About Airlabs

Airlabs is just a framework based on pytorch that handles automatic differentiation and thereby optimization and loss calculation. This also means that airlabs is a deep learning iterative framework where transformation matrix are optimized iteratively for every pair of image separately ulike supervise/unsupervised techniques.
This is in a way akin to what ANTS/FSL does but airlabs uses pytorch instead of other C++ backend used by ANTS/FSL.
Similar to ANTS/FSL this has both rigid/affine and Deformable registration albeit some parts of deformable registration is still under development/enhancement. Just like ANTS/FSL we can configure loss functions like Cross correlation/Means squared error for Intra modal and Mutual Information for Intermodal registration.
Differences between airlabs and ANTS/FSL are also quality of the output and ease of use. ANTS/FSL takes care of most of the preprocessing and additional preprocessing has been nicely packaged so the code is not much. This also means this is more generalizable. Airlabs doesn't provide out of the box preprocessing like Intensity normalization, isotropic voxel resampling, skull stripping, padding etc leaving users to write bulky code using other libraries to take care of these tasks. This also means that the preprocessing pipeline is mostly data dependent and cannot be generalizable.
Quality of registration is not as good as ANTS/FSL and really dependent on preprocessing.

Run process: Command line call looks like below with run screenshots

```
python driver.py --stationary_img_path
"/Users/surajshashidhar/git/image_registration/T1_images/IXI020-Guys-0700-T1.nii.gz"
--moving_img_path
"/Users/surajshashidhar/git/image_registration/T1_images/IXI021-Guys-0703-T1.nii.gz"
--output_warped_image_path
"/Users/surajshashidhar/git/image_registration/airlabs_warped_image.nii.gz" --reorient_flag
"False" --loss_fnc "MSE"
```

```
(image_registration) surajshashidhar@surajs-MacBook-Air:~/git/image_registration/dl_airlabs_ir$ python driver.py --stationary_img_path "/Users/surajshash
idhar/git/image_registration/T1_images/IXI020-Guys-0700-T1.nii.gz" --moving_img_path "/Users/surajshashidhar/git/image_registration/T1_images/IXI021-Guys
-0703-T1.nii.gz" --output_warped_image_path "/Users/surajshashidhar/git/image_registration/airlabs_warped_image.nii.gz" --reorient_flag "False" --loss_fn
c "MSE"
Logging has been disabled

Using device: cpu


 ============ preprocessing started ==================
Logging has been disabled

 ============ ============= ==================
Image 1 voxel resolution before resampling: [0.93749994 0.9375001  1.200002  ]
 ============ ============= ==================
Image 2 voxel resolution before resampling: [0.9374997 0.9375004 1.1999964]
 ============ ============= ==================
Image 1 centre before resampling: [-84.06200408935547, 134.8540802001953, -82.89192199707031]
 ============ ============= ==================
Image 2 centre before resampling: [-86.29121398925781, 144.5850830078125, -87.38084411621094]
 ============ ============= ==================
original t1 affine: [[-2.45350320e-02  5.46792187e-02  1.19754744e+00 -8.40620041e+01]
 [-9.12290633e-01 -2.15646565e-01 -1.44908112e-02  1.34854080e+02]
 [-2.14545935e-01  9.10721242e-01 -7.53313377e-02 -8.28919220e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  1.00000000e+00]]
 ============ ============= ==================
original t2 affine: [[-8.41714721e-03 -2.18238756e-02  1.19962287e+00 -8.62912140e+01]
 [-9.12521243e-01 -2.14561880e-01 -1.68853607e-02  1.44585083e+02]
 [-2.14801878e-01  9.12356257e-01  2.47244034e-02 -8.73808441e+01]
 [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  1.00000000e+00]]
 ============ ============= ==================
original t1 Orientation: ('P', 'S', 'R')
 ============ ============= ==================
original t2 Orientation: ('P', 'S', 'R')
 ============ ============= ==================
 ============ ============= ==================
Not reorienting the images as reorient flag is false
 ============ ============= ==================
```

```
 ============ ============= ==================
Chosen resampling method: both_moving_and_stationary
 ============ ============= ==================
Shape of resampled 1 image: (188, 291, 299)
 ============ ============= ==================
resampled t1 affine: [[   1.            0.            0.          -90.31843725]
 [   0.            1.            0.         -154.78412796]
 [   0.            0.            1.         -148.07219147]
 [   0.            0.            0.            1.        ]]
 ============ ============= ==================
Shape of resampled 1 image: (188, 291, 293)
 ============ ============= ==================
resampled 2 affine: [[   1.            0.            0.          -94.00267481]
 [   0.            1.            0.         -145.33703203]
 [   0.            0.            1.         -142.15532294]
 [   0.            0.            0.            1.        ]]
 ============ ============= ==================
 ============ ============= ==================
Image 1 voxel resolution after resampling: [1. 1. 1.]
 ============ ============= ==================
Image 2 voxel resolution after resampling: [1. 1. 1.]
 ============ ============= ==================
Image 1 centre after resampling: [-90.31843566894531, -154.7841339111328, -148.0721893310547]
 ============ ============= ==================
Image 2 centre after resampling: [-94.00267791748047, -145.3370361328125, -142.1553192138672]
padding all tensors to their max values per dimensions
 ============ ============= ==================
torch.Size([188, 291, 299])
 ============ ============= ==================
torch.Size([188, 291, 299])
 ============ ============= ==================
torch.FloatTensor
 ============ ============= ==================
torch.FloatTensor
 ============ ============= ==================

 ============ preprocessing completed ==================
```

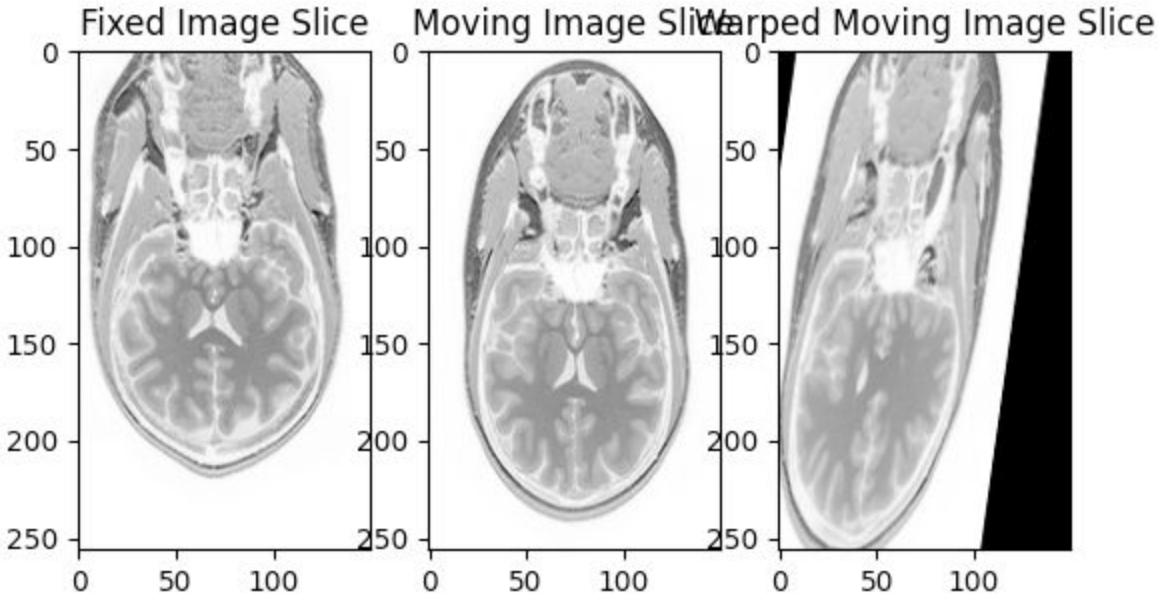
```
 ============= starting registration ====================
Logging has been disabled

 ============= fixed image size, spacing, origin and datatype ====================
[188, 291, 299]
[1. 1. 1.]
[-90.31843566894531, -154.7841339111328, -148.0721893310547]
torch.float32
 ============= moving image size, spacing, origin and datatype ====================
[188, 291, 299]
[1. 1. 1.]
[-94.00267791748047, -145.3370361328125, -142.1553192138672]
torch.float32
 ============= ============== ====================
Using Affine transformation
 ============= ============== ====================
Using Mean squared error loss
0 /Users/surajshashidhar/anaconda3/envs/image_registration/lib/python3.7/site-packages/torch/nn/functional.py:3384: UserWarning: Default grid_sample and
affine_grid behavior has changed to align_corners=False since 1.3.0. Please specify align_corners=True if the old behavior is desired. See the documentat
ion of grid_sample for details.
  warnings.warn("Default grid_sample and affine_grid behavior has changed "
mse: 128241.8984375
1 mse: 191701.53125
2 mse: 185836.421875
3 mse: 190884.5
4 mse: 166942.796875
5 mse: 172028.375
6 mse: 179323.46875
7 mse: 175069.015625
8 mse: 177310.890625
9 mse: 172867.03125
 ============= ============== ====================
Registration done in:  48.05361533164978  s
Result parameters:
_phi_z 0.32397663593292236
_t_x 0.06890081614255905
_t_y 0.07056795805692673
_center_mass_x -0.0985608845949173
```

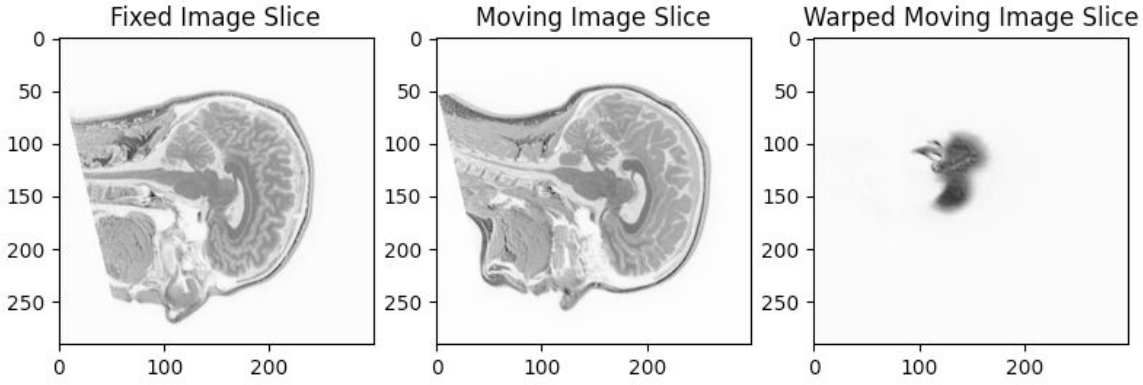
```
_center_mass_x -0.0985608845949173
_center_mass_y -0.07470966875553131
_t_z -0.0028387438505887985
_phi_x 0.17361512780189514
_phi_y 0.001196743920445442
_center_mass_z -0.18368257582187653
_scale_x 1.009875774383545
_scale_y 1.1781355142593384
_scale_z 0.8974235653877258
_shear_y_x -0.13136659562587738
_shear_x_y -0.017857957631349564
_shear_z_x 0.02051101252436638
_shear_z_y -0.05952554568648338
_shear_x_z -0.0505535751581192
_shear_y_z 0.08986826241016388
 ============= ============== ====================
tensor([[ 0.9614, -0.4904,  0.0920,  0.0925],
        [ 0.3092,  1.0447, -0.2055,  0.0745],
        [-0.0495,  0.2831,  0.8718,  0.0044]], grad_fn=<SliceBackward>)
 ============= ============== ====================
objc[1689]: Class FIFinderSyncExtensionHost is implemented in both /System/Library/PrivateFrameworks/FinderKit.framework/Versions/A/FinderKit (0x7fff9476
e3d8) and /System/Library/PrivateFrameworks/FileProvider.framework/OverrideBundles/FinderSyncCollaborationFileProviderOverride.bundle/Contents/MacOS/Find
erSyncCollaborationFileProviderOverride (0x12b770f50). One of the two will be used. Which one is undefined.
 ============= registration ended====================

 ============= starting post processing and saving warped image to disk ====================

=============== transformation matrix before ==============================
tensor([[ 0.9614, -0.4904,  0.0920,  0.0925],
        [ 0.3092,  1.0447, -0.2055,  0.0745],
        [-0.0495,  0.2831,  0.8718,  0.0044]], grad_fn=<SliceBackward>)
============== transformation matrix after ==================
[[ 0.96138555 -0.49041435  0.09196466  0.09245307]
 [ 0.30924863  1.04472423 -0.20551527  0.07449637]
 [-0.04952885  0.28311217  0.8717925   0.00444113]
 [ 0.          0.          0.          1.        ]]
============== ========================== ==================
Saving file to : /Users/surajshashidhar/git/image_registration/airlabs_warped_image.nii.gz
```



Bad quality registration for inappropriately preprocessed images in airlabs



About Code

Code has been packaged into 4 pieces so that preprocessing pipeline is somewhat configurable. We have implemented 3-D Affine registration for airlabs

Image Registration

Image registration activities are put into img_registration.py

Image registration consists of registration type, optimizers and loss functions and just does image registration. Currently we have only 3-D affine registration, but loss functions are configurable hence we can use them for both intra and inter modal registration.

```
class ImageRegistrationUtils:
    def __init__(self, preprocessed_stationary_img_tnsr, preprocessed_moving_img_tnsr, preprocessed_stationary_img_voxel_dim,
    preprocessed_moving_img_voxel_dim, preprocessed_stationary_img_centre, preprocessed_moving_img_centre,
    img_shape, device, loss_fnc = const.LOSS_FNC, logging_flag=const.LOGGING_FLAG, log = None):

    def three_dim_affine_reg(self):
```

Image Processing

File processing and preprocessing activities are put into img_processing.py file, more preprocessing can be added here without much changes to other files.

Current preprocessing pipeline consists of reading nifti files, reorienting images to standard Right Anterior Superior (RAS) orientation, resampling images to 1mm isotropic or resampling moving image to match that of stationary image, padding to get images of same dimension, saving warped image into nifti files. The name of the methods reflect the above steps. More preprocessing can be added if required. Each step of preprocessing can be turned on/off by a flag and some parameters such as resampling voxel sizes can be changed.

```
class ImageprocessingUtils:

    def __init__(self, stationary_image_file_path, moving_image_file_path,
    output_warped_image_file_path, reorient_flag, resample_flag, resampling_type,  resampling_size,
    padding_flag, logging_flag, log = None):

    def read_input_images(self):


    def reorient_images(self):


    def resample_image(self):


    def convert_nifti_to_tensor(self):


    def convert_tensor_to_nifti(self, warped_img_tnsr, transformation, displacement):


    def save_warped_image(self, warped_nifti_img):
```

Parametrization

Application constants and default values to be used by the application are parameterized in dl_airlabs_constants.py file. This consists of default values of application and command line parameters if not given

```
# File path and name constants
FILE_PATH_STATIONARY_IMG = "/Users/surajshashidhar/git/image_registration/T1_images/IXI002-Guys-0828-T1.nii.gz"
FILE_PATH_MOVING_IMG = "/Users/surajshashidhar/git/image_registration/T2_images/IXI002-Guys-0828-T2.nii.gz"
FILE_PATH_WARPED_IMG = "/Users/surajshashidhar/git/image_registration/warped_image.nii.gz"
FILE_PATH_LOG = os.path.join(os.getcwd(), 'AIRLABS_Image_Registration_log.log')
FILE_MODE = "w"
FILEPATH_AIRLABS_LIB = '/Users/surajshashidhar/git/airlab'


LOGGING_LEVEL = 20
RESAMPLING_SIZE = [1, 1, 1]
LOSS_FNC = "MSE"
REORIENT_FLAG=True
RESAMPLE_FLAG=True
RESAMPLING_TYPE="both_moving_and_stationary"
PADDING_FLAG=True
LOGGING_FLAG=False
```

Main Driver

The main driver file driver.py takes in command line parameters such as image paths, preprocessing types and orchestrates the registration process. If parameters are not passed from command line application would take default parameters from constants file.

```
class Driver:
    def __init__(self, stationary_img_path, moving_img_path, output_warped_image_path, ⋯

    def start_process(self): ⋯


if __name__ == "__main__":
    ap = argparse.ArgumentParser()

    # Add the arguments to the parser and parse the arguments from command line
    ap.add_argument( "--stationary_img_path", nargs= "?", required=False, help=" stationary_img_path", default = cons
    ap.add_argument("--moving_img_path", nargs= "?", required=False, help="moving_img_path", default = const.FILE_PAT
    ap.add_argument("--output_warped_image_path", nargs= "?", required=False, help=" output_warped_image_path", defau
    ap.add_argument("--loss_fnc", nargs= "?", required=False, help="loss_fnc", default = const.LOSS_FNC)
    ap.add_argument("--resampling_size", nargs= "?", required=False, help="resampling_dim", default = const.RESAMPLIN
    ap.add_argument("--logging_flag", nargs= "?", required=False, help="logging_flag", default = "False")
    ap.add_argument("--padding_flag", nargs= "?", required=False, help="padding_flag", default = "True")
    ap.add_argument("--reorient_flag", nargs= "?", required=False, help="reorient_flag", default = "True")
    ap.add_argument("--resample_flag", nargs= "?", required=False, help="resample_flag", default = "True")
    ap.add_argument("--resampling_type", nargs= "?", required=False, help="resampling_type", default = const.RESAMPLI
```