Richard Guo
Neuro Data Design II

ANTS Image Registration

- Image Normalization software package built on the ITK framework

Linear Registration:

Overview:

1. Transformation models – kernels used to regularize image
2. Similarity (correspondence) measures – matching objects together based on how "similar they are to each other"
3. Optimization – fastest way to register images (optimal transformations needed for registration)

Linear Transformations:

- Rigid or Affine Transformations
    - Rigid (translation, rotation)
    - Affine (translation, rotation, scale, shear)
- These linear transformations within ANTS are optimized with
    - MSQ - mean squared difference (I believe our current pipeline is using this method)
    - MI – mutual information
    - Similarity metrics are optimized with respect to a translation, rotation, shear, or scale
        - This successive optimization for each part of the linear transformation allows for control over degrees of freedom

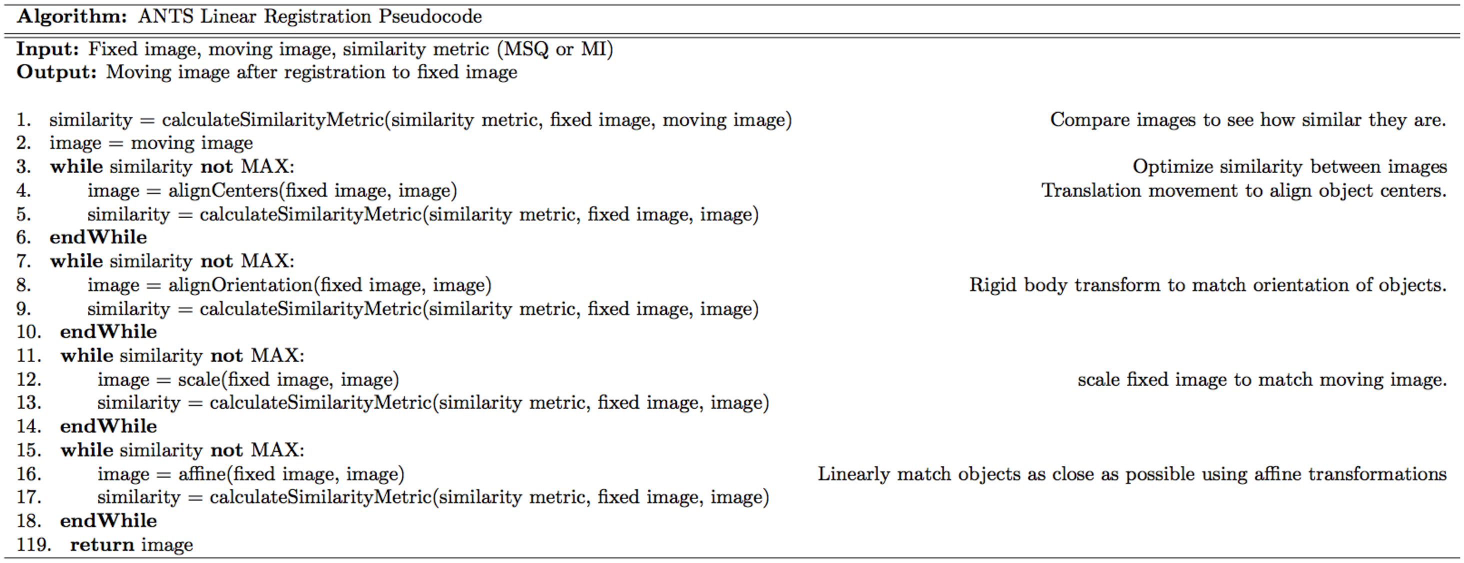
Similarity Metrics for Linear Transformations:

1. Mean Squared Distance – self-explanatory – the object closest to the original is registered (current method in our pipeline)

2. Mutual Information (http://people.csail.mit.edu/sw/papers/IJCV-97.pdf)

    a. Intensity based – compare intensity patterns in images
        i. Doesn't look at image features such as points, lines, and contours.
        ii. Compares via correlation metrics?
        iii. Based off of pixel intensities
    b. Pros: more robust than normal correlation (paper did not specify what was normal)
    c. Compares similarity based off of entropy of pixels (grayness)

Pseudocode:



```
Algorithm: ANTS Linear Registration Pseudocode

Input: Fixed image, moving image, similarity metric (MSQ or MI)
Output: Moving image after registration to fixed image

similarity = calculateSimilarityMetric(similarity metric, fixed image, moving image)       Compare images to see how similar they are.
image = moving image
while similarity not MAX:                                                                   Optimize similarity between images
    image = alignCenters(fixed image, image)                                                Translation movement to align object centers.
    similarity = calculateSimilarityMetric(similarity metric, fixed image, image)
endWhile
while similarity not MAX:
    image = alignOrientation(fixed image, image)                                            Rigid body transform to match orientation of objects.
    similarity = calculateSimilarityMetric(similarity metric, fixed image, image)
endWhile
while similarity not MAX:
    image = scale(fixed image, image)                                                       scale fixed image to match moving image.
    similarity = calculateSimilarityMetric(similarity metric, fixed image, image)
endWhile
while similarity not MAX:
    image = affine(fixed image, image)                              Linearly match objects as close as possible using affine transformations
    similarity = calculateSimilarityMetric(similarity metric, fixed image, image)
endWhile
return image
```

Note: The (**while** similarity **not** MAX) is actually gradient descent on a function based off of the similarity metric (MSQ or MI) of the two images.