Read Me

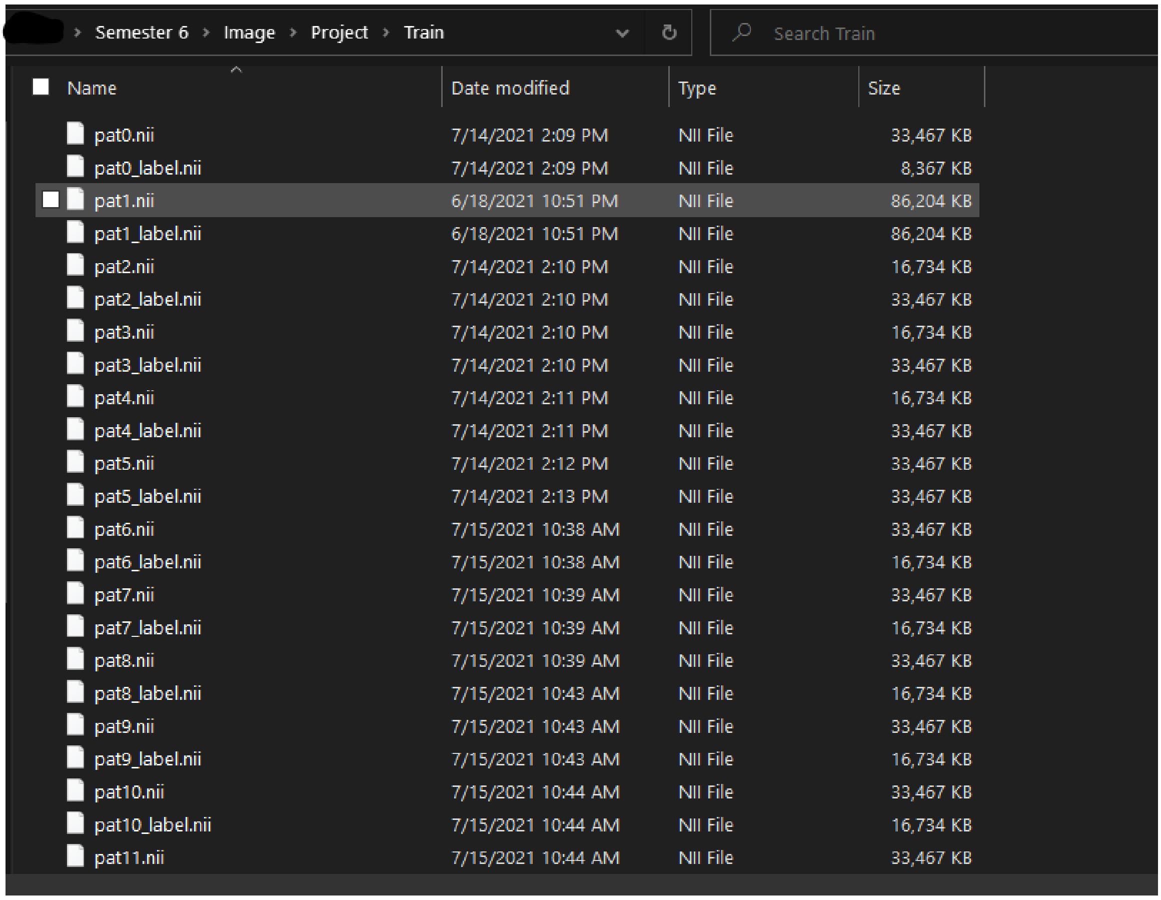
by Ali Ghavampour

1 Preparations

It is essential to follow these steps accurately.

1)
Change the name of the healthy subject data and label to ”pat0.nii” and ”pat0_label.nii”.
”00.nii” → ”pat0.nii”.
”00_mask.nii” → ”pat0_label.nii”.

2)
These folders are available with the uploaded code:
”Functions”
”Train”
Copy all the ”.nii” files to the ”Train” folder. You have to see something like figure 1 in the ”Train” folder.



Name	Date modified	Type	Size
pat0.nii	7/14/2021 2:09 PM	NII File	33,467 KB
pat0_label.nii	7/14/2021 2:09 PM	NII File	8,367 KB
pat1.nii	6/18/2021 10:51 PM	NII File	86,204 KB
pat1_label.nii	6/18/2021 10:51 PM	NII File	86,204 KB
pat2.nii	7/14/2021 2:10 PM	NII File	16,734 KB
pat2_label.nii	7/14/2021 2:10 PM	NII File	33,467 KB
pat3.nii	7/14/2021 2:10 PM	NII File	16,734 KB
pat3_label.nii	7/14/2021 2:10 PM	NII File	33,467 KB
pat4.nii	7/14/2021 2:11 PM	NII File	16,734 KB
pat4_label.nii	7/14/2021 2:11 PM	NII File	33,467 KB
pat5.nii	7/14/2021 2:12 PM	NII File	33,467 KB
pat5_label.nii	7/14/2021 2:13 PM	NII File	33,467 KB
pat6.nii	7/15/2021 10:38 AM	NII File	33,467 KB
pat6_label.nii	7/15/2021 10:38 AM	NII File	16,734 KB
pat7.nii	7/15/2021 10:39 AM	NII File	33,467 KB
pat7_label.nii	7/15/2021 10:39 AM	NII File	16,734 KB
pat8.nii	7/15/2021 10:39 AM	NII File	33,467 KB
pat8_label.nii	7/15/2021 10:43 AM	NII File	16,734 KB
pat9.nii	7/15/2021 10:43 AM	NII File	33,467 KB
pat9_label.nii	7/15/2021 10:43 AM	NII File	16,734 KB
pat10.nii	7/15/2021 10:44 AM	NII File	33,467 KB
pat10_label.nii	7/15/2021 10:44 AM	NII File	16,734 KB
pat11.nii	7/15/2021 10:44 AM	NII File	33,467 KB

Figure 1

Also, do not change the contents of ”Functions” folder.

3)
When files are ready, change the MATLAB curnet folder to the project folder. It should look like this:

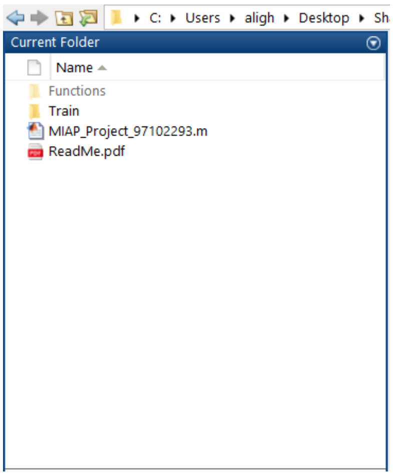


Figure 2

2 Running The Code

First you have to run the very first section of the code with the ”addpath” functions.
Then you can run the sections one by one to see the resuslts. The only variable you have to change in the code is ”fileName”. You can see it in several parts of the code. It is the name of the patient you want to register to the atlas. (’pat0’, ’pat2’, ...)
Note that ’pat1’ and ’pat20’ data are not available.

Multiple methods are implemented in this project. You have to run the sections one by one. Each section may take a while to fully run. Each method has a ”Score” section that evaluates the performance of the registeration.

3 Functions

```
[data,mask,sz] = loadData(fileName)
```

Input: e.g. ’pat9’
Output: MRI data, Segmentation mask and size of the matrices

This function loads the patient data in the workspace. Also, it will modify the value of labels. L1 to L5 will be labeled as 5,6,7,8,9 respectively.

```
verteb = extractVertebra(fileName)
```

Input: e.g. ’pat9’
Output: a cell that holds the segmentation mask of L1 to L5 vertebras.

`[dataSlice,labelSlice,inds] = extractSlice(mask,data))`

Input: segmentation mask and MRI data.
Output: The best slice of each cuts (frontal, axial and latera). *dataSlice* and *labelSlice* are cells that hold the slices. *inds* is the index of slices in each dimension.

`[dataSlice,labelSlice,inds] = extractSlice_ver2(mask,data,inds)`

Input: segmentation *mask* and MRI *data*. *inds* forces the slice indices it is not empty.
Output: The best slice of each cuts (frontal, axial and latera). *dataSlice* and *labelSlice* are cells that hold the slices. *inds* is the index of slices in each dimension.

`[x,y,z,boundary] = outerLayer(mask)`

Input: segmentation *mask*.
Output: *x,y,z* are vertical vectors that contain the coordinates of the boundary of the input mask. *boundary* is the 3D boundary of *mask*.

`labelMat = ReconstructInterpMat(x_tformed,y_tformed,z_tformed,xq,yq,zq,vertebMask)`

Input:
verterbMask: segmentation mask of only L1 to L5. Could be obtained using `extractVertebra()`.
xq,yq,zq: The coordinates of the non zero points in *verterbMask*.
x_tformed,y_tformed,z_tformed: The coordinates of interpolated transformed points. The length of the vectores must equal to *xq,yq,zq*.
Output: transformed segmentation matrix.

`[x_tformed,y_tformed,z_tformed] = cpdInterpolation(x,y,z,regPoints,xq,yq,zq,method)`

Input:
x,y,z: Coordinates of the points in patient mask that used in registeration.
regPoints: The registered points. Length must equal to *x,y,z*.
xq,yq,zq: The points that we want to interpolate the values in.
method: The interpolation mathod.
Output: The interpolated coordinates coresponding to *xq,yq,zq*.

`[patFlag,pat] = isPat(pat)`

Input: e.g. 'pat2'
Output: puts condition on some patients.

This function is not needed after TA provided the corrected data.
`color = colorPC(mask)`

Input: segmentation maks.
Output: a color vector that is sutiable for plotting point cloud. The color are: L1: pink , L2: Yellow , L3: Purple , L4: Green , L5: Orange

```
[d12,d21] = surfd(img1,img2,searchWin,plot)
```

Input: *searchWin* is the search window around each point. *plot* is the plot handle.
Output: caculates the surface distance $d(s, s')$ and $d(s', s)$ between two images.

```
asdScore = asd(img1,img2,searchWin)
```

Input: similar to `surfd()`
Output: caculates the Average Surface Distance measure.

```
hdScore = hd(img1,img2,searchWin)
```

Input: similar to `surfd()`
Output: caculates the Huasdorff Distance of two images.

```
det_J = myJacobian(sx,sy,sz)
```

Input: Displacement field matrices in each dimension x,y,z. Must be 3D matrices.
Output: Calculates the Jacobian determinant of displacement field.

```
jDetDisp(sx,sy,sz)
```

Input: Displacement field matrices in each dimension x,y,z. Must be 3D matrices.
Output: Figures that show the sign of the Jacobian determinant. If negative, red and if positive, blue.

* The idea of the algorithm -ICP then CPD- was developed with cooperation with Maryam Maghsoudi.