MP574 Lecture 18:

Learning Objectives

- Introduce linear mappings and matrix transforms.
- Define basic operations of the affine transform matrix.
  - Implementation and interpolation considerations (Matlab script: Affine_transform.m)
- Basic image registration
  - Supervised fiducial or control-point image registration (Matlab script: Registration_script.m)
- Non-linear transforms
  - Projective
  - Polynomials and B-spline

Linear Mappings

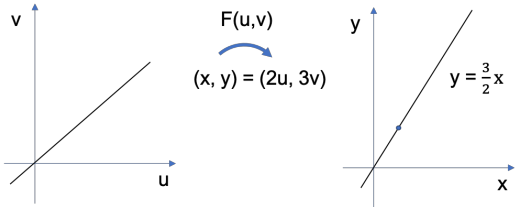


Figure 1: A specific example of a linear mapping taking points a line to a new line the transform coordinate system. The matrix form of this mapping is Eq. 1.

- What is a mapping?

Mathematically a mapping is a function that operates on a space to transform one coordinate system into another. From the perspective of image registration some unknown mapping has taken place between one image position or space (time), we will term this the "fixed" image, and another image position or space (time), we will term this the "moving" image. Alternatively, these image domains are called the "source" image and "target" image, respectively.

- Example: Linear transform

Consider points P(u, v) on that fall on the line u=v, and some mapping of this line in u-v space, F(u,v) = (2u, 3v), to x-y coordinate space (Figure 1).

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:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} au + bv \\ cu + dv \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}. \qquad [1]$$

This is a system of linear equations with a unique solution if $ad - bc \neq 0$.

$$T\vec{\boldsymbol{u}} = \vec{\boldsymbol{x}} \qquad [2]$$

Linear Matrix Transformations are represented by the matrix T in Eq. 2. Each linear transform has different degrees of freedom which designates the time of mapping as "rigid" or "affine."

MP574 Lecture 18:

The 2D affine matrix transform expands the basic 2D matrix mapping in Eq. 1 to a 3 X 3 matrix using the "homogeneous" coordinate vector (Figure 2a). This is a convention that accommodates translation, which is not strictly a linear transform since the resulting new coordinates are not linear combinations of the u,v. We will use this convention in class. The coordinates (u, v)->(x,y) will refer to continuous mappings, while the coordinates (x, y) -> (x',y') will refer to mapping of discrete pixel coordinates. Let me know if you find this too confusing as I can change the convention for future classes.

The basic structure of the 2D affine (exclusive of projective elements) transform matrix using the homogeneous coordinate system has linear operations represented by elements in the upper left 2 X 2 corner, translation by elements (1,3) and (2,3) as shown in Figure 2b. The affine transform maps coordinates on a line in one space to lines in the transform space. Moreover, parallel lines will remain parallel after transformation and magnification will be the same for all coordinates in the fixed and moving coordinate spaces. Note that the projective transform terms (which represent non-linear transform elements) are represented in the third row.

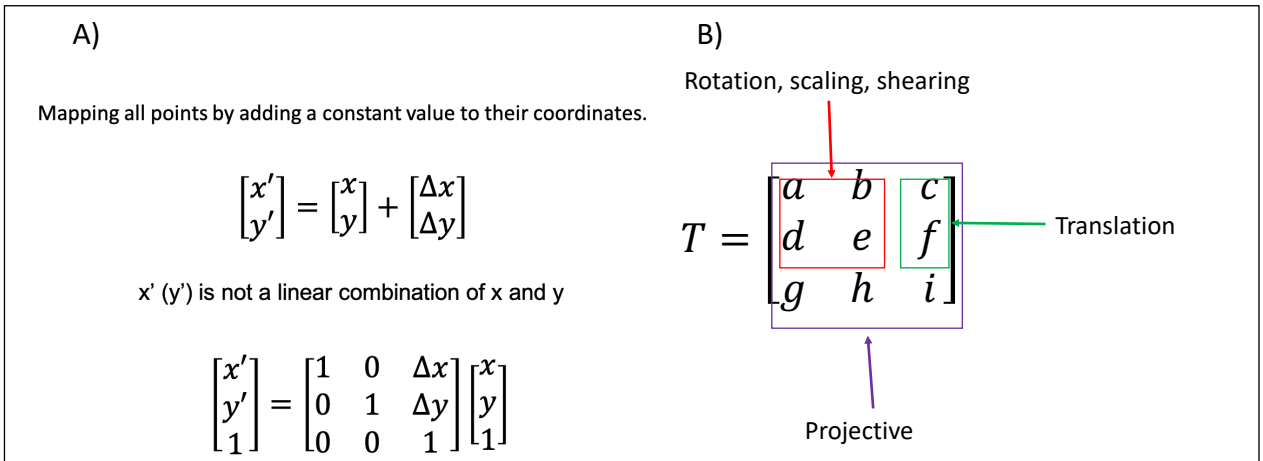


Figure 2: A) The homogeneous coordinate vector allow translation to be easily incorporated in the affine transform matrix. B) The structure of the transform matrix and the operations represented by the various matrix elements. Projective transform elements represent a non-linearmapping and are not considered part of the affine transform.

[Insert Affine transformation mathematics here Example calculations of 2D translation, rotation, shear, scaling. Extension to 3D matrix formulation and ambiguity of order of operations. Introduce degrees of freedom.]…

The four affine transformations are accomplished by matrix multiplications on the coordinate or point, P. The matrix operators are translation (T), rotation (R), scaling or magnification (S), and shear (H). These operators can be applied singly or in combination by matrix multiplication to generate a single affine transform matrix, **A**:

$$A\vec{\boldsymbol{u}} = \vec{\boldsymbol{x}},$$

where x is a column vector:

MP574 Lecture 18:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The affine transform operations and their matrix formulation are as follows:

Translation:

$$T(\Delta x, \Delta y) = \begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation:

$$R(\theta) = \begin{bmatrix} \cos(\theta) & \mp\sin(\theta) & 0 \\ \pm\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Scaling:

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

Shearing in the x direction:

$$H_x(\phi) = \begin{bmatrix} 1 & \tan(\phi) & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

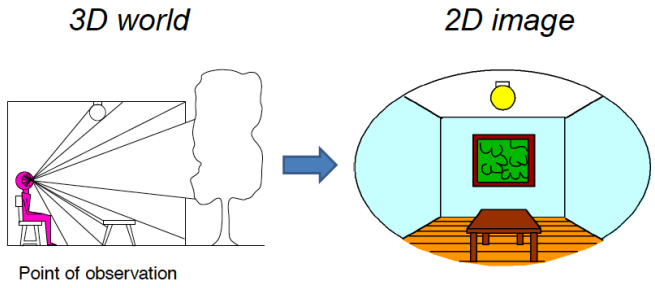
Shearing in the y direction:

$$H_y(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ \tan(\phi) & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

In the notation for rotation, the default convention is for rotation to be anti-clockwise about the upper left element of the matrix. However, in some formulations (such as Matlab) the center of rotation is about the center of the image, and the rotation is clockwise. The sense of the rotation can be easily implemented by changing the signs on the sin(θ) elements of the rotation matrix. In the notation for shearing, the angle ϕ is the angle of shear opening, a positive shear being clockwise for shear in x and anti-clockwise for shear in y.
Note also that the transform matrix operates on points or coordinates, not signal intensities. The signal intensity associated with a given coordinate or pixel in the image must be interpolated to the transformed coordinate. In practice, the interpolation is performed based on the fixed coordinate space so that signal values can be assigned to conserve the original signal intensities.

MP574 Lecture 18:

A)

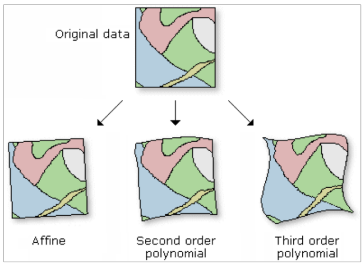


B)

- 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.

Figure 3: A) Projective transforms are commonly used in computer graphics applications to portray 3D perspective on 2D display. B) Deformable transforms are used commonly in medical image registration. One common class of deformable transforms use interpolation theory with higher order splines or polynomial functions depending on the desired degrees of freedom.

Non-linear Transforms

Non-linear mappings are also increasingly used but are fundamentally different in their form from linear approaches. Projective mappings are a type of transitional non-linear transform that can be accommodated using the conventional matrix formulation (Figure 3a), but projection mappings are a special case used most commonly for computer graphics application to portray 3D perspective on a 2D display. One class of deformable registration methods that is commonly used for medical image registration is built on interpolation theory, including transforms using higher order polynomials (Figure 3b) and splines. The order depends on the degree of deformation required. 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.

Image Registration Examples

For most real-world medical image registration, some mix of both affine and deformable registration is used We will discuss strategies for improving image registration performance in more detail in Lecture 20. For this lecture we stick to rigid and affine transformations and basic image registration methods to gain intuition and experience.

Basic Image Registration

As introduced in the first lecture, the registration problem is in general ill-posed in that there are more degrees of freedom constraints. However, one way to constrain the problem is by limiting the degrees of freedom (DOF) of the transform mapping.

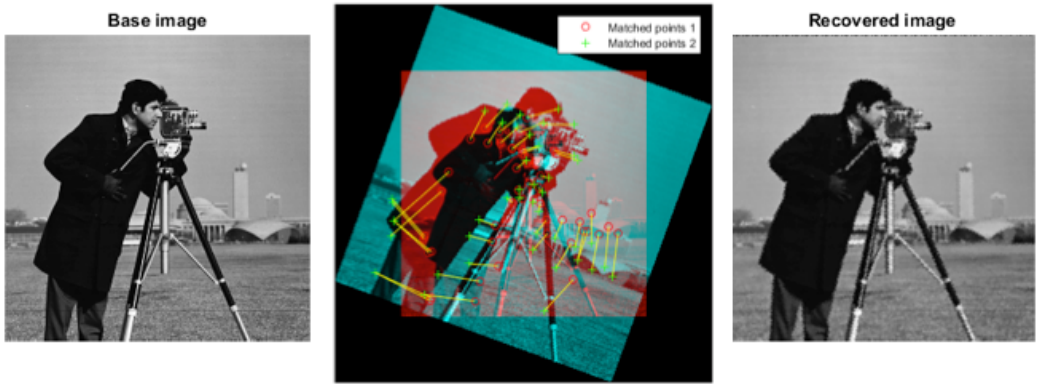


Figure 4: An example of feature-based segmentation. Note the interpolation error prominent in the restored image. The source of this error comes from errors in restoration of the signal intensity when the image coordinates from the moving image are mapped back to the fixed space.

The affine transform is an example of this. In such circumstances where fiducial markers are included in the imaging field of view or a strictly rigid transform is assumed, then the transform will operate the same on all coordinates in the moving and fixed space. In such circumstances, a common approach to image registration is “control point” or “fiducial” marker-based registration. A related approach is called “Feature-Based” registration in which anatomic markers within the two images are used to perform a registration based on a rigid transform matrix (Figure 4). Observe some examples using the scripts provided (e.g. Affine_transform.m and Registration_script.m). Specifically look for errors due to interpolation and the impact of global vs. local feature selection.

In summary, the affine transform is a powerful tool for linear mapping of on coordinate system to another that provides a strong framework for basic image registration tasks under specific geometric constraints. The limitations of the affine transform 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.