Point cloud to point cloud rigid transformations

Russell Taylor
600.445

Minimizing Rigid Registration Errors

Typically, given a set of points $\{\mathbf{a}_i\}$ in one coordinate system and another set of points $\{\mathbf{b}_i\}$ in a second coordinate system Goal is to find $[\mathbf{R},\mathbf{p}]$ that minimizes

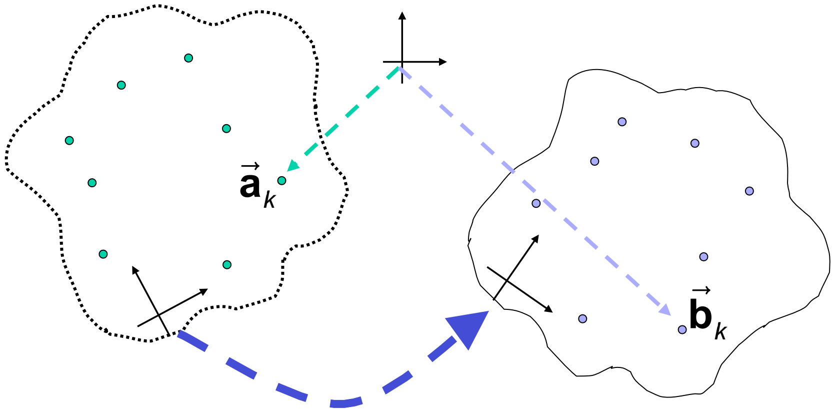
$$\eta = \sum_i \mathbf{e}_i \bullet \mathbf{e}_i$$

where

$$\mathbf{e}_i = (\mathbf{R} \bullet \mathbf{a}_i + \mathbf{p}) - \mathbf{b}_i$$

This is tricky, because of **R**.

Point cloud to point cloud registration



$$\mathbf{R}\bullet\vec{\mathbf{a}}_k+\vec{\mathbf{p}}=\vec{\mathbf{b}}_k$$

Minimizing Rigid Registration Errors

Step 1: Compute

$$\overline{\mathbf{a}}=\frac{1}{N}\sum_{i=1}^{N}\vec{\mathbf{a}}_i \qquad \overline{\mathbf{b}}=\frac{1}{N}\sum_{i=1}^{N}\vec{\mathbf{b}}_i$$

$$\tilde{\mathbf{a}}_i=\vec{\mathbf{a}}_i-\overline{\mathbf{a}} \qquad \tilde{\mathbf{b}}_i=\vec{\mathbf{b}}_i-\overline{\mathbf{b}}$$

Step 2: Find $\mathbf{R}$ that minimizes

$$\sum_i(\mathbf{R}\cdot\tilde{\mathbf{a}}_i-\tilde{\mathbf{b}}_i)^2$$

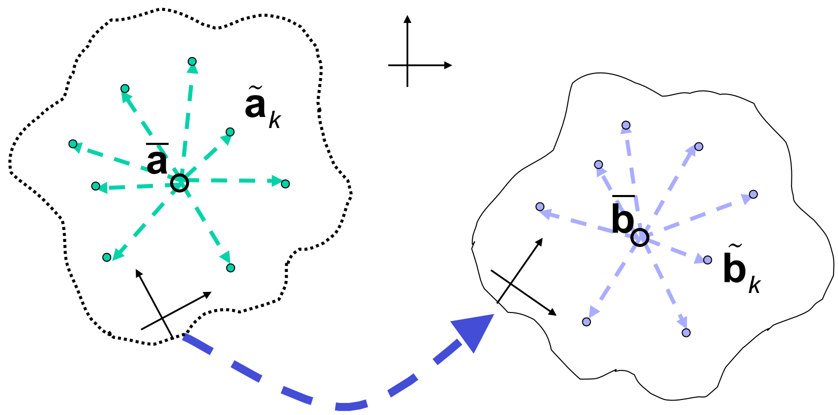
Step 3: Find $\vec{\mathbf{p}}$

$$\vec{\mathbf{p}}=\overline{\mathbf{b}}-\mathbf{R}\cdot\overline{\mathbf{a}}$$

Step 4: Desired transformation is

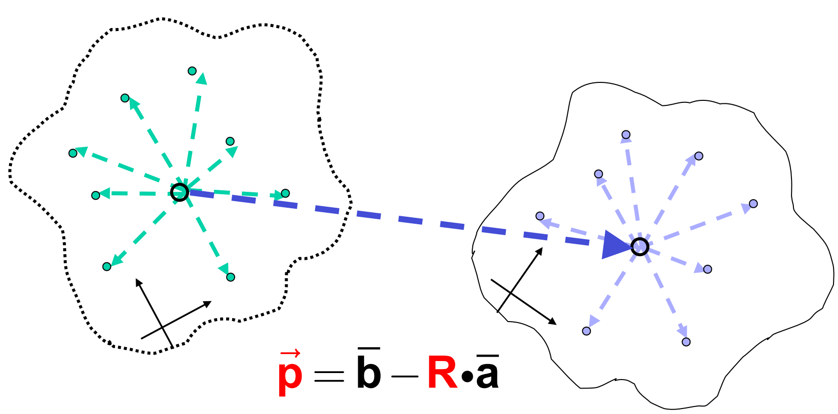
$$\mathbf{F}=Frame(\mathbf{R},\vec{\mathbf{p}})$$

Point cloud to point cloud registration



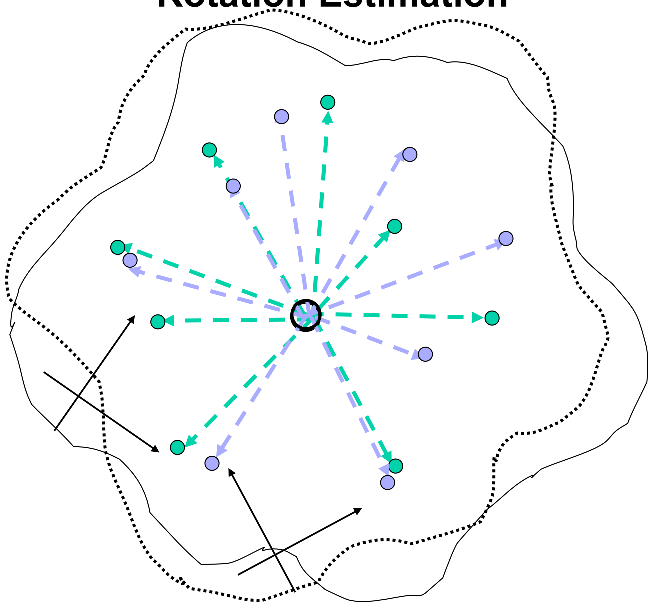
$$\mathbf{R} \bullet \vec{a}_k + \vec{p} = \vec{b}_k$$

Point cloud to point cloud registration

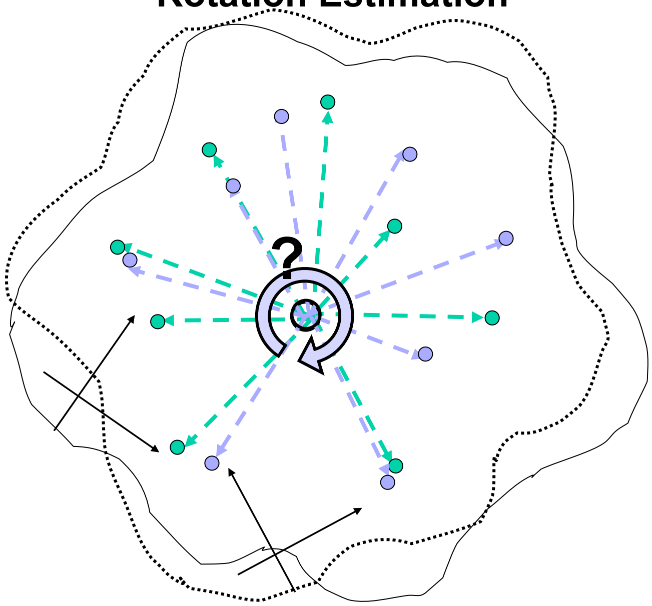


$$\vec{p} = \bar{b} - \mathbf{R} \bullet \bar{a}$$

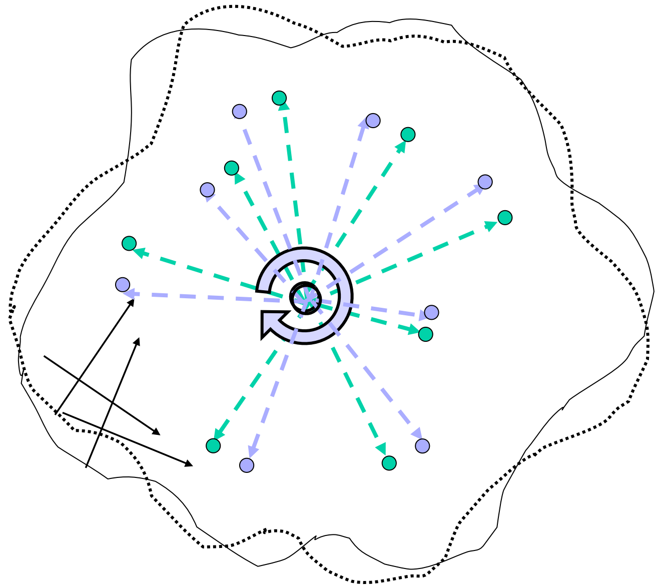
Rotation Estimation



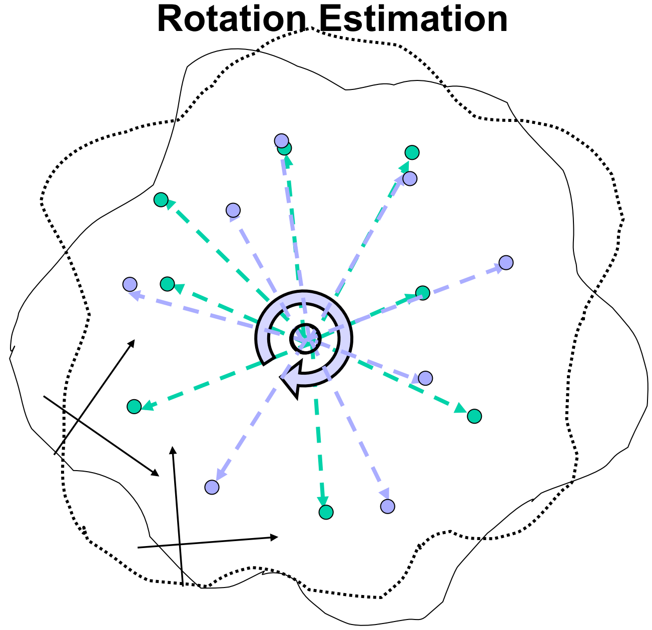
Rotation Estimation



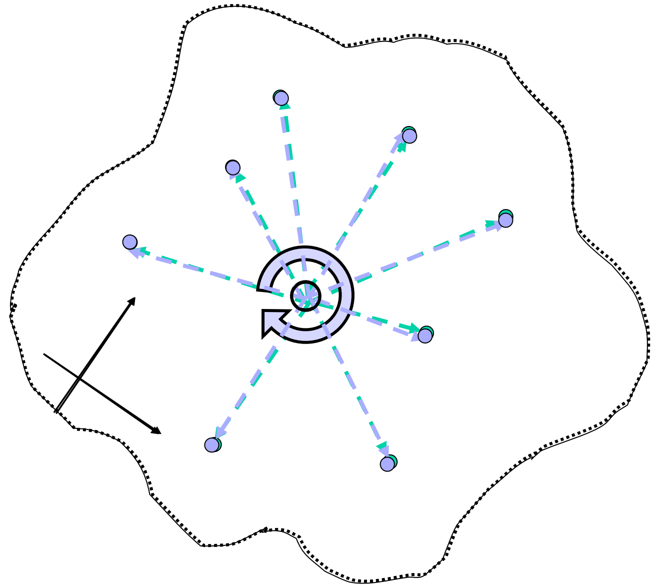
Rotation Estimation



Rotation Estimation



Point cloud to point cloud registration



Solving for R: iteration method

Given $\left\{\cdots,\left(\tilde{\mathbf{a}}_i,\tilde{\mathbf{b}}_i\right),\cdots\right\}$ want to find $\mathbf{R}=\arg\min\sum_i\left(\mathbf{R}\tilde{\mathbf{a}}_i-\tilde{\mathbf{b}}_i\right)$

Step 0: Make an initial guess $\mathbf{R}_0$

Step 1: Given $\mathbf{R}_k$, compute $\breve{\mathbf{b}}_i=\mathbf{R}_k^{-1}\tilde{\mathbf{b}}_i$

Step 2: Compute $\Delta\mathbf{R}$ that minimizes

$$\sum_i(\Delta\mathbf{R}\,\tilde{\mathbf{a}}_i-\breve{\mathbf{b}}_i)^2$$

Step 3: Set $\mathbf{R}_{k+1}=\mathbf{R}_k\Delta\mathbf{R}$

Step 4: Iterate Steps 1-3 until residual error is sufficiently small (or other termination condition)

Iterative method: Getting Initial Guess

We want to find an approximate solution $\mathbf{R}_0$ to

$$\mathbf{R}_0 \bullet \left[\cdots \tilde{\mathbf{a}}_i \cdots\right] \approx \left[\cdots \tilde{\mathbf{b}}_i \cdots\right]$$

One way to do this is as follows. Form matrices

$$\mathbf{A} = \left[\cdots \tilde{\mathbf{a}}_i \cdots\right] \quad \mathbf{B} = \left[\cdots \tilde{\mathbf{b}}_i \cdots\right]$$

Solve least-squares problem $\mathbf{M}_{3x3}\mathbf{A}_{3xN} \approx \mathbf{B}_{3xN}$

Note: You may find it easier to solve $\mathbf{A}^T_{3xN}\mathbf{M}^T_{3x3} \approx \mathbf{B}^T_{3xN}$

Set $\mathbf{R}_0 = \textit{orthogonalize}(\mathbf{M}_{3x3})$. Verify that $\mathbf{R}$ is a rotation

Our problem is now to solve $\mathbf{R}_0 \Delta\mathbf{R}\mathbf{A} \approx \mathbf{B}$. I.e., $\Delta\mathbf{R}\mathbf{A} \approx \mathbf{R}_0^{-1}\mathbf{B}$

Iterative method: Solving for ΔR

Approximate $\Delta\mathbf{R}$ as $(\mathbf{I} + \textit{skew}(\vec{\alpha}))$. I.e.,

$$\Delta\mathbf{R} \bullet \mathbf{v} \approx \mathbf{v} + \vec{\alpha} \times \mathbf{v}$$

for any vector $\mathbf{v}$. Then, our least squares problem becomes

$$\min_{\Delta\mathbf{R}} \sum_i (\Delta\mathbf{R} \bullet \tilde{\mathbf{a}}_i - \breve{\mathbf{b}}_i)^2 \approx \min_{\vec{\alpha}} \sum_i (\tilde{\mathbf{a}}_i - \breve{\mathbf{b}}_i + \vec{\alpha} \times \tilde{\mathbf{a}}_i)^2$$

This is linear least squares problem in $\vec{\alpha}$.

Then compute $\Delta\mathbf{R}(\vec{\alpha})$.

Direct Techniques to solve for R

- Method due to K. Arun, et. al., IEEE PAMI, Vol 9, no 5, pp 698-700, Sept 1987

Step 1: Compute

$$\mathbf{H} = \sum_i \begin{bmatrix} \tilde{a}_{i,x}\tilde{b}_{i,x} & \tilde{a}_{i,x}\tilde{b}_{i,y} & \tilde{a}_{i,x}\tilde{b}_{i,z} \\ \tilde{a}_{i,y}\tilde{b}_{i,x} & \tilde{a}_{i,y}\tilde{b}_{i,y} & \tilde{a}_{i,y}\tilde{b}_{i,z} \\ \tilde{a}_{i,z}\tilde{b}_{i,x} & \tilde{a}_{i,z}\tilde{b}_{i,y} & \tilde{a}_{i,z}\tilde{b}_{i,z} \end{bmatrix}$$

Step 2: Compute the SVD of $\mathbf{H} = \mathbf{USV}^{\mathbf{t}}$

Step 3: $\mathbf{R} = \mathbf{VU}^{\mathbf{t}}$

Step 4: Verify $Det(\mathbf{R}) = 1$. If not, then algorithm may fail.

- Failure is rare, and mostly fixable. The paper has details.

Quarternion Technique to solve for R

- B.K.P. Horn, "Closed form solution of absolute orientation using unit quaternions", *J. Opt. Soc. America*, A vol. 4, no. 4, pp 629-642, Apr. 1987.
- Method described as reported in Besl and McKay, "A method for registration of 3D shapes", *IEEE Trans. on Pattern Analysis and Machine* Intelligence, vol. 14, no. 2, February 1992.
- Solves a 4x4 eigenvalue problem to find a unit quaternion corresponding to the rotation
- This quaternion may be converted in closed form to get a more conventional rotation matrix

Digression: quaternions

Invented by Hamilton in 1843. Can be thought of as

4 elements: $\mathbf{q} = \left[q_0, q_1, q_2, q_3 \right]$

scalar & vector: $\mathbf{q} = s + \vec{\mathbf{v}} = \left[s, \vec{\mathbf{v}} \right]$

Complex number: $\mathbf{q} = q_0 + q_1 i + q_2 j + q_3 k$

where $i^2 = j^2 = k^2 = i\,j\,k = -1$

Properties:

Linearity: $\lambda \mathbf{q}_1 + \mu \vec{\mathbf{q}}_2 = \left[\lambda s_1 + \mu s_2, \lambda \vec{\mathbf{v}}_1 + \mu \vec{\mathbf{v}}_2 \right]$

Conjugate: $\mathbf{q}^* = s - \vec{\mathbf{v}} = \left[s, -\vec{\mathbf{v}} \right]$

Product: $\mathbf{q}_1 \circ \mathbf{q}_2 = \left[s_1 s_2 - \vec{\mathbf{v}}_1 \bullet \vec{\mathbf{v}}_2, s_1 \vec{\mathbf{v}}_2 + s_2 \vec{\mathbf{v}}_1 + \vec{\mathbf{v}}_1 \times \vec{\mathbf{v}}_2 \right]$

Transform vector: $\mathbf{q} \circ \vec{\mathbf{p}} \quad = \mathbf{q} \circ \left[0, \vec{\mathbf{p}} \right] \circ \mathbf{q}^*$

Norm: $\left\| \mathbf{q} \right\| = \sqrt{s^2 + \vec{\mathbf{v}} \bullet \vec{\mathbf{v}}} = \sqrt{q_0^{\ 2} + q_1^{\ 2} + q_2^{\ 2} + q_3^{\ 2}}$

Digression continued: unit quaternions

We can associate a rotation by angle θ about an axis $\vec{\mathbf{n}}$ with the unit quaternion:

$$Rot(\vec{\mathbf{n}}, \theta) \Leftrightarrow \left[\cos\frac{\theta}{2}, \ \sin\frac{\theta}{2} \vec{\mathbf{n}} \right]$$

Exercise: Demonstrate this relationship. I.e., show

$$Rot((\vec{\mathbf{n}}, \theta) \bullet \vec{\mathbf{p}} = \left[\cos\frac{\theta}{2}, \ \sin\frac{\theta}{2} \vec{\mathbf{n}} \right] \circ [0, \vec{\mathbf{p}}] \circ \left[\cos\frac{\theta}{2}, \ -\sin\frac{\theta}{2} \vec{\mathbf{n}} \right]$$

A bit more on quaternions

Exercise: show by substitution that the various formulations for quaternions are equivalent

A few web references:

http://mathworld.wolfram.com/Quaternion.html
http://en.wikipedia.org/wiki/Quaternion
http://en.wikipedia.org/wiki/Quaternions_and_spatial_rotation
http://www.euclideanspace.com/maths/algebra/realNormedAlgebra/quaternions/index.htm

Rotation matrix from unit quaternion

$$\mathbf{q} = [q_0, q_1, q_2, q_3]; \quad \|\mathbf{q}\| = 1$$

$$\mathbf{R}(\mathbf{q}) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

Unit quaternion from rotation matrix

$$\mathbf{R}(\mathbf{q}) = \begin{bmatrix} r_{xx} & r_{yx} & r_{zx} \\ r_{xy} & r_{yy} & r_{zy} \\ r_{xz} & r_{yz} & r_{zz} \end{bmatrix}; \quad \begin{matrix} a_0 = 1 + r_{xx} + r_{yy} + r_{zz};\ a_1 = 1 + r_{xx} - r_{yy} - r_{zz} \\ a_2 = 1 - r_{xx} + r_{yy} - r_{zz};\ a_3 = 1 - r_{xx} - r_{yy} + r_{zz} \end{matrix}$$

$a_0 = \max\{a_k\}$	$a_1 = \max\{a_k\}$	$a_2 = \max\{a_k\}$	$a_3 = \max\{a_k\}$
$q_0 = \frac{\sqrt{a_0}}{2}$	$q_0 = \frac{r_{yz} - r_{zy}}{4q_1}$	$q_0 = \frac{r_{zx} - r_{xz}}{4q_2}$	$q_0 = \frac{r_{xy} - r_{yx}}{4q_3}$
$q_1 = \frac{r_{xy} - r_{yx}}{4q_0}$	$q_1 = \frac{\sqrt{a_1}}{2}$	$q_1 = \frac{r_{xy} + r_{yx}}{4q_2}$	$q_1 = \frac{r_{xz} + r_{zx}}{4q_3}$
$q_2 = \frac{r_{zx} - r_{xz}}{4q_0}$	$q_2 = \frac{r_{xy} + r_{yx}}{4q_1}$	$q_2 = \frac{\sqrt{a_2}}{2}$	$q_2 = \frac{r_{yz} + r_{zy}}{4q_3}$
$q_3 = \frac{r_{yz} - r_{zy}}{4q_0}$	$q_3 = \frac{r_{xz} + r_{zx}}{4q_1}$	$q_3 = \frac{r_{yz} + r_{zy}}{4q_2}$	$q_3 = \frac{\sqrt{a_3}}{2}$

Rotation axis and angle from rotation matrix

Many options, including direct trigonemetric solution.
But this works:

$[\vec{\mathbf{n}},\theta] \leftarrow \mathit{ExtractAxisAngle}(\mathbf{R})$

$\{$

$\quad [s,\vec{\mathbf{v}}] \leftarrow \mathit{ConvertToQuaternion}(\mathbf{R})$

$\quad\quad \mathit{return}([\vec{\mathbf{v}} / \|\vec{\mathbf{v}}\|, 2\mathrm{atan}(s / \|\vec{\mathbf{v}}\|)$

$\}$

Quaternion method for R

Step 1: Compute

$$\mathbf{H} = \sum_i \begin{bmatrix} \tilde{a}_{i,x}\tilde{b}_{i,x} & \tilde{a}_{i,x}\tilde{b}_{i,y} & \tilde{a}_{i,x}\tilde{b}_{i,z} \\ \tilde{a}_{i,y}\tilde{b}_{i,x} & \tilde{a}_{i,y}\tilde{b}_{i,y} & \tilde{a}_{i,y}\tilde{b}_{i,z} \\ \tilde{a}_{i,z}\tilde{b}_{i,x} & \tilde{a}_{i,z}\tilde{b}_{i,y} & \tilde{a}_{i,z}\tilde{b}_{i,z} \end{bmatrix}$$

Step 2: Compute

$$\mathbf{G} = \begin{bmatrix} trace(\mathbf{H}) & \Delta^T \\ \Delta & \mathbf{H} + \mathbf{H}^T - trace(\mathbf{H})\mathbf{I} \end{bmatrix}$$

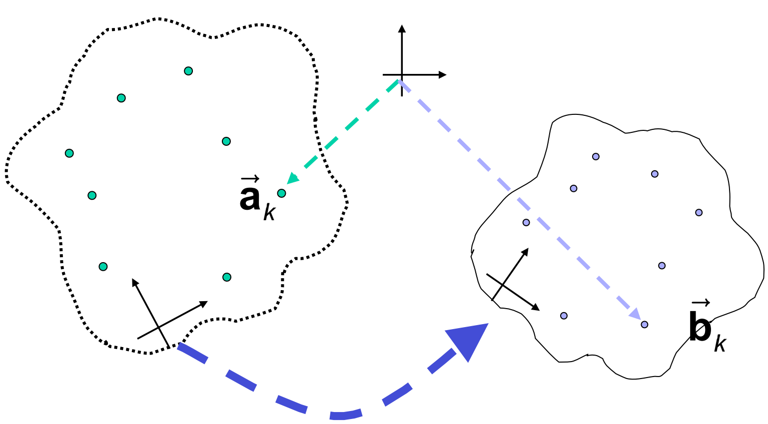
where $\Delta^T = \begin{bmatrix} \mathbf{H}_{2,3} - \mathbf{H}_{3,2} & \mathbf{H}_{3,1} - \mathbf{H}_{1,3} & \mathbf{H}_{1,2} - \mathbf{H}_{2,1} \end{bmatrix}$

Step 3: Compute eigen value decomposition of $\mathbf{G}$

$\text{diag}(\bar{\lambda}) = \mathbf{Q}^T\mathbf{G}\mathbf{Q}$

Step 4: The eigenvector $\mathbf{Q}_k = \begin{bmatrix} q_0, q_1, q_2, q_3 \end{bmatrix}$ corresponding to the largest eigenvalue λ_k is a unit quaternion corresponding to the rotation.

Non-reflective spatial similarity (rigid+scale)



$$\sigma\mathbf{R} \bullet \vec{\mathbf{a}}_k + \vec{\mathbf{p}} = \vec{\mathbf{b}}_k$$

Non-reflective spatial similarity

Step 1: Compute

$$\bar{\mathbf{a}} = \frac{1}{N}\sum_{i=1}^{N}\vec{\mathbf{a}}_i \qquad \bar{\mathbf{b}} = \frac{1}{N}\sum_{i=1}^{N}\vec{\mathbf{b}}_i$$

$$\tilde{\mathbf{a}}_i = \vec{\mathbf{a}}_i - \bar{\mathbf{a}} \qquad \tilde{\mathbf{b}}_i = \vec{\mathbf{b}}_i - \bar{\mathbf{b}}$$

Step 2: Estimate scale

$$\sigma = \frac{\sum_i \|\tilde{\mathbf{b}}_i\|}{\sum_i \|\tilde{\mathbf{a}}_i\|}$$

Step 3: Find $\mathbf{R}$ that minimizes

$$\sum_i (\mathbf{R}\cdot(\sigma\tilde{\mathbf{a}}_i) - \tilde{\mathbf{b}}_i)^2$$

Step 4: Find $\vec{\mathbf{p}}$

$$\vec{\mathbf{p}} = \bar{\mathbf{b}} - \mathbf{R}\cdot\bar{\mathbf{a}}$$

Step 5: Desired transformation is

$$\mathbf{F} = \mathit{SimilarityFrame}(\sigma, \mathbf{R}, \vec{\mathbf{p}})$$