EBME/CSDS 361/461. Homework 4. Spring.
David L. Wilson, PhD
Department of Biomedical Engineering
Homework 4: Image Registration

All homework assignments must be completed individually (no teams/groups).

Your responses to each question to each should include:

- A short discussion of each problem. ***Code/images alone will not suffice.*** A brief description/explanation should accompany each image/algorithm.
- Any processed images
- All MATLAB code used to generate results and images. NOTE: MATLAB/Python code may be randomly tested by TA for functionality!
- Be sure to comment your MATLAB/Python Scripts thoroughly! We need to understand your approach to each problem.

Please submit the following files in Canvas:

- Your responses to each question in PDF format, titled "`[caseID]_HW04.pdf`". **Be sure to include all generated images in your PDF report.**
- All generated images, MATLAB/Python code in one zip file titled "`[caseID]_HW04_Code.zip`"

Please read the question descriptions carefully. Be sure to address every question. We try to give you sufficient information to get you successfully started on this assignment.

The following files will be provided by the TA:

- Contrast1-new.tif
- Contrast2-new.tif
- mri1.tif
- mri2.tif
- origmri.tif
- taskmri.tif
- live_new.tif
- mask_new.tif
- Similarity Measures.pdf

In this homework you will develop a gray-scale image registration method to correct the displacement of an image. Please read the entire file prior to doing the project, as some useful functions and pseudo code are included below!

Questions

1. You will recall from class 3 primary similarity metrics: (1) Normalized Cross-Correlation, (2) Mutual Information, and (3) Sum of Squared Error (also known as sum of squared differences). Here you will investigate another, Sum of Absolute Difference. Write a function "mySumAbs.m" which computes normalized cross correlation between two input images. You will call the function as follows:

```
R = mySumAbs(image1, image2)
```

Here R is a measure of how much similarity there is between the two images (R=0 → perfectly similar).

Paste the code for this function in your submission.

2. Contrast2_new.tif is a translated version of Contrast1-new.tif. It is translated by 1.8 pixels and 2.1 pixels in the x and y directions, respectively. When you do this operation, there will be an "overlap region." Note: in general, for registration, when an image is transformed, it will not longer exactly overlap. This will be an issue below. It is not necessary for you to compute a similarity metric to the ends of an image. Often one might use a sub-region for the calculation.
    a. Calculate mySumAbs with the two input images.
    b. Translate Contrast2-new.tif by the required amounts so that the images are registered. (This will require image interpolation,) Calculate mySumAbs. Report the value(s) obtained. *This is a QUANTITATIVE method of evaluating registration. Of course, to do this, you will need to use an image interpolation approach such as bilinear. NOTE: You should use matrix operations versus looping to do this problem. It will come in handy for the subsequent problems.*
    c. Create a subtracted image to ensure that the images are registered. Display your final subtracted image. *This is a QUALITATIVE way of evaluating registration.*

3. Now, register the same image pair from problem 2 using the "Nelder-Mead simplex minimization algorithm", **fminsearch**.
    a. Use your function, mySumAbs.
    b. Use x and y translation only (no rotation, no scaling, no shearing). You will need to evaluate a similarity cost function (that is, SSE) and *iteratively minimize the cost function*. Start from a reasonable initialization, say (0,0).
    c. The minimization algorithm should iteratively compute the optimum x and y translation parameters and evaluate the cost function. Display Contrast1_new side by-by-side with translated Contrast2_new and report the "optimal" translation in x and y yielded by the algorithm.
    d. At each step in the optimization algorithm (when each new Tx and Ty is computed), please using MATLAB's getframe() to iteratively capture the difference image between the fixed image and the translated image. The first frame should be the difference image before any translation and the last frame should be the difference image after registration has been optimized. You can visualize your frames as a movie using movie(); but please also export this movie using movie2avi() for ease in grading.
    e. For any evaluation of registration, you should include 2 original images, the subtracted image without registration, and the subtracted image with registration.

4. *Adjust **fminsearch** parameters.* Modify the algorithm in Problem 3 as follows:
    a. Use different stopping criteria (*the tolerances*). *Sample a range of values.*
    b. In addition, implement the scaling parameter described later which solves the problem of making step sizes in the optimization too small. *Sample a range of values.*
    c. Describe your experiences. *I suggest making a small table with the results of registration as you modified parameters. I'd like to know how these parameters affected*

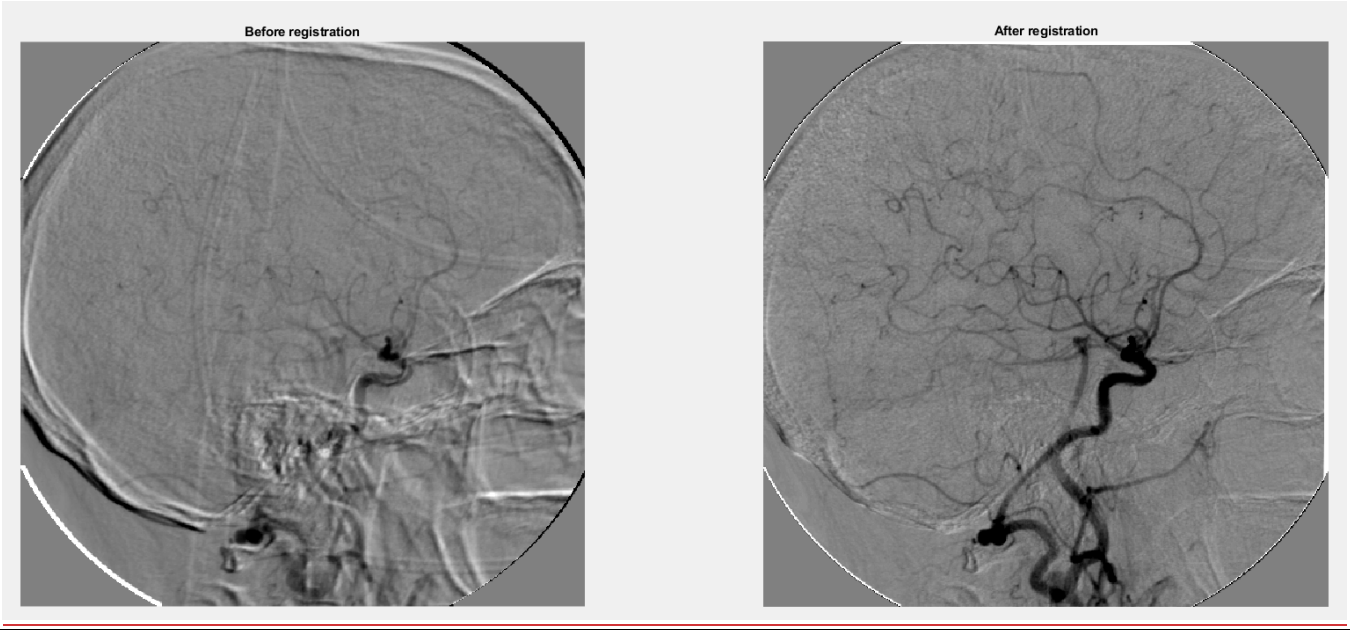
(1) the number of iterations until completion and (2) whether or not there was a successful registration.

5. *Validation of your algorithm with digital subtraction angiography (DSA) images having an unknown displacement*. Use live-new.tiff (containing contrast) and mask-new.tiff (no contrast).
    a. Use your program to register the images.
    b. As the contrast image contains the vessels of interest, it should not be distorted. It will be the reference image. Mask will be the floating image. For display, always subtract the mask_new from live_new, yielding dark arteries, the convention in DSA.
    c. For any evaluation of DSA registration, you must include: 2 original images, the subtracted image without registration, and the subtracted image with registration.
    d. What are the optimal translation parameters?

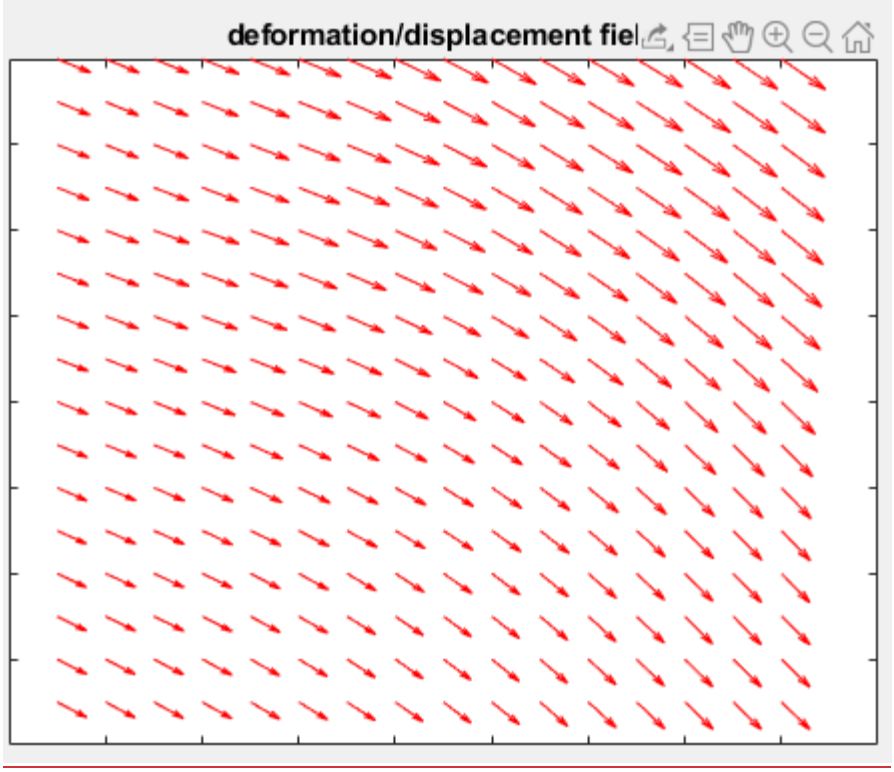
6. The displacement in the DSA image pair (live-new.tif, mask-new.tif) is not entirely captured by simple translation, suggesting the need for non-rigid registration. We will Apply the MATLAB function imregdemons, which uses the famous demons non-rigid registration algorithm (see references below). Use suggestions above for reference and floating. Present your image results. You should use similar window and level for any subtracted images. You should be able to get a result similar to that below obtained by the TA.

Here are some optimizations to try.

1- Iteration array=[5000 400 200], Accumulated Field Smoothing (AFS)= 30
2- Iteration array=[5000 400 200], Accumulated Field Smoothing (AFS)= 3
3- Iteration array=[50 20 5], Accumulated Field Smoothing (AFS)= 30
4- Iteration array=[50 20 5], Accumulated Field Smoothing (AFS)= 3



Visualize the displacement field using quiver plot.



Important functions: interp2(), fminsearch(), imrotate()

Suggested Pseudo code

Objective: register image1 to image2

Main script:

```
% image1 define over x,y
% image2 define over xi,yi
% start at dx=0, dy=0 if only translation is used
p0 = [0 0];
% define options e.g. (doc optimset for details)
options = optimset('Display','iter');
% actual call to optimization
[phat] = fminsearch(@funregister,p0,options,image1,image2,param);
% find the optimal transformation
% … your code here to compute xi and yi from x, y, and phat
% Recompute the final image
image2_new = interp2(x,y,image2,xi,yi)
```

Optimization function:

```
function [error] = funregister(p,image1,image2,param)
% param is a variable that you may want to use to pass parameters or
% variables to the function
% steps here are:
% 1) compute new grid (xi,yi) using p
% 2) interpolate new image: image2_new
% 3) compute similarity measure between image1 and image2_new and
% return it as error
```

Note: There is a well-known problem with **fminsearch** sometimes found at start up. Essentially, **fminsearch** starts to wander around in small steps, e.g., the first simplex might be (0,0), (0.01,0.01), (0,0.01). Which is a translation of only 1/100th of a pixel! Instead, you may want to force it to look around at the pixel level. In that case, you need to multiply the parameters inside "**funregister**" by 100. Don't forget this scaling factor when you use the optimized parameters. Use the options in **fminsearch** to display all the steps, it will be much clearer.

Olivier Salvado, PhD, who took this class many years ago, even created a solution and posted it on MATLAB Central.

https://www.mathworks.com/matlabcentral/fileexchange/5157-fminsearch-modifed-for-higher-scale-smooth-function?s_tid=prof_contriblnk

References

- *DIGITAL IMAGE MATCHING METHOD USING NORMALIZED CROSS-CORRELATION (NCC).*
- *A Comparison of Similarity Measures for Use in 2-D–3-D Medical Image Registration. Graeme P. Penney, Jurgen Weese, John A. Little, Paul Desmedt, Derek L. G. Hill, and David J. Hawkes. IEEE TMI, 17:586-596. 1998.*
- Thirion, J.-P. "Image matching as a diffusion process: an analogy with Maxwell's demons". *Medical Image Analysis*. Vol. 2, Number 3, 1998, pp. 243–260.
- Vercauteren, T., X. Pennec, A. Perchant, N. Ayache, "Diffeomorphic Demons: Efficient Non-parametric Image Registration", *NeuroImage*. Vol. 45, Number 1, Supplement 1, March 2009, pp. 61–72.