Intensity Based Image Registration

Medical Image Registration and Applications

Kaouther Mouheb Joaquin Oscar Seia

I. Introduction

The process of *image registration* consists in inferring or estimating the geometric transformation that allows an *unregistered/moving image* to be aligned with a *template* or *fixed image* of the same scene. These methods are milestones in many medical image pipelines, for example in following the evolution of a patient across different examinations with the same or different image modalities.

Across the years many variations of these techniques have been proposed, however they mostly share a common pipeline. The first step is deciding which kind of transformation between the images we want to estimate, mainly it can be parametric or non-parametric. In the former, the parameters of the transformation are going to be estimated through an optimisation process pursuing the minimisation of the difference between the images (or the maximisation of their similarity). In order to properly transform the moving image, in each step an interpolation method is applied. The desired intensity values in the moving image are obtained by mapping its pixels coordinates to the fixed image space and inferring their values by combinations of the intensities of their neighbouring pixels.

II. Objective

The primary objective of this laboratory was to interact with an intensity based image registration framework. By analysing its components, modifying it to incorporate other similarity metrics and trying out the different kinds of affine transformations, a deeper understanding of the technique was achieved. Finally we arrived to a conclusion on the benefits of using a multi-resolution against a single resolution approach.

III. Theoretical Background

A. Components of an image registration framework

1) Inputs

Fixed image: This is the image used as a reference across the complete registration process.

Moving image: This is the image that is going to be iteratively modified in order to match the fixed one.

Parameters to estimate: The affine transformation is the general transformation described by 6 parameters that can model the composition of rotation, translation, shearing and scaling. However, if we have some knowledge on the transformation, we can constraint the procedure to infer just one of the previously mentioned individual transformations. This can have an impact on the number of parameters estimated and therefore on the computational cost of the complete procedure.

Interpolation method: During the interpolation phase of the procedure (see the following item), different methods can be used. The main difference between them is the number of neighbouring points they use to infer the desired pixel intensity value. Choosing one or the other is always a trade-off between computational cost and the accuracy required in the estimation. Ordered from the simplest to the most complex, the most common ones are [1]: nearest neighbour(1), bilinear(4), bicubic(9).

2) Components of the iterative procedure

Similarity measure: First we compute the similarity between the two images. This gives us a quantitative measure of how different (mis-registered) are the images, the main idea of the framework is that we want to modify the transformation function in order to maximise the similarity between the images. Optimisation processes are conceived as minimisation one, so we construct error measures that are opposed to the similarity ones. Three possible error measurements are:

- *Mean Squared Error:*

$$MSE = \frac{1}{N}\sum_{i,j}(Im_{i,j} - If_{i,j})^2 \tag{1}$$

- *Normalised Cross Correlation Error:*

$$NCCE = 1 - \frac{\sum_{i,j}(Im_{i,j} - \overline{Im})(If_{i,j} - \overline{If})}{\sqrt{\sum_{i,j}(Im_{i,j} - \overline{Im})^2 \sum_{i,j}(If_{i,j} - \overline{If})^2}} \tag{2}$$

- *Normalised Gradient Cross Correlation Error:*

$$p(i,j) = \left|\frac{\partial Im(i,j)}{\partial i}\frac{\partial If(i,j)}{\partial i} + \frac{\partial Im(i,j)}{\partial j}\frac{\partial If(i,j)}{\partial j}\right| \tag{3}$$

$$e_1(i,j) = \frac{\partial Im(i,j)}{\partial i}^2 + \frac{\partial Im(i,j)}{\partial j}^2 \tag{4}$$

$$e_2(i,j) = \frac{\partial If(i,j)}{\partial i}^2 + \frac{\partial If(i,j)}{\partial j}^2 \tag{5}$$

$$NGCCE = 1 - \frac{\sum_{i,j} p(i,j)}{\sqrt{\sum_{i,j} e_1(i,j) \sum_{i,j} e_2(i,j)}} \tag{6}$$

Where N denotes the total number of pixels in the image, *Im* the moving image, *If* the fixed image, and *(i,j)* the pixel in the i^{th} row and j^{th} column.

Optimiser: The registration process is achieved iteratively by estimating the parameters of the registration transformation function using an optimisation process that aims to find the optimal set of parameters that minimise the error measure. The optimiser used in our framework is the Nelder-Mead Simplex Method described in [1]. This optimisation algorithm tries to minimise a non-linear function f, in our case the error measure between the fixed and moving image, using values of this function at some points without calculating its derivatives.

[1]Between parenthesis, the number of points used by each algorithm to interpolate the desired intensity

The method starts with a simplex of n+1 vertices for an n-dimensional space and iteratively applies a transformation by computing one or more test points of the function f. By comparing the simplex members of the complex to the test points, the algorithm redefines the simplex to decrease the function values at its vertices. The process ends when the simplex becomes sufficiently small or when the function values at the points inside it becomes close in some sense. The algorithm is not guaranteed to converge to a global minimum.

Examples of the converging optimisation process showing the iterative minimisation of the mean square error measure for image 'brain1.png' can be appreciated in the Appendix in Figures 6 and 7 .

Transform: a transformation function that defines the deformation that each pixel in the moving image suffers during to obtain the registration. In this work, we used two different parametric transformations. The first one was the *rigid transformation* which models two operations: translation and rotation. In this case, all the pixels in the image are subject to the same transformation. The parameters to be estimated in this case are: the translation along the y-axis, the translation along the y-axis and the rotation angle. The second is the complete *affine transformation* which includes translation, rotation, scaling, and shearing. In this case, different pixels can have a different transformation, thus, this type is non-rigid. The parameters to be estimated in this case are: the translation along x-axis, the translation along y-axis, the rotation angle, the scale along x-axis, the scale along y-axis, shearing on the *xy* direction and shearing on the *yx* direction.

Interpolation: Applying the forward transformation of an image registration algorithm usually results in non-integer values for the corresponding coordinates in the moving image which makes inferring the pixel intensity values difficult at multiple positions in the registered image. For this reason, the inverse transformation is used. In such a way, for each pixel in the registered image we can infer the intensity value based on the result of the inverse mapping. However, these results are usually non-integer as well. Interpolation aims to find the value of the pixel in the moving image based on this non-integer result and the neighbourhood surrounding it. Multiple interpolation techniques exist. In our implementation, the cubic interpolation was used.

3) Quantitative evaluation

To evaluate the results of the registration quantitatively, we used two metrics. First, the *mean squared error* between the fixed image and the result according to equation (1). The second metric is a similarity measure called *mutual information*. It is a measure of how well you can predict the signal in the second image, given the signal intensity in the first. It is given by equation (7).

$$MI(If, Im) = \sum_{i,j} p(i,j) log(\frac{p(i,j)}{p(i)p(j)}) \tag{7}$$

where p(i, j) is the normalised 2D histogram between the fixed image and the registered image, p(i) is the marginal for i over j and p(j) is the marginal for j over i.

IV. Assignments

A. Registration Framework

Following the previous section, we can identify every block of the registration framework in the generated codes (which are available in the following repository: github.com/kakou34/mira_lab/tree/main/lab1).

The complete registration is performed by the function *affine_registration_2d(Im, If, mtype, ttype)*. In this function we can clearly see that the inputs are the same as the ones presented in the previous section. If we want to locate each component of the previous section is the code we would get:

- **Error measure**: They are defined in the *affine_registration_function.m* file inside the *affine_registration_function*. The measure to be used is selected using the *mtype* argument of the function. This function is what the optimiser tries to minimise. The framework supports 3 types of metrics:
  - Mean Squared Error: line 72
  - Normalised Cross-Correlation: [74-79]
  - Normalised Gradient Cross-Correlation: [81-96]
- **Optimiser**: in this setup, the fminsearch function provided my MatLab is used to perform the optimisation process in *affine_registration_2d.m* lines [31-36]. Is important to notice that originally the provided codes were meant to be used with a gradient descent (GD) algorithm and therefore there was a **scaling vector** provided. In the GD method this vector contained the learning rate that was going to be used to scale the update of the parameters in the gradient direction at each iteration. However, since the Nelder-Mead is not a gradient descent method, these scaling factors are not meaningful and can further affect the framework method. For this reason, the scaling vector is set to ones in all its elements (lines 23 and 26).
- **Transform**: the framework supports two types of transformations, rigid and affine(non-rigid) as explained in the previous section. The transformations are defined in *affine_registration_2d.m*. The rigid transformation is defined by a 3x3 matrix of the form:

  $$\begin{bmatrix} cos(\theta) & sin(\theta) & t_x \\ -sin(\theta) & cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

  where θ is the rotation angle t_x is the translation in x parameter and t_y is the translation in y parameter. This is defined in lines [44-46].
  The affine transformation is defined as the multiplication of four 3x3 matrices (lines [50-66]) :
  - Translation Matrix:

    $$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

    where t_x and t_y are the translation parameters.
  - Rotation Matrix:

    $$\begin{bmatrix} cos(\theta) & sin(\theta) & 0 \\ -sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

    where θ is the rotation parameter.

– Scaling Matrix:

$$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where s_x and s_y are the scaling parameters.

– Shearing Matrix:

$$\begin{bmatrix} 1 & sh_{xy} & 0 \\ sh_{yx} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where sh_{xy} and sh_{yx} are the shearing parameters.

Note that in both transformations, the resulting transformation matrix is inverted since the inverse mapping technique is used to perform interpolation. The transformation operation is applied in *affine_transform_2d_double.m*. Initially, scaling parameters were set to 1 while all remaining parameters where set to 0. As we can see in this file (lines 38-39) the center of the coordinates (0, 0) is assigned to the center of the moving image (registration output img_out) therefore, **the center of rotation of the transformation is the center of the registered image**.

It is important to notice that all parameters involved in the transformations are defined by the x array, initialised in lines 22 and 25.

- **Interpolation**: the interpolation function is used as given in *image_interpolation.m*

B. Experiments on Similarity Metric. Normalised Cross-correlation and Normalised Gradient Cross-correlation.

In order to be sure the registration framework was working correctly, before analysing the provided brain images we decided to test our codes with simple images. We created 4 images, (see Figure 1):

1) white star in a black background
2) translation of (1)
3) affine transform of (1)
4) negative of (3)

For these images we applied the registration framework to estimate a **rigid transformation**. After the images are transformed we computed two performance metrics, Mean Square Error (MSE) and Mutual Information (MI). This quantitative analysis will be common to all the following experiments. It is important to highlight that in the images with negated intensities, MSE is not a useful metric to evaluate performance since the intensities of the structures don't match even if the images are aligned, Mutual Information is instead robust against this.

The results [2] can be appreciated in Table I. As it was expected, the rigid transformation parameters estimation in the case of real rigid deformation worked perfect, with null mean square error and high mutual information. Visual results [3] can be seen in Figure 1. When estimating rigid transformation parameters, as expected, the resulting transformation matrix fails to register perfectly affine transformations. In the same figure, we can see that the use of the normalised gradient cross-correlation does not led to good results, the images are practically the same (changes in the transform parameters have been checked to exist). this was expected for the inverted intesities image, since NGCC uses the edges information and should be robust to it. Later use in multi-scale approach will demonstrate a good performance for this measure.

Real Transf.	Time	Iter.	Error measure	MSE	MI
Rigid	**4.7191**	317	MSE	**0.0000**	0.8891
	4.7670	300	NCC	**0.0000**	**0.8892**
	5.6783	**247**	NGCC	0.1239	0.2167
Affine	**2.9742**	**175**	MSE	**0.0542**	**0.4514**
	4.1556	264	NCC	**0.0542**	0.4513
	8.8970	399	NGCC	0.0560	0.4485
Affine Negative	5.5264	354	MSE	**0.6007**	0.0105
	3.0044	**197**	NCC	0.6556	0.0259
	5.5675	210	NGCC	0.8265	**0.1872**

TABLE I
RIGID TRANSFORMATION PARAMETERS ESTIMATION USING THE SIMPLE **STAR** IMAGES IN **SINGLE SCALE**.

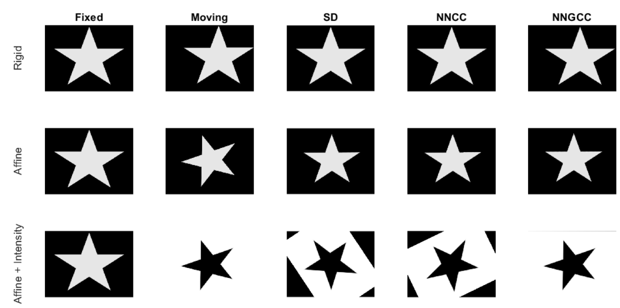

Fig. 1. Example of rigid transformation parameters estimation using the simple star images in single scale.

Once the framework was tested, we proceeded with the brain images. Table II presents the results. Figure 2 shows the visual results. We can clearly appreciate, that when the real transformation is a rigid one the method can almost perfectly estimate it. The mean square error function does not reach zero due to the missing part of the skull in the moving image, which doesn't allow a perfect match. In the same figure we can appreciate that the more complex error measures used in the optimisation process did not appear to help in this case. The main reason for this might be the algorithm stopping at local minima and not reaching the global minima. This hypothesis was backed up with some tests done changing the initialisation values for the parameters which led to different and visually better results. Also, the promising results obtained with the simple images back up this hypothesis, confirming that the algorithm works well even for those error measures. NGCC error measure still seems not to generate profitable results, not even in the negative intensity images.

C. Transformation. Affine Transformation

The same procedure done for rigid transformation was applied for affine transformation estimation: first we worked with the star and then with with the brain images. However, for better

[2]For all the tables: MSE: Mean Square Error, MI: Mutual Information, NCC: Normalised cross-correlation, NGCC: Normalised Gradient cross correlation

[3]All the figures in the report include the three different images with specific underlying transformations in the rows and the three error measures results for each of them as columns.

Real Transf.	Time	Iter.	Error measure	MSE	MI
	7.1785	328	MSE	**0.0039**	**2.5128**
Rigid	11.5378	576	NCC	0.0537	0.3285
	5.9923	**218**	NGCC	0.0580	0.2787
	4.3571	**221**	MSE	0.0566	0.3152
Affine	10.5846	547	NCC	**0.0527**	**0.3740**
	8.9761	317	NGCC	0.0572	0.3291
	12.3416	632	MSE	**0.2037**	0.0884
Affine Negative	15.0678	684	NCC	0.2524	**0.3179**
	5.6933	**211**	NGCC	0.3829	0.2746

TABLE II
RIGID TRANSFORMATION PARAMETERS ESTIMATION USING BRAIN IMAGES AND SINGLE SCALE.

Real Transf.	Time	Iter.	Error measure	MSE	MI
	15.5425	1000	MSE	**0.0487**	**0.4208**
Rigid	16.2623	**909**	NCC	0.0493	0.4081
	22.3102	1000	NGCC	0.0629	0.2441
	16.0919	**1000**	MSE	**0.0257**	**0.7592**
Affine	16.5559	1000	NCC	0.0257	0.7300
	23.9389	1000	NGCC	0.0417	0.4300
	14.0976	1000	MSE	**0.0457**	0.3511
Affine Negative	**11.7673**	**670**	NCC	0.1576	0.3962
	22.1548	1000	NGCC	0.3519	**0.5518**

TABLE III
AFFINE TRANSFORMATION PARAMETERS ESTIMATION USING BRAIN IMAGES AND SINGLE SCALE.

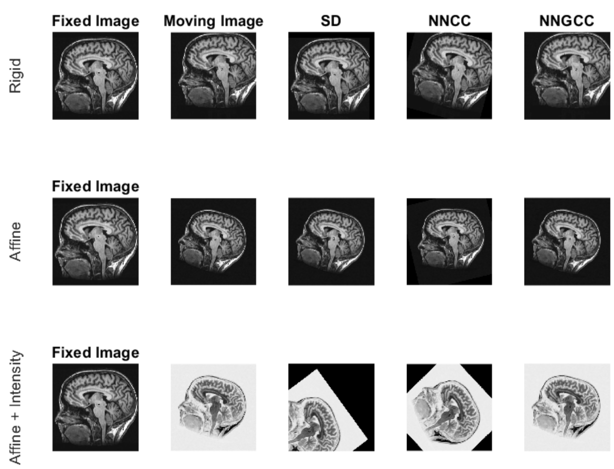

Fig. 2. Example of rigid transformation estimation using brain images in single scale.

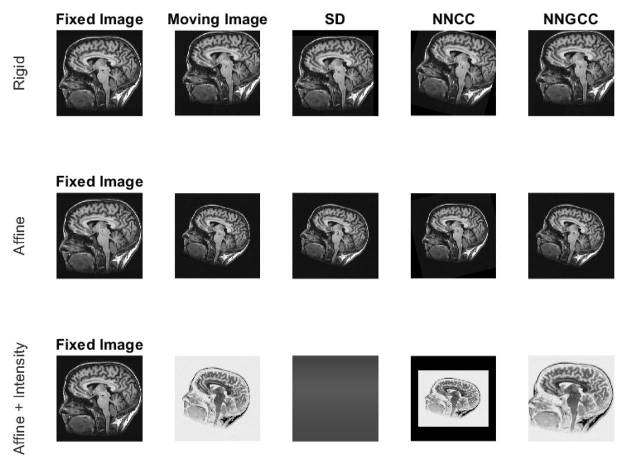

Fig. 3. Example of affine transformation estimation using brain images in single scale

clarity only the results obtained with brain images are presented here.

Table III presents the results. Figure 3 shows the visual results. We can observe that even if the affine transformation does a decent job in terms of metrics, visually the results are not the best ones. When the real transformation to be estimated was affine the algorithm performs a little bit better according to the quantitative measures, but visually it seems that the scaling parameter is not being estimated properly. It can be seen from the number of iterations that in most cases the algorithm early stopped after 1000 iterations, suggesting that perhaps more iterations can improve these results. Another necessary comment is the comparison of computation time between affine and rigid parameters estimation (Tables III and II of previous section), being the first one between twice or three times higher. Both commented results are logical, the more parameters to be estimated, the more complex and more demanding in time is the optimisation procedure. Finally, in this experiments NGCC seems to have a positive impact in negative intensity images, visually the registration worked much better.

D. Multi-resolution

The developed registration framework can be further improved by its application in a multi-scale fashion. The idea behind this method is to construct a scale/resolution pyramid out of the image by downsizing it several times (halving the resolution on each step). After this, the registration procedure starts using the smallest image, the low resolution details and the small size allow us to obtain of an initial coarse transformation matrix very fast. The resulting estimated parameters (properly scaled) are used as initial values in the next resolution level. This procedure is repeated until we reach the biggest resolution. Finally, the resulting transformation matrix is used over the original moving image to obtain its final transformed version.

This procedure was applied with 3 different number of scaling steps ({2,4,6}), for the two transformation parameters (rigid and affine) for the three images. For each image one error measure was chosen, NGCC was the logical choice for the negative image, for the other two each of them was selected in order to try them all.

The results for rigid parameters estimation are shown in Table IV. When trying to estimate rigid transformation parameters, the multi-scale approach does not imply a better results. However, it can be seen that the computational times of the multi-scale approach are better than the single one. This can be a consequence of less iterations needed due to better parameters initialisation in the multiscale approach helping the fast convergence of the optimiser.

The results for affine parameters estimation are shown in Table V. This results clearly support the use of multi-scale approach in affine transformation's parameters estimation. Moreover, the use of four scaling steps led to relatively good results in

Real Transf.	Time	Iter.	Scales	Error	MSE	MI
	7.1785	**328**	1	MSE	**0.0039**	**2.5128**
Rigid	**1.2397**	429	2	MSE	0.0474	0.4902
	3.5132	767	4	MSE	**0.0039**	2.4044
	29.4457	1006	6	MSE	0.0790	0.1441
	10.5846	547	1	NCC	**0.0527**	**0.3740**
Affine	**1.4750**	**539**	2	NCC	0.0545	0.3669
	5.0558	919	4	NCC	0.0547	0.3423
	55.4829	956	6	NCC	0.0572	0.3086
	5.6933	**211**	1	NGCC	0.3829	0.2746
Affine Neg.	**2.5204**	289	2	NGCC	**0.3836**	**0.2954**
	5.8454	597	4	NGCC	**0.3836**	0.2941
	59.0380	919	6	NGCC	0.3836	0.2882

TABLE IV
RIGID TRANSFORMATION PARAMETERS ESTIMATION USING BRAIN IMAGES AND MULTIPLE SCALES.

all the images. For that reason in Tables VI and VII, we compute the results for all error measures and all images using 4 levels of resolution. Similar results as the ones discussed before suggest that when the underlying transformation is rigid or affine, either MSE or NCC error measures are good choices in multi scale approach. In the case of negative images, NGCC is the best choice.

In Figures 4 and 5, we can see the obtained transformations, which are visually consistent with the good metrics previously shown in the tables.

Real Transf.	Time	Iter.	Scales	Error	MSE	MI
	15.5425	**1000**	1	MSE	0.0487	0.4208
Rigid	**3.3590**	2000	2	MSE	0.0523	0.3212
	7.5636	2796	4	MSE	**0.0039**	**2.2486**
	88.0282	4020	6	MSE	0.0790	0.1441
	16.5559	**1000**	1	NCC	0.0257	0.7300
Affine	**6.2790**	1664	2	NCC	0.0206	0.9517
	7.1434	2381	4	NCC	**0.0003**	**2.4642**
	88.5930	3399	6	NCC	0.0463	0.4418
	22.1548	**1000**	1	NGCC	0.3519	0.5518
Affine Neg.	**8.4214**	2000	2	NGCC	0.3517	0.7443
	13.0196	2578	4	NGCC	0.3808	**2.1097**
	122.4563	3792	6	NGCC	**0.3323**	0.4996

TABLE V
AFFINE TRANSFORMATION PARAMETERS ESTIMATION USING BRAIN IMAGES AND MULTIPLE SCALES.

Real Transf.	Time	Iter.	Error measure	MSE	MI
	3.5132	767	MSE	0.0039	2.4044
Rigid	3.7110	838	NCC	**0.0039**	**2.4982**
	7.6199	**709**	NGCC	0.0538	0.2987
	4.1800	**749**	MSE	**0.0546**	0.3422
Affine	5.0558	919	NCC	0.0547	**0.3423**
	7.7569	890	NGCC	0.0579	0.3106
	3.6047	1139	MSE	**0.1812**	0.1079
Affine Negative	4.4993	764	NCC	0.2150	0.1704
	5.8454	**597**	NGCC	0.3836	**0.2941**

TABLE VI
RIGID TRANSFORMATION PARAMETERS ESTIMATION USING BRAIN IMAGES, MULTISCALE FRAMEWORK WITH 4 SCALING STEPS.

Real Transf.	Time	Iter.	Error measure	MSE	MI
	7.5636	**2796**	MSE	0.0039	2.2486
Rigid	9.3974	2878	NCC	**0.0039**	**2.3298**
	30.6148	4000	NGCC	0.0517	0.3458
	6.4752	**2313**	MSE	0.0003	2.4639
Affine	7.1434	2381	NCC	0.0003	2.4642
	14.5383	2762	NGCC	**0.0003**	**2.4647**
	14.8292	4000	MSE	0.1670	0.0504
Affine Negative	21.2549	3474	NCC	**0.1584**	0.4203
	13.0196	**2578**	NGCC	0.3808	**2.1097**

TABLE VII
AFFINE TRANSFORMATION PARAMETERS ESTIMATION USING BRAIN IMAGES, MULTISCALE FRAMEWORK WITH 4 SCALING STEPS.

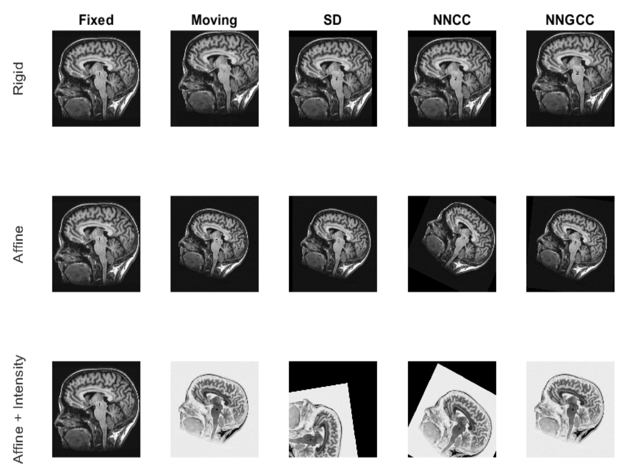

Fig. 4. Example of **Rigid** transformation estimation using brain images in **multi-resolution** framework with 4 scales.

V. CONCLUSIONS

Across this laboratory we got exposed to the multiple stages of a registration framework, we deal with technical challenges as understanding a new optimisation method or implementing the multi-resolution approach. After all the experiments, we conclude that multi-scale approach is a better choice when the parameters to be estimated are affine. IF the parameters trying to be estimated are rigid, single scale approach is the best choice.

Furthermore, we didn't find major evidence suggesting that normalised cross correlation is a better error measure than mean square error when the real transformations between the images are affine or rigid. When intensity changes occur between the images the normalised gradient cross correlation is the right choice as error measure when estimating affine parameters or when using multi-scale approach.

REFERENCES

[1] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, "Convergence properties of the nelder–mead simplex method in low dimensions," *SIAM Journal on Optimization*, vol. 9, no. 1, pp. 112–147, 1998. [Online]. Available: https://doi.org/10.1137/S1052623496303470

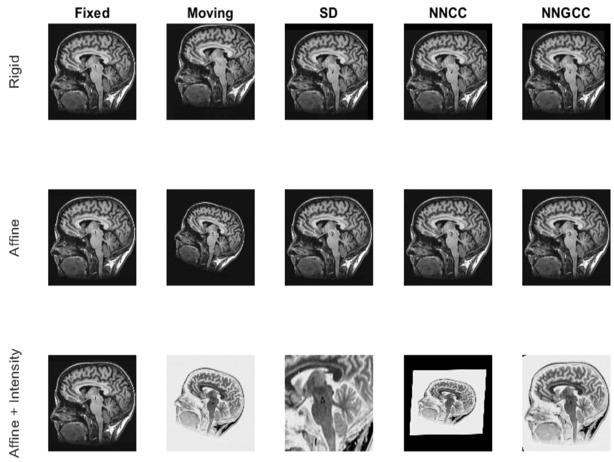

Fig. 5. Example of **affine** transformation estimation using brain images in **multi-resolution** framework with 4 scales.

APPENDIX

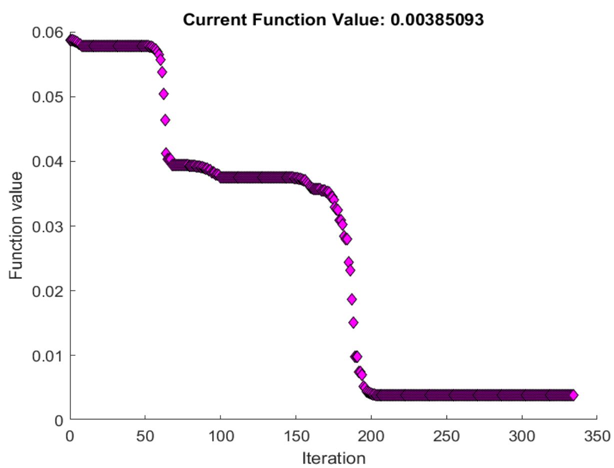

Fig. 6. MSE Error per iteration using a rigid transformation

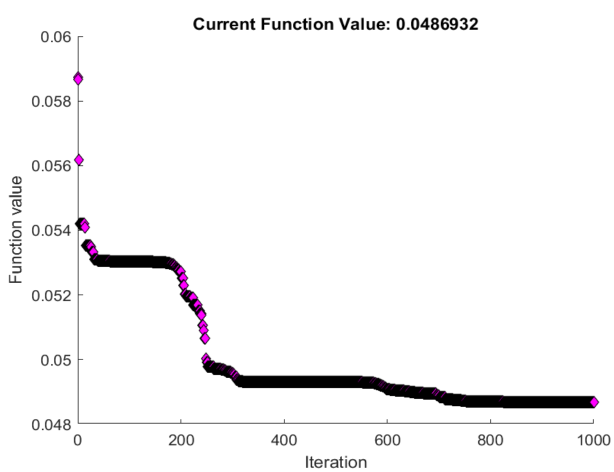

Fig. 7. MSE Error per iteration using an affine transformation