UNIVERSITAT DE GIRONA

MEDICAL IMAGING ANALYSIS

LABORATORY REPORT

Image Registration

Author
Mohit VAISHNAV

Supervisor
Dr. Robert MARTI
Tutor
Richa AGARWAL

Submission Last Date 4^{th}, May

1 Introduction

Image registration is a process of aligning pixels of an image to another template image. It involves the process of mapping coordinate of an image to the new coordinate system from another one where the transformation involved can be linear or non linear. There might be scenario when images are taken at various circumstances and different time which results in misalignment of various types.

Registration methods could be of many types like determining the corresponding points between the image where the points could be either manually marked or one can detect the features between views. Another way is the transformation between the corresponding points where it can either be assumed that all pairs of corresponding points are related by same transformation or the transformation parameters can be computed for given corresponding points.

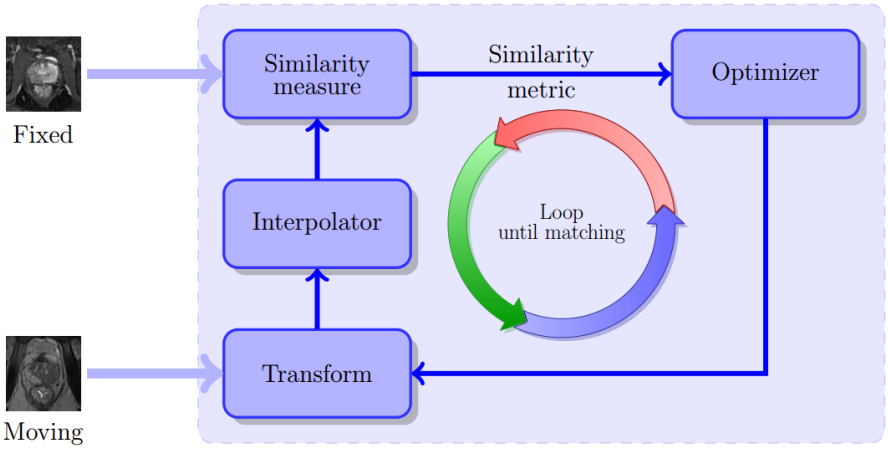

Figure 1: Algorithm

2 Components

Various components associated with the registration (Fig. 1) are:

- *Transform*: It allows movement of moving image depending on some parameters.
- *Metric*: Measures the similarity between fixed and moving.

- *Interpolator*: Determines the pixel intensity given the position and neighbouring pixels.
- *Optimizer*: It finds the best metric value with respect to the transformation parameters.

2.1 Transformation Function

Transformation function can be of two types Rigid / Affine and Deformable. First requires a few parameters and is known as global representation of the transformation whereas *deformable* registration requires many parameters and is known as local representation of the transformation. Angles and length of line segment are preserved in translation and rotation. Various transformation functions can be seen in terms of equations below and their visualization is seen in Fig. 2

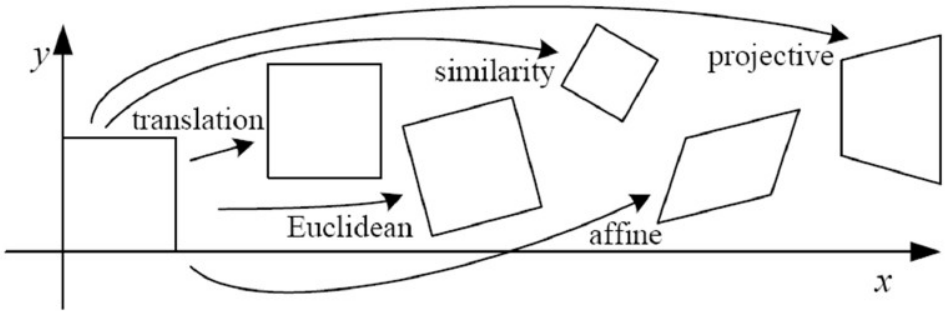

Figure 2: Different Projection

Rigid:

$$f_x(x,y) = xcos(\phi) + ysin(\phi) + t_x$$
$$f_y(x,y) = -xsin(\phi) + ycos(\phi) + t_y$$

Affine:

$$f_x(x,y) = a_x x + a_y y + t_x$$
$$f_y(x,y) = b_x x + b_y y + t_y$$

Projective:

$$f_x(x,y) = \frac{a_x x + a_y y + t_x}{c_x x + c_y y + 1}$$
$$f_y(x,y) = \frac{b_x x + b_y y + t_y}{c_x x + c_y y + 1}$$

2.2 Similarity Metrics

It is the measure of similarity between images which could be done via techniques like *Root mean square, Normalized cross correlation, Entropy related measures, Entropy of the difference image, Mutual information*. Cross Correlation can be summarized in the form of following equation: (Fig. 3). It calculates correlation between the pixels in both the images and if are high in value denotes high similarity. Its maximum value, 1 represents perfectly identical whereas for completely uncorrelated values it gives 0 and it has value -1 for completely anti correlated values.

$$CC(i,j) = \frac{\sum_W (W - E(W))(I_{(i,j)} - E(I_{(i,j)}))}{\sqrt{\sum_W (W - E(W))^2} \sqrt{\sum_{I_{(i,j)}} (I_{(i,j)} - E(I_{(i,j)}))^2}}$$

Figure 3: Normalized Cross Correlation

2.3 Optimizer

An optimizer finds the transformation parameters that maximizes the *similarity measure*. Gradient Descent Method is one of the common method in use where it is required to specify the gradient of the matrix w.r.t. transformation parameters. Exhaustive search is required to be done in whole image which becomes computationally demanding. Few of the techniques can be summarized as follows: Gradient Descent optimization finds the maximum of *Mutual* information , LM optimization minimizes the variance in intensities of corresponding pixels.

3 Registration Framework

3.1 Transformation

In file *affineReg2D.m* on line number 18, parameter for using type of transformation (*ttype*) is defined with *a* representing affine and *r* for rigid. Different set of parameters are used for each type of transformation.

3.2 Similarity Metric

Amongst various similarity metric, *sum of squared differences* and *normalized cross correlation* have been utilized in this program. This is included in file *affine_function.m* at line number 54-57.

3.3 Interpolator

In file *affine_function.m* at line 47, 3^{rd} parameter to be passed represents *interpolation* technique to be used while executing the function file *affine_transform_2d_double.m*. Various cases in the same are *bilinear, bicubic and nearest neighbour with replicate and zero boundary* and could be selected from values ranging from 0-5 as seen in the same function file with line number 55-74. At the end for final output of the interpolated image, *image_interpolation* is called.

3.4 Optimizer

Two type of optimizer have been included in the file *affineReg2D.m* at line number 47-48 with the name *fminunc and fminsearch*. They are nonlinear programming solver and searches for the minimum of a problem specified with the difference of one using derivative method while the other doesn't.

3.5 Part II

- Scale
  - It is used to set the translation, rotation and resize parameters of real image. Its dimension is 3 when rigid option is chosen in *ttype* whereas 7 parameters are chosen in case of affine transformation. It has been used in *affineReg2D.m* at line number 24 and 31 with $float$ type. Also in case of *multi resolution* part, scaling changes different at each level like translation is directly related to change of scale of an image while rotation and shear are kept the same.
- Gaussian Smoothing
  - Many a times there is noise present in the image so to get rid of this extra information and blur the image, *Gaussian Smoothing* is applied over the image. It is very effective when there is zero

mean and is also rotational invariant. *affineReg2D.m* applies this on line number 14-15.

- Center of Rotation
  - It is defined from the origin of the Z axis in a coordinate plane of an image.

4 Similarity Metric

Following equation is used to implement the normalized cross correlation of an image wrt another.

$$e = -\frac{\sum(Ifixed - I_A) \times (I - I_B)}{\sqrt{\sum(Ifixed - I_A)^2 \times \sum(I - I_B)^2}}$$

Switch case has been included with the variable *mtype* and having variables *cc* and *s* representing *Normalized Cross Correlation* and *Sum of Squared differences*. Negative sign is included in the cross correlation function to get the minimum instead maximum value.

5 Affine Transformation

As explained in Subsection 3.1, two types of functions has been created with the initials *a* for *affine* and *r* for rigid type. They too are passed in the optimizer function for evaluation purpose. Switch case has been used to make user selects which type of function do they want the program to run. Three parameters are required for the *rigid* case whereas seven parameters have been defined to reduce the coding complexity in second case. The same has to be appended in the MATlab function filed named *affine_function*.

For $ttype = r$

$$scale = [1.0 \quad 1.0 \quad 0.1]$$

$$x = [0.0 \quad 0.0 \quad 0.0]$$

For $ttype = a$

$$scale = [1.0 \quad 1.0 \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.1]$$

$$x = [0.0 \quad 0.0 \quad 0.0 \quad 1.0 \quad 1.0 \quad 1.0 \quad 1.0]$$

Different set of weights are assigned to each values because of their different intensity of contribution in the initial phase. Giving them all weights as 1 changed to output of the program and resulted in distorted image as they cannot be converged.

6 Multi Resolution

Scaling factor of 2 is used to change each level in multi resolution. Input Moving and Fixed images are scaled from the lowest level moving gradually to the original shape.

$$ISmoving\_temp = imresize(ISmoving, \frac{1}{(2^{i-1})}) \quad \forall i = 3, 2, 1$$

$$ISfixed\_temp = imresize(ISfixed, \frac{1}{(2^{i-1})}) \quad \forall i = 3, 2, 1$$

In addition to implementing this, scaling factor for the translation has to be changed in parallel to the size of scaling. Moving in a scale of 2, helps in changing this parameter by a factor of 2 instead any other dissimilar value.

$$x(1:2) = 2 \times x(1:2) \quad \forall i = 3, 2, 1$$

Fig. 4, 5, 6, 7, 8, 9, 10, 11, shows the result when testing the code for different brain images. Using brain4.jpg we were not able to run this program.

7 Difficulties Faced

Implementation of *Cross Correlation* was one of the biggest challenge which took most of the time looking for the bug in the program. Although the implementation was right but conceptually there was an issue where it is required to get the maximum correlation and then revert the sign of it. Doing this shall send optimizer the value which is least instead of maximum and hence it converges both the images altogether.

Moving ahead, *Multi-resolution* is to be implemented where there was a missing factor in translation factor of the affine/rigid transformation matrix. As the image is scaled down/up, the same has to be done with the parameters associated with it too. If any image scale changes, translation vector varies directly with it. Another problem faced was to initialize the parameters x and *scale* outside this loop which if not done reverts the effect of any kind of scaling and hence distorts the whole purpose.

Figure 4: Distorted Image to be registered

References

[1] Tutorial,
http://www.math.tau.ac.il/t̃urkel/notes/registration.pdf

[2] Image Registration Techniques: A Survey,
https://arxiv.org/pdf/1712.07540.pdf

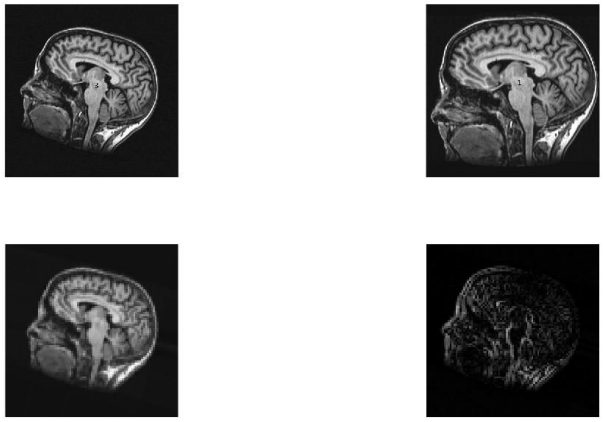

Figure 5: Result in Iteration 1

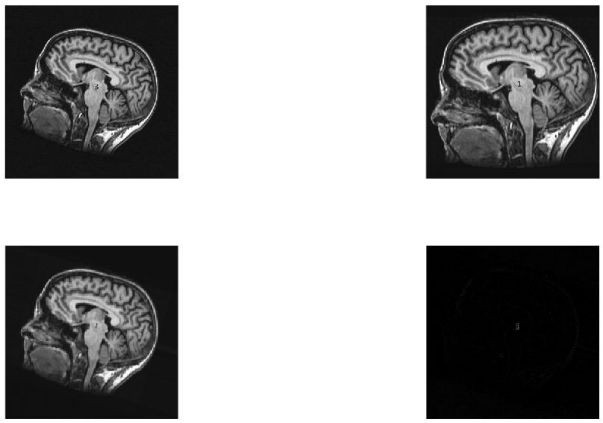

Figure 6: Result in Iteration 2

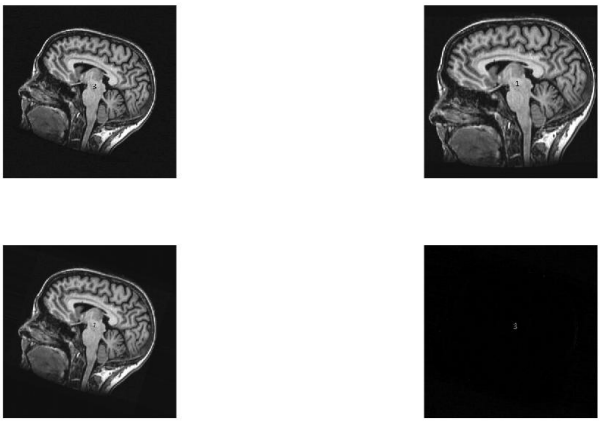

Figure 7: Result in Iteration 3

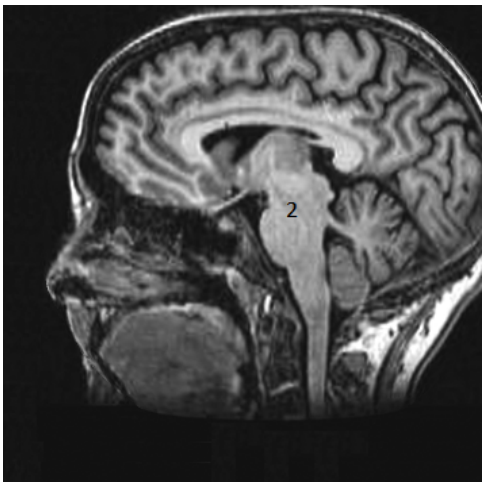

Figure 8: Distorted Image to be registered brain 2

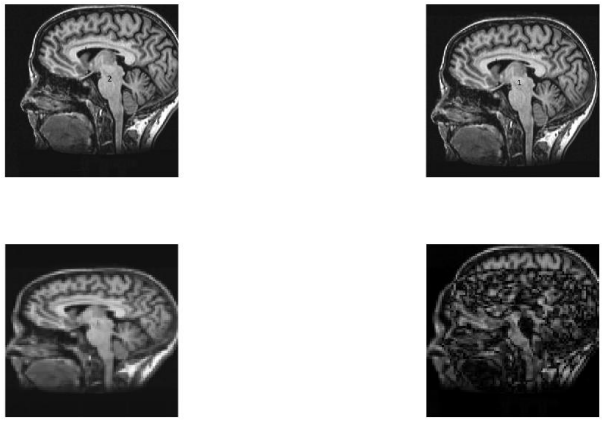

Figure 9: Result in Iteration 1

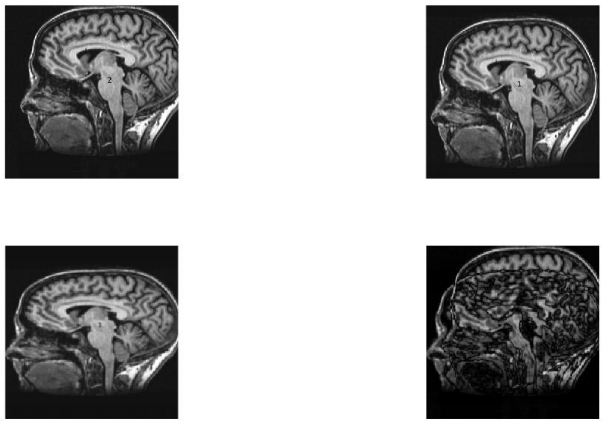

Figure 10: Result in Iteration 2

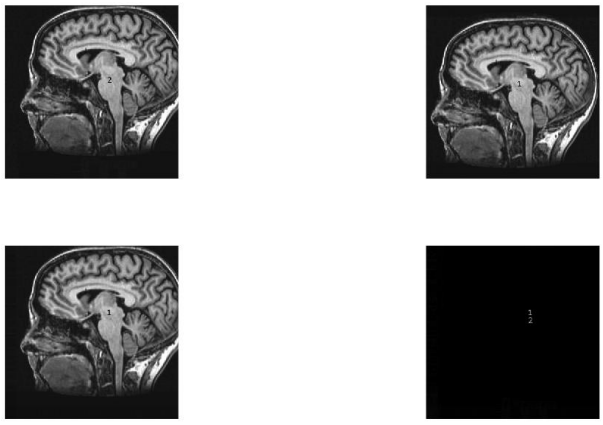

Figure 11: Result in Iteration 3