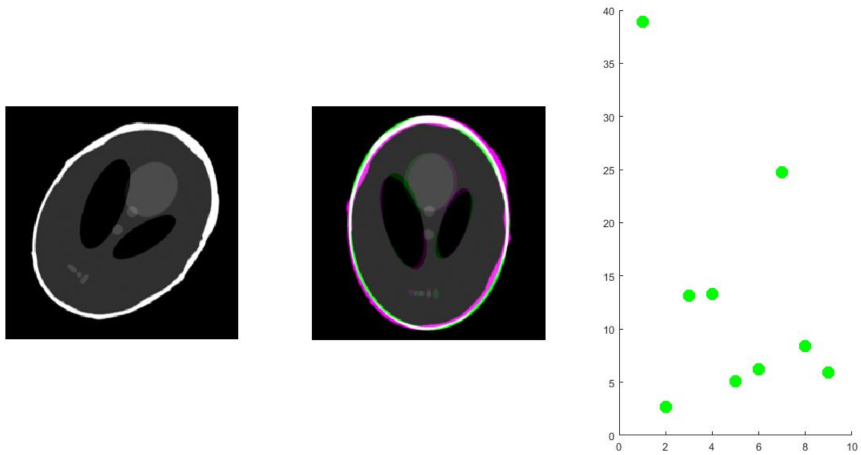


Figure 7: Scaled Image Test

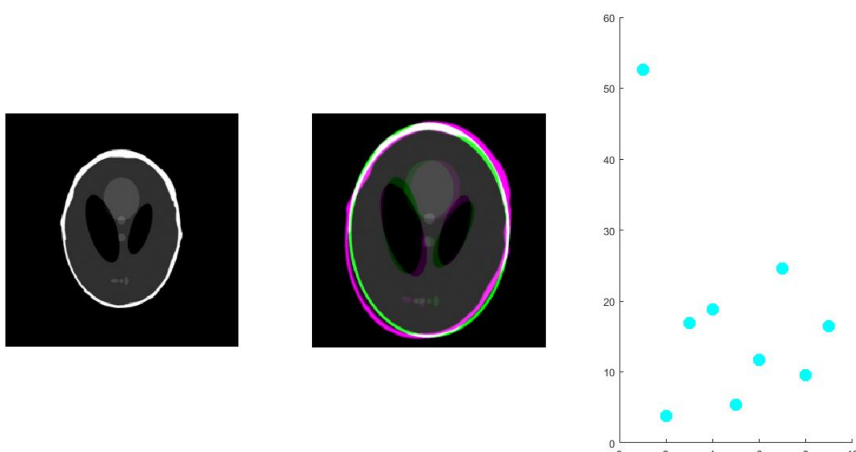


Figure 8: Translated Image Test

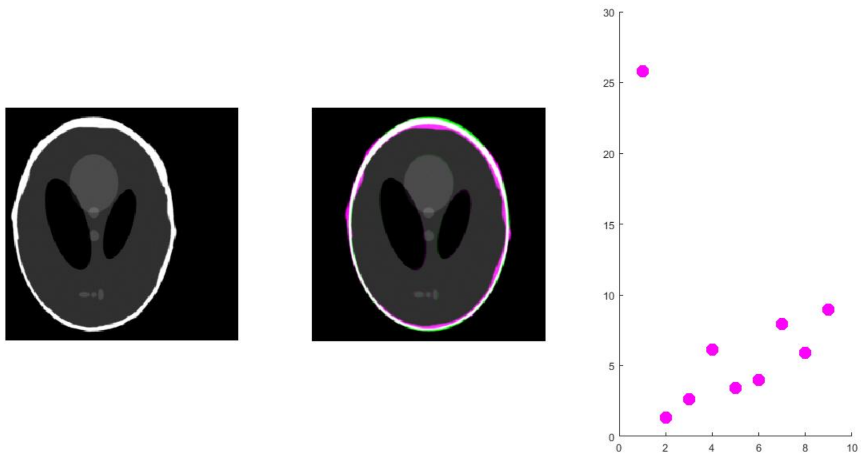
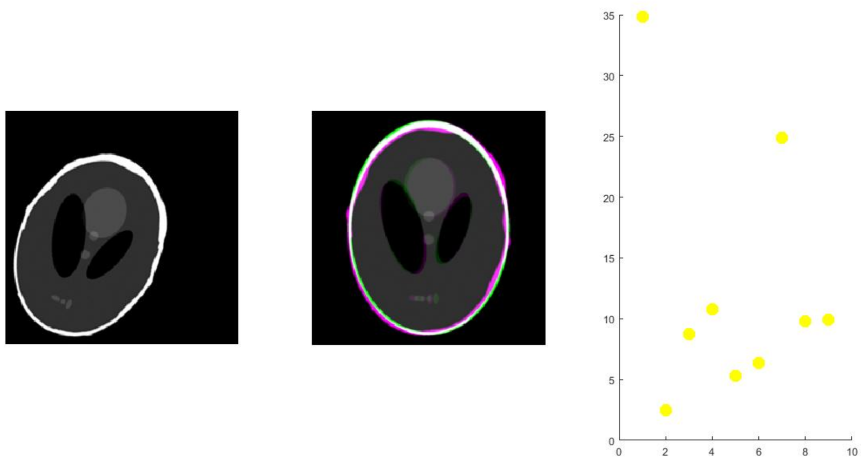


Figure 9: Registration Combination Test



Matlab is able to register these images with varying levels of success. While the errors for every region are well above zero, the program does still give a much higher error for region one as expected. It can be seen however, that for the smaller regions, the mask no longer overlaps the majority of the desired region in the registered image and thus, the error calculated is larger than merited. Registration of images is the shortcoming of this program but can be much improved over the Matlab generic registration by traditional image processing methods. Additionally, for the purpose of comparing the same image after undergoing a variety of preprocessing methods, the final result images should be similar in terms of rotation angle, scaling, and translation. Thus, the error associated with faulty image registration should be comparable for each result image. Thus, these errors can be easily subtracted from one another to cancel each other out. This leaves only the errors due to the preprocessing and reconstruction and still allows for accurate comparison of image quality in each region.

4 Conclusion

The outcome of this project was creating a Matlab program that can compare the quality of images to an original "model" image in selected regions. Programs that reconstruct or otherwise create images often have no way of evaluating the resulting image quality. Though the human eye is capable of detecting most variances, it is not reasonable to rely on a human to go through hundreds or thousands of images and determine which is of highest quality compared to an original image. Computer programs are obviously capable of comparing images but a lot of time can be lost comparing points which are not particularly important and the location of the variances is not always recorded. By creating a masking image that highlights and locates areas of importance such as edges or smaller details, this program only compares the necessary areas and records in which area the variance is occurring. This program seeks to alleviate wasteful human and computational power in comparing images by comparing only the regions which are important.