Lecture 18
Matrix Mappings and Transforms

MP574: Applications

Sean B. Fain, PhD (sfain@wisc.edu)

Diego Hernando, PhD (dhernando@wisc.edu)

ITK/VTK Applications: Andrew Hahn, PhD (adhahn@wisc.edu)

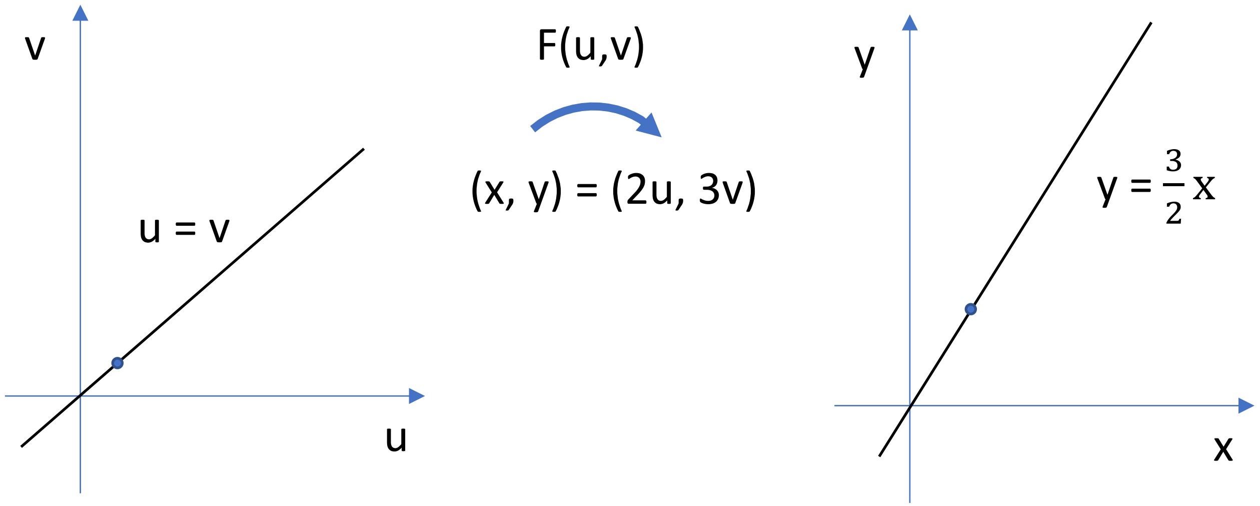
Learning Objectives

- Introduce the 2nd module of the course, lecture schedule, in-class activities and course materials.
- Provide a rationale for bridging the optimization content in Module I with Module 2.

Outline

- Transformations (Mappings)
  - Rigid
  - Affine
  - Non-rigid
  - Deformable
- Examples:
  - Control Point Registration
  - Fiducial or “Feature-Based” Registration
- Observations:
  - Interpolation error
  - Local vs. global feature selection

Example: Linear Mapping



$$\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 2u \\ 3v \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}$$

In general, $F(u, v) \rightarrow \big(f(u, v), g(u, v)\big)$ and for $f(u, v) = au + bv;\ g(u, v) = cu + dv$, then:

$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} au + bv \\ cu + dv \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}$; system of linear equations with a solution if $ad - bc \neq 0$.

$$T\vec{\boldsymbol{u}} = \vec{\boldsymbol{x}}$$

Linear Mapping and the Jacobian

In general, $F(u,v) \rightarrow \big(f(u,v), g(u,v)\big)$ and the Jacobian, J of the 2D Mapping is:

J = $abs\begin{vmatrix} f_u & f_v \\ g_u & g_v \end{vmatrix}$, where f_u=$\frac{\partial f}{\partial u}$ etc…

Example: For a specific matrix formulation of a linear mapping, the magnification of the mapping will be the same for all locations throughout the uv-plane:

$\begin{bmatrix} 2 & 3 \\ 1 & 2 \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 2u+3v \\ u+2v \end{bmatrix}$; J = $|4uv - 3uv| = uv.$

Homogeneous Coordinate Vector: Accommodating Translation

Mapping all points by adding a constant value to their coordinates.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

x’ (y’) is not a linear combination of x and y

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

2D Transforms

Rotation, scaling, shearing

$$T = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

Translation

Projective

Rotation:

Note rotation about (0,0) and z-axis is presumed. This form of the rotation matrix will perform a clockwise rotation for positive theta.

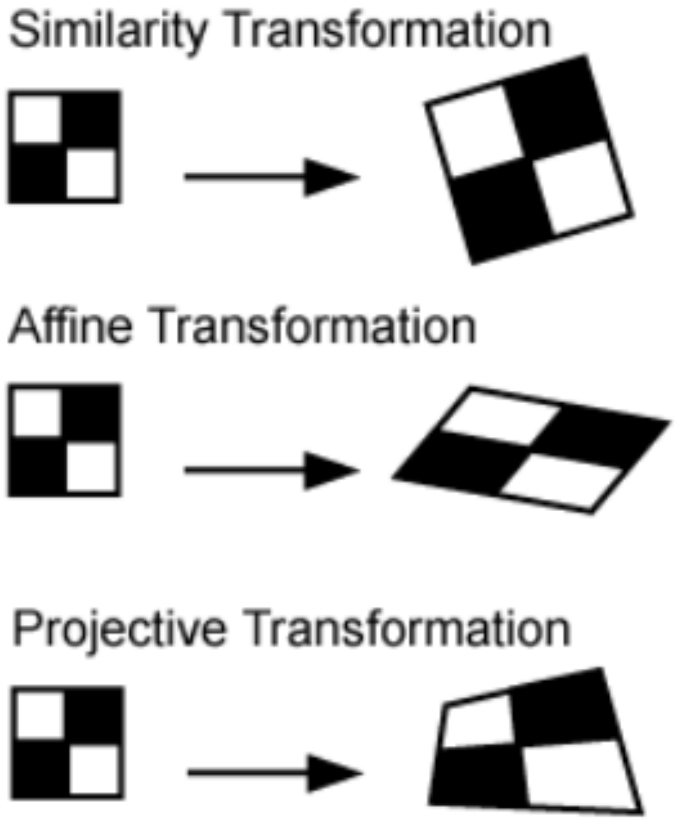
$$R = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Scale and Shear

$$S = \begin{bmatrix} \alpha & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$H_x = \begin{bmatrix} 1 & \tan\phi_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad H_y = \begin{bmatrix} 1 & 0 & 0 \\ \tan\phi_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Composite Transformations



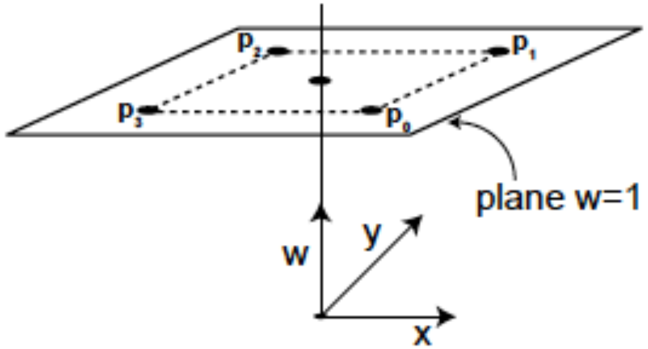
Projective Transformation

The projective transformation does not preserve parallelism.
Squares will be transformed into a quadrilateral.

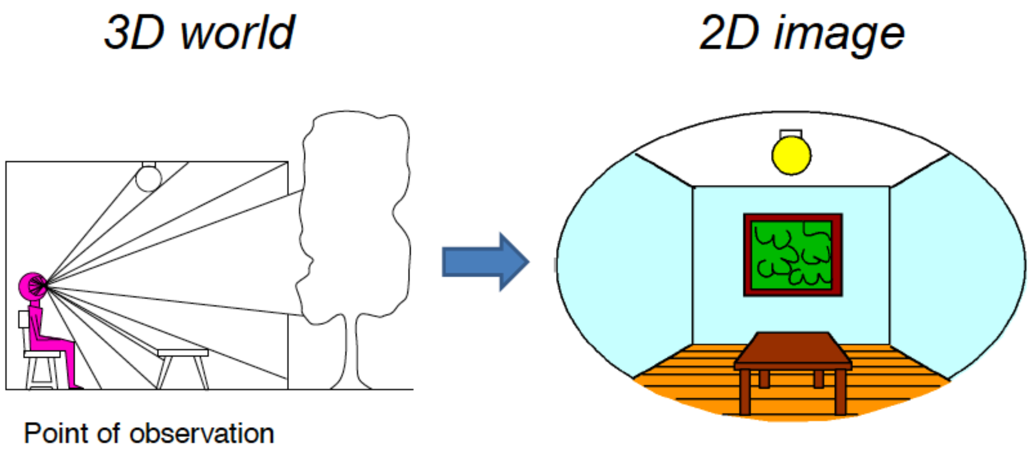
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x' = \frac{ax + by + c}{gx + hy + i} \qquad y' = \frac{dx + ey + f}{gx + hy + i}$$

- Homogenous coordinate system places a plane at w = 1.
- Can accommodate “back-projections” from infinity.
- Our operations in 2D are on the plane at w=1.



Projective Transforms



- Projective mappings are a type of transitional non-linear transform
- Most commonly for computer graphics applications
- Portray 3D perspective on a 2D display

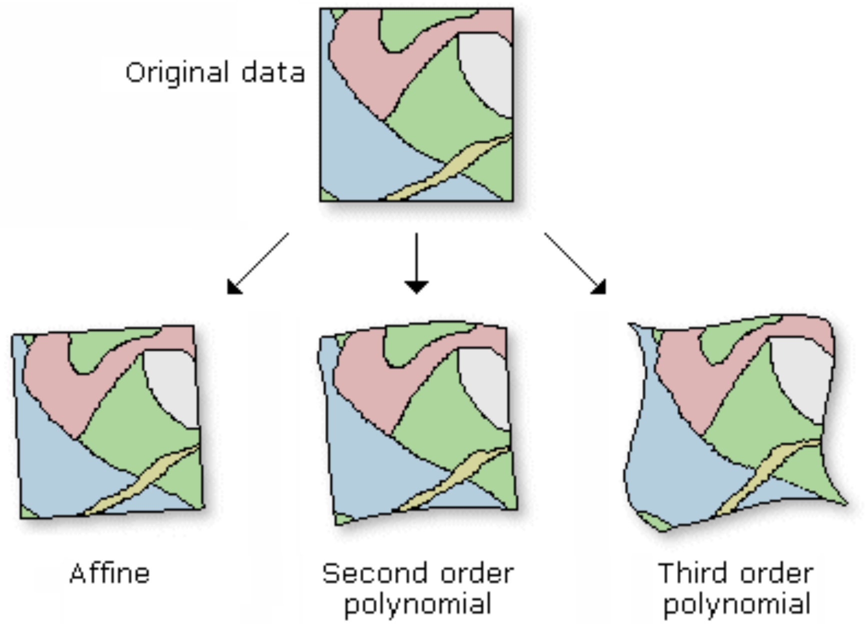
Courtesy Stephen Palmer

Deformable Example: Polynomial Transformation

- Mapping based on given mapping functions

$$x' = \sum_{i=0}^{I} \sum_{j=0}^{J} a_{ij} x^i y^j$$

$$y' = \sum_{i=0}^{I} \sum_{j=0}^{J} b_{ij} x^i y^j$$



- The higher the transformation order, the more complex the distortion.

3D Template Matrix in Homogenous Coordinates

Rotation, scaling, shearing

$$T = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix}$$

Translation

Projective

$$\text{Translate:} \begin{bmatrix} 1 & 0 & 0 & \Delta x \\ 0 & 1 & 0 & \Delta y \\ 0 & 0 & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \text{Scale:} \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\text{Shear:} \begin{bmatrix} 1 & h_{xy} & h_{xz} & 0 \\ h_{yx} & 1 & h_{yz} & 0 \\ h_{zx} & h_{zy} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

3D Extension: Rotation

- 3D Rotation allows rotation about x,y, or z axes.

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_x & \sin\theta_x & 0 \\ 0 & -\sin\theta_x & \cos\theta_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

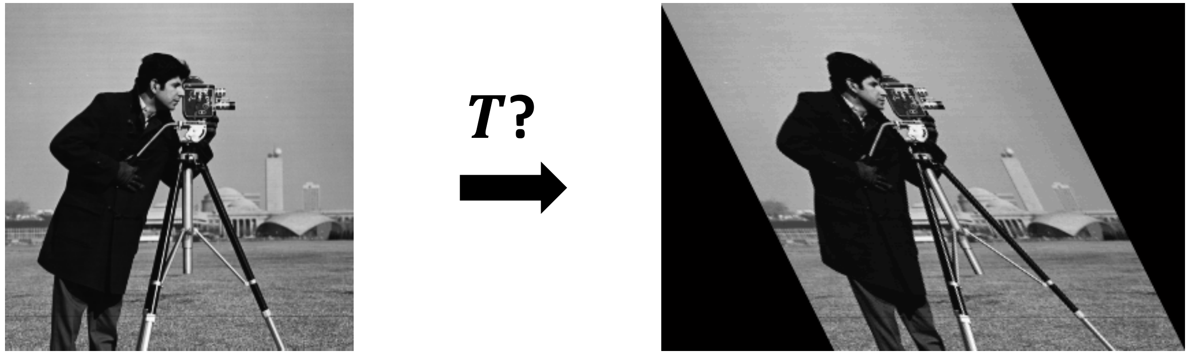
$$R_y = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_z = \begin{bmatrix} \cos\theta_z & \sin\theta_z & 0 & 0 \\ -\sin\theta_z & \cos\theta_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- For 3D rotations, the order specified will lead to different results.
- Need to specify the rotation angles (Euler angles) and order of rotation
- Depending on dimensionality, the order of affine operations are associative but not, in general, commutative.

Image Registration

Image registration is the process of geometric alignment of one image with another



Estimating the optimal spatial transformation which makes an image similar to another.

Feature-based Registration

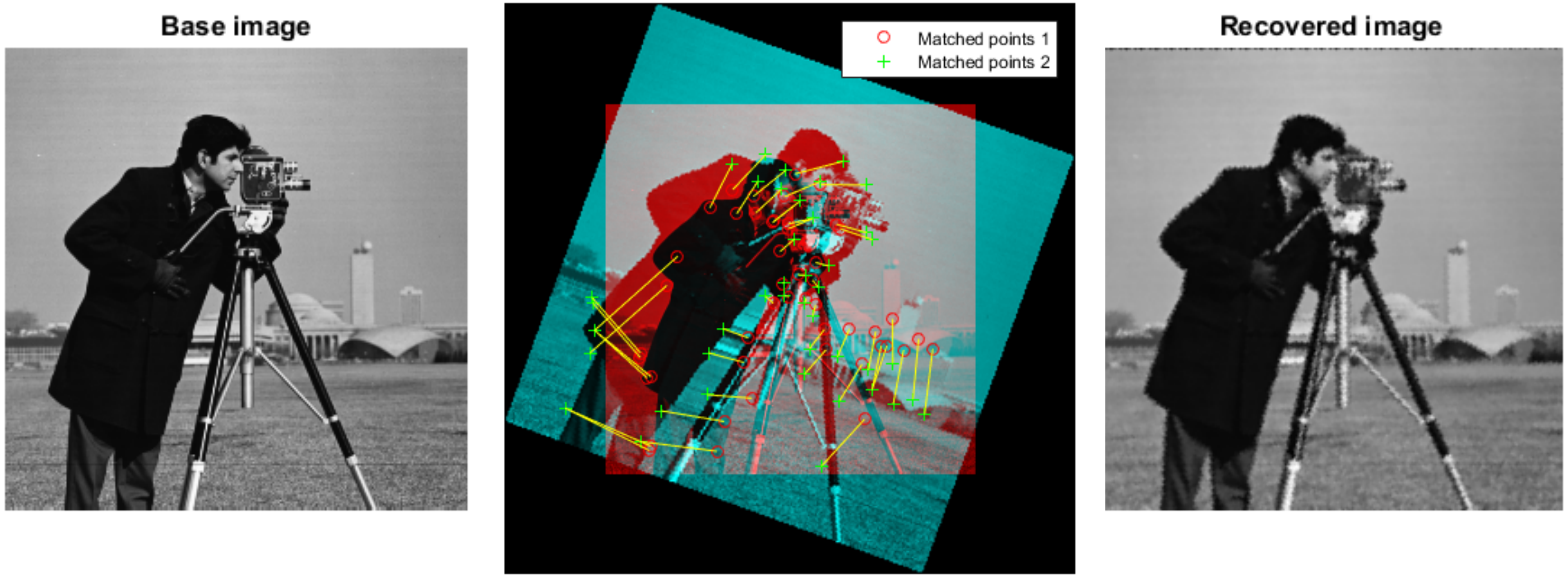

```
points1 = detectSURFFeatures(I1);
[f1, vpts1] = extractFeatures(I1, points1);
indexPairs = matchFeatures(f1, f2);
```

Feature-based Registration

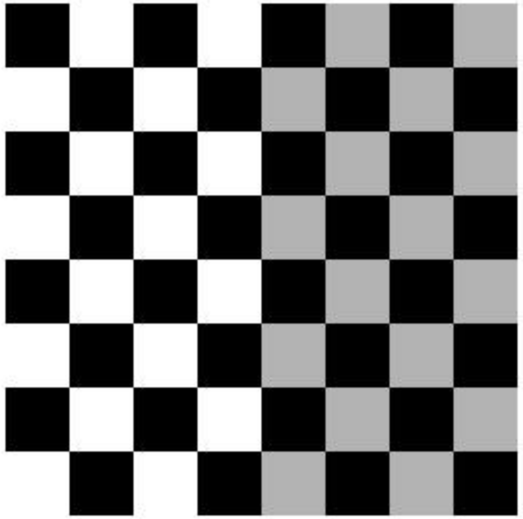
Base image

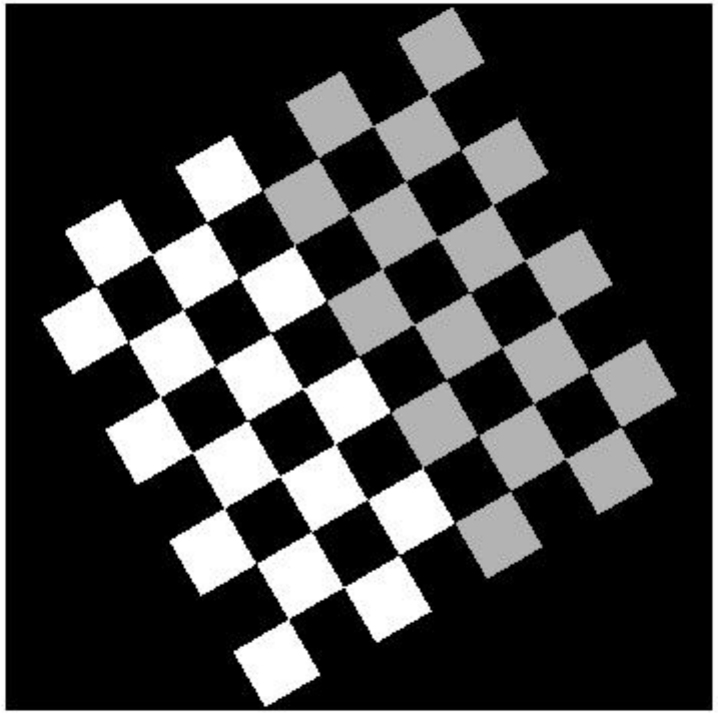
Recovered image

```
tform = estimateGeometricTransform(matched1, matched2, 'similarity');
```


Base image
Matched points 1
Matched points 2
Recovered image

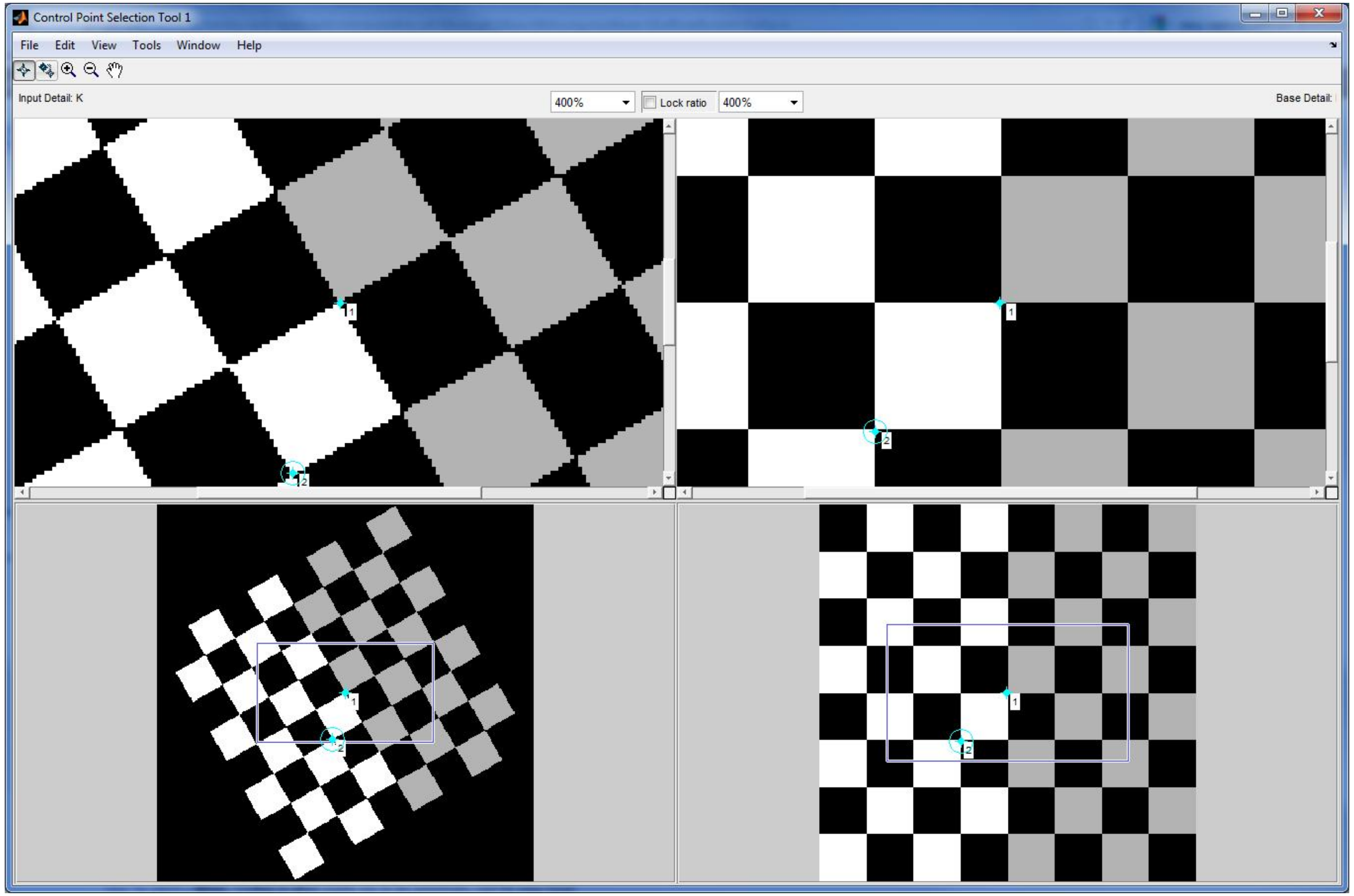
Control Point Registration





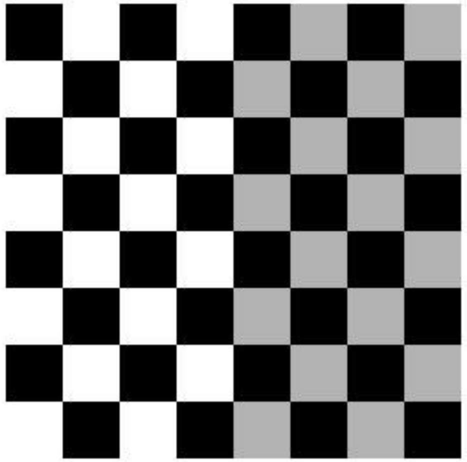
Rotated by 30^o

Control Point Placement



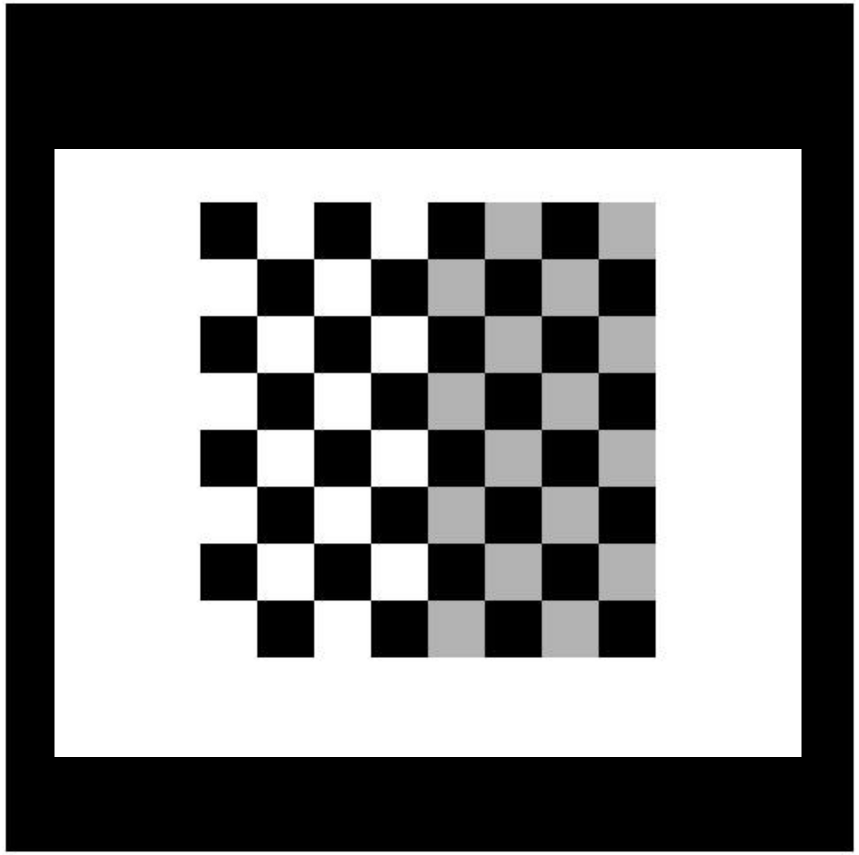
[movingPoints,fixedPoints] = cpselect(moving,fixed)

Registration Result



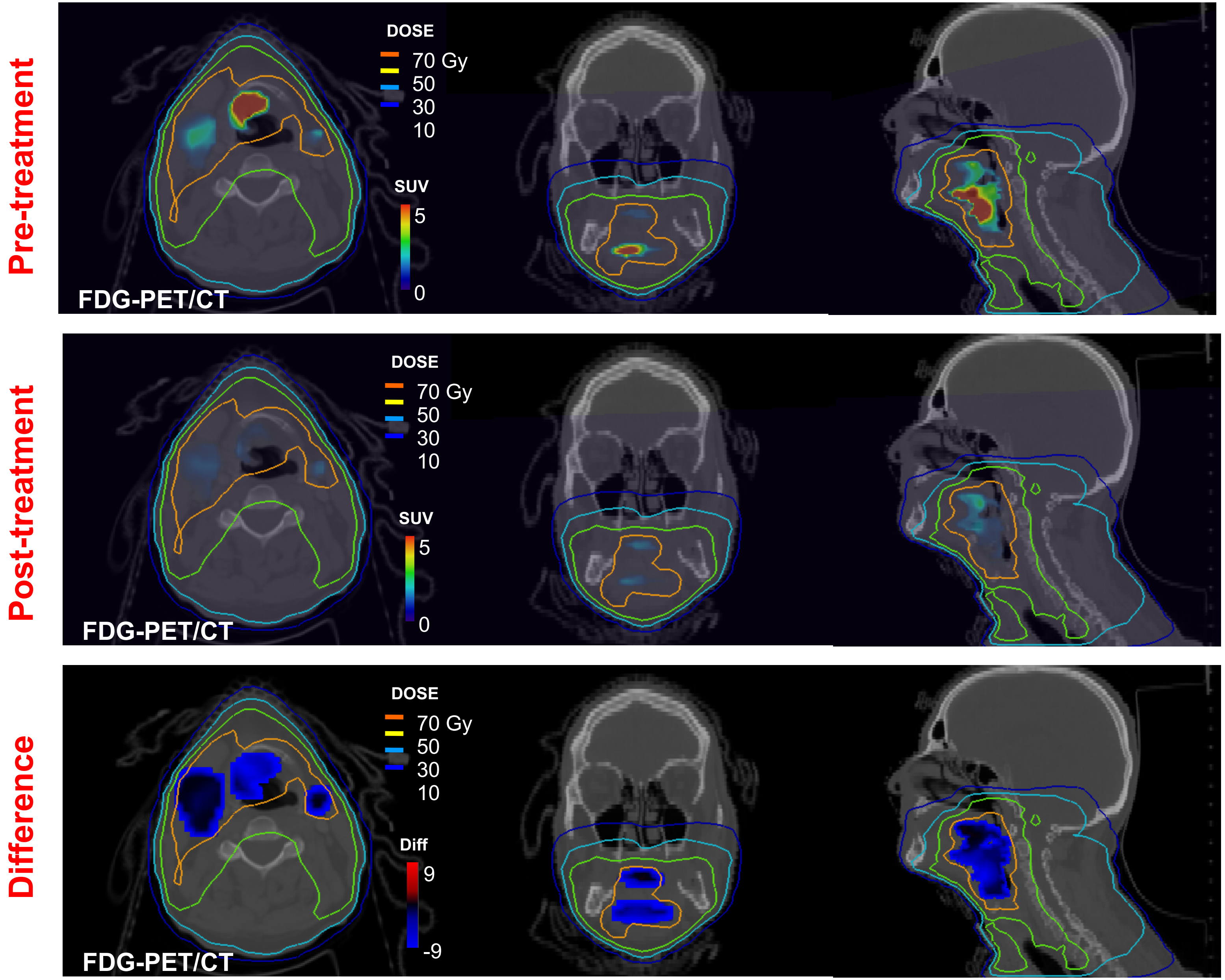
scale = 0.986

angle = 28.5°



tform = fitgeotrans(movingPoints,fixedPoints,transformationType)

Longitudinal Deformable Registration:



Affine Transform Degrees of Freedom

- Degrees of freedom 2D (3D)

  2(3) + 1(3) + 2(3) + 1(1) + 1(1) (+ $H_z(\phi)$)

  = 7 (12).

$$I(x', y') = T(\Delta x, \Delta y) \cdot R(\theta) \cdot S(\alpha, \beta) \cdot H_x(\phi) \cdot H_y(\phi) \cdot I(x, y)$$

$$Generalize,$$

$$T_k = T(\Delta x, \Delta y) \cdot T_R(\theta) \cdot T_S(\alpha, \beta) \cdot T_{sx}(\phi) \cdot T_{sy}(\phi) = \prod_{i=1}^{Q} T_i$$

- Matrix multiplication is not commutative

Summary

- The affine transform is a powerful tool for linear mapping of on coordinate system to another
- Provides a strong framework for basic image registration tasks under specific geometric constraints.
- Limitations in medical image registration are apparent in circumstances of progressive disease and physiological motion.
- To address these challenges, deformable methods are needed.
  - The degree of freedom must be traded off against computational complexity and stability of the solution.
  - This tradeoff, as well as different deformable registration approaches will be discussed more in Lecture 19.