

Bioengineering Imaging Track Qualifying Exam Study Book

Markus Foote and Blake Zimmerman

August 8, 2017

Contents

1	Affine Transform	2
2	Magnetic Resonance Imaging	4
3	Image Processing: Hough Transform	5

1. Affine Transform

An *affine transform* is the combination of a linear transformation with a translation. In a linear algebra contest, a point in space is a column vector of individual values along each dimension, i.e. $\begin{bmatrix} x \\ y \end{bmatrix}$ or $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$ in 2 or 3 dimensions, respectively. This location is noted simply as \vec{x} .

The affine transform can thus be written as a matrix-vector multiplication, followed by vector addition:

$$\vec{y} = \mathbf{A}\vec{x} + \vec{b} \quad (1.1)$$

The linear transformation matrix \mathbf{A} has some important special cases:

Scaling This transformation scales each direction of a vector space by the corresponding c value:

$$\mathbf{A} = \begin{bmatrix} c_x & 0 \\ 0 & c_y \end{bmatrix} \text{ or } \begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & c_z \end{bmatrix}$$

Rotation This transformation rotates by an angle θ about the origin in 2 dimensions:

$$\mathbf{A} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

Translation This only translates all the points by a constant amount:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ and } \vec{b} = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Skew

Others?

These transformations are all well and good, but with them so far we can only take points and apply some fun tricks to move them around in cute ways. What if there were pairs of points with *known* (or *assumed*) correspondence, and we want to find an affine transform that makes the point pairs match?

Using notation:

$$\vec{x}_i \quad \text{source point(s)} \quad (1.2)$$

$$\vec{y}_i \quad \text{target point(s)} \quad (1.3)$$

$$n \quad \text{total number of point pairs} \quad (1.4)$$

$$\mathbf{A} \quad \text{linear transform matrix} \quad (1.5)$$

$$\vec{T} \quad \text{translation vector} \quad (1.6)$$

In a perfect world, all the points match exactly after being transformed, such that

$$\vec{y}_i = \mathbf{A}\vec{x}_i + \vec{T} \quad (1.7)$$

is satisfied. This linear system of equations could be easily solved with some linear algebra.

However, this world is not perfect. Due to noise, human error, systematic error, and spilling coffee on samples before *carefully* wiping it up, there will be noise and the points will not exactly match. There will be some error. We must account for this error.

We want to estimate the best transformation given some number of point pairs that will not all match exactly. Thus, we should minimize the error in the matching using a euclidean distance for the error, and squared error so that this function is convex (which is important when very close to zero error):

$$\min_{\mathbf{A}, T} \sum_i \|(\mathbf{A}\vec{x}_i + T) - \vec{y}_i\|^2 \quad (1.8)$$

How do we solve this?

First, if we are given \mathbf{A} , then

$$\delta T = 2 \sum_i^n (\mathbf{A}\vec{x}_i + T - \vec{y}_i) = 0 \quad (1.9)$$

The variation of T is set to zero because the first derivative of a function is zero at a minimum. Solving for T :

$$T = \frac{1}{n} \sum_{i=1}^N (\vec{y}_i - \mathbf{A}\vec{x}_i) \quad (1.10)$$

This translation is simply the difference of the centroids of the source and target points. Let us define

$$\tilde{y}_i = \vec{y}_i - \vec{c}_y \quad (1.11)$$

$$\tilde{x}_i = \vec{x}_i - \vec{c}_x \quad (1.12)$$

where c_x and c_y are the centroids of each point set. This modification of the problem implicitly removes T since both point sets will have the same centroid (the origin).

Assuming the points are centered the problem

$$\min_{\mathbf{A}} \sum_i \left\| \mathbf{A} \left(\vec{x}_i - \frac{1}{n} \sum_j \vec{x}_j \right) - \left(\vec{y}_i - \frac{1}{n} \sum_j \vec{y}_j \right) \right\|^2 \quad (1.13)$$

simplifies to

$$\min_{\mathbf{A}} \sum_i \|\mathbf{A}\tilde{x}_i - \tilde{y}_i\|^2 \quad (1.14)$$

Similar to the procedure performed to find and remove T , we now find the variation of \mathbf{A} and set it equal to zero:

$$\frac{\partial}{\partial \mathbf{A}} = 2 \sum_i (\mathbf{A}\tilde{x}_i - \tilde{y}_i) \tilde{x}_i^T = 0 \quad (1.15)$$

Then solving for \mathbf{A} :

$$\mathbf{A} = \left(\sum_i \tilde{x}_i \tilde{x}_i^T \right)^{-1} \left(\sum_i \tilde{y}_i \tilde{x}_i^T \right) \quad (1.16)$$

A special note here about the matrix $\sum_i \tilde{x}_i \tilde{x}_i^T$: A minimum of 3 non-co-linear points in 2D (or 4 non-coplanar points in 3D) are required for this matrix to be invertible.

2. Magnetic Resonance Imaging

3. Image Processing: Hough Transform

In image processing the **Hough Transform** is useful for *global* filtering, especially to find sets of pixels that lie along curves of a specified shape. It provides an elegant solution using a *discretized parameter space*, while the naïve, brute-force approach quickly becomes daunting. In the case of finding lines in an image of n points, the naïve approach involves iteration over all $\sim n^2$ possible lines and performing $\sim n^3$ comparisons for every point to each line. The $O(n^3)$ complexity exponentially worsens for shapes with higher dimensional parameter spaces. This approach is computationally prohibitive for non-trivial applications.

Instead, the Hough Transform accumulates non-background points in a discretized parameter space. The dimensionality of the parameter space is equal to the number of the parameters that describe the desired shape. In the case of a line, there are two parameters. However, the parameters must be defined with care. The naïve parameter choices for a line might be *slope* and *intercept*, but if the objective is to find vertical (or nearly vertical) lines, a method to accurately discretize infinity (or very large values) would be required. A better approach is to parameterize lines by their *normal* representation:

$$x \cos(\theta) + y \sin(\theta) = \rho . \tag{3.1}$$

In this parameterization, all parameters are bounded: $-\pi/2 \leq \theta \leq \pi/2$ and $-D \leq \rho \leq D$ for an image with D length between the two most distant corners.