

# Point cloud to point cloud rigid transformations

Russell Taylor  
601.455-655

1 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



1

## Minimizing Rigid Registration Errors

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

$$\eta = \sum_i e_i \bullet e_i$$

where

$$e_i = (R \bullet a_i + p) - b_i$$

This is tricky, because of  $R$ .

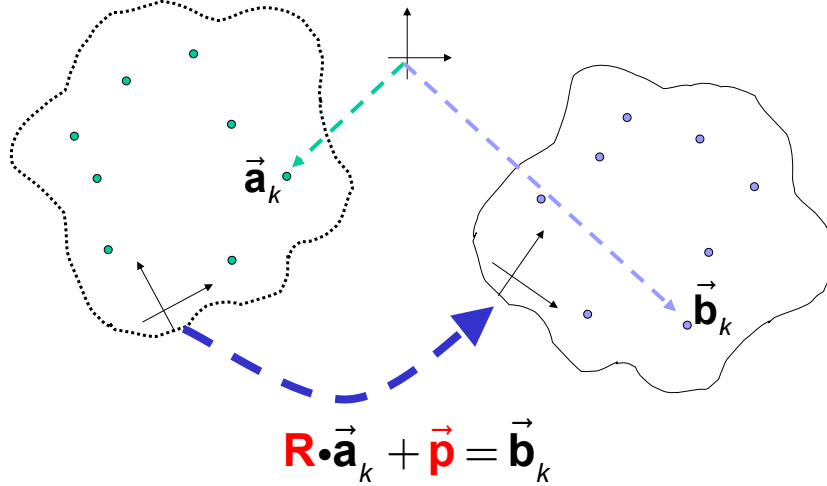
2 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



2

## Point cloud to point cloud registration



3 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



3

## Minimizing Rigid Registration Errors

Step 1: Compute

$$\begin{aligned} \bar{\mathbf{a}} &= \frac{1}{N} \sum_{i=1}^N \vec{\mathbf{a}}_i & \bar{\mathbf{b}} &= \frac{1}{N} \sum_{i=1}^N \vec{\mathbf{b}}_i \\ \tilde{\mathbf{a}}_i &= \vec{\mathbf{a}}_i - \bar{\mathbf{a}} & \tilde{\mathbf{b}}_i &= \vec{\mathbf{b}}_i - \bar{\mathbf{b}} \end{aligned}$$

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

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

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

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

Step 4: Desired transformation is

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

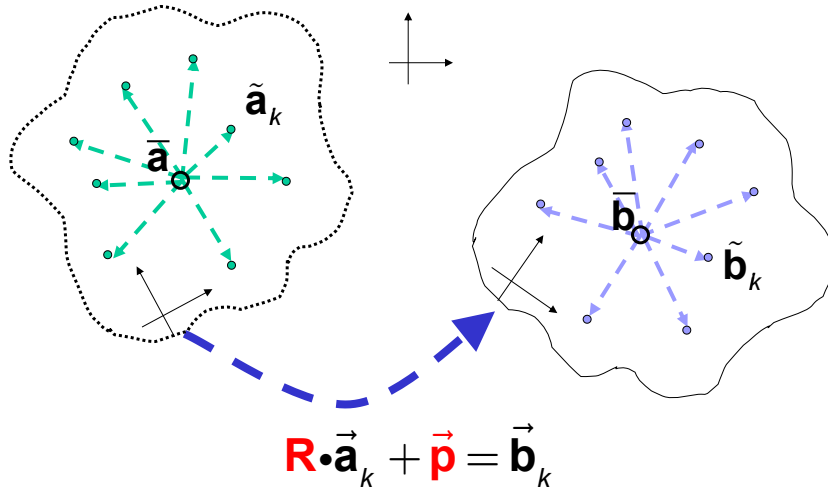
4 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



4

## Point cloud to point cloud registration



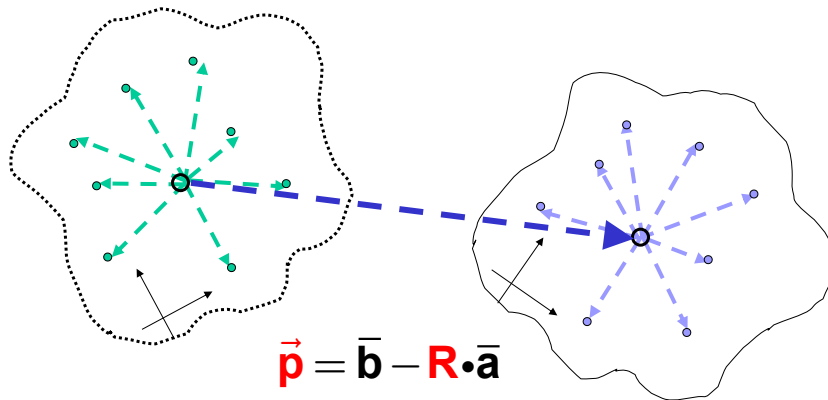
5 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



5

## Point cloud to point cloud registration

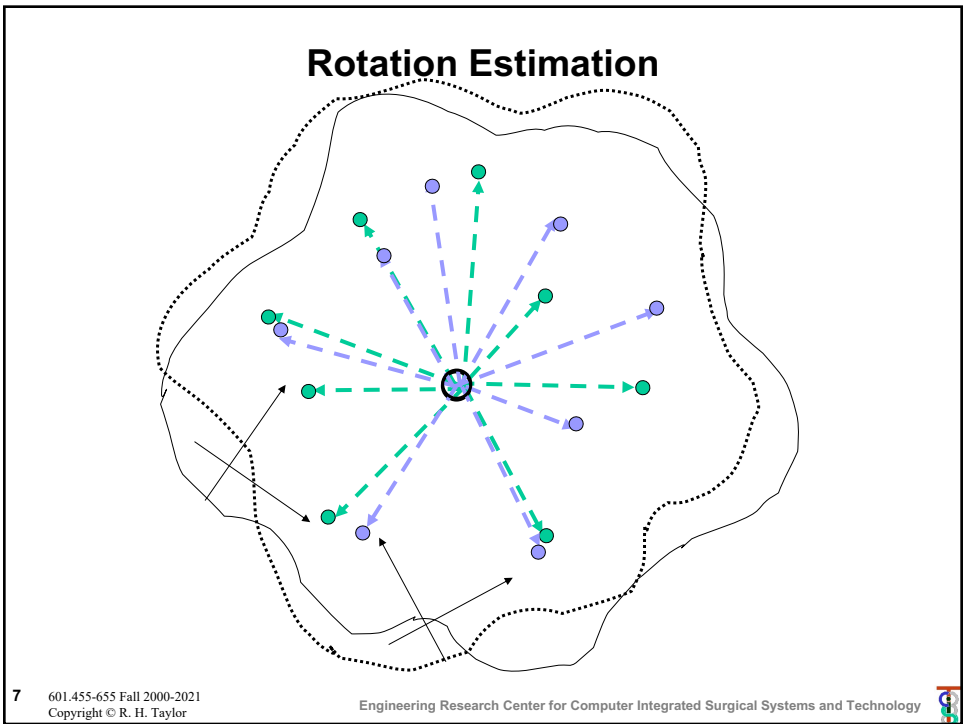


6 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

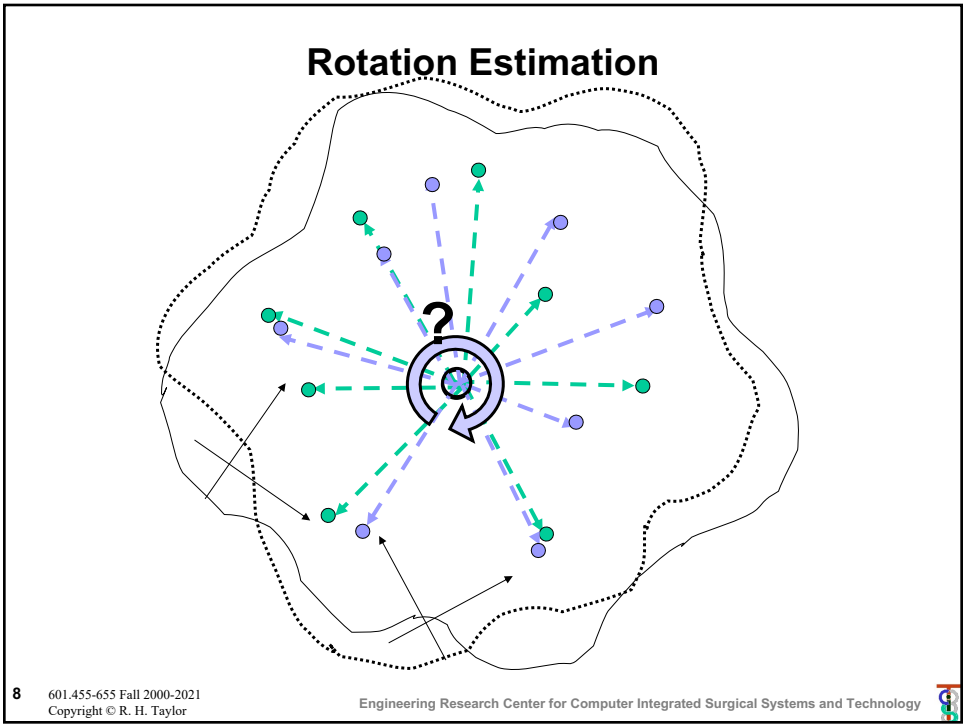
Engineering Research Center for Computer Integrated Surgical Systems and Technology



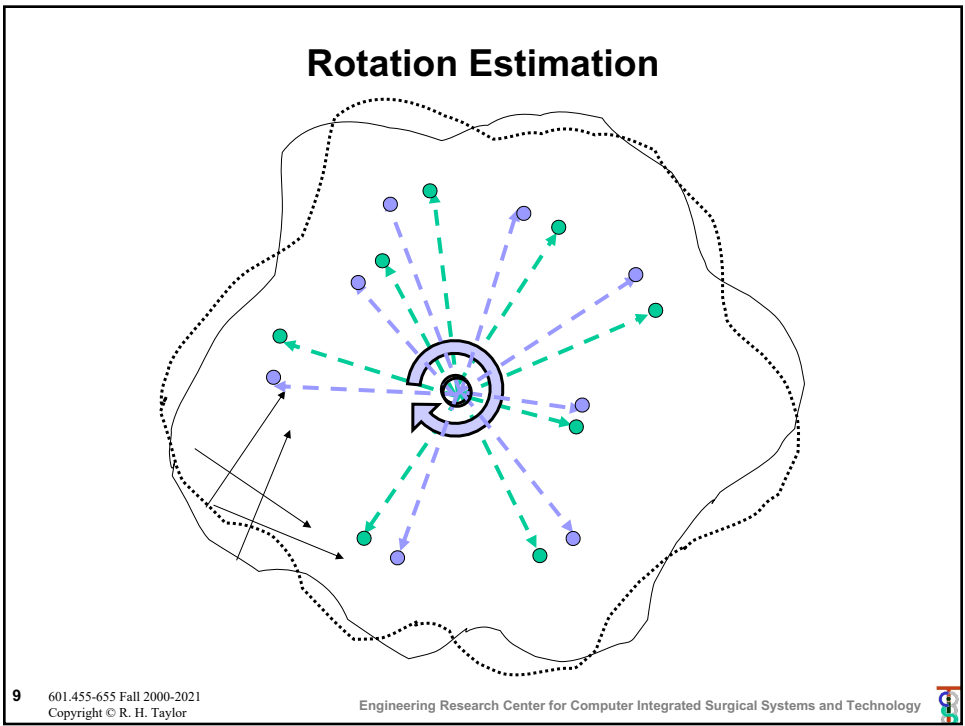
6



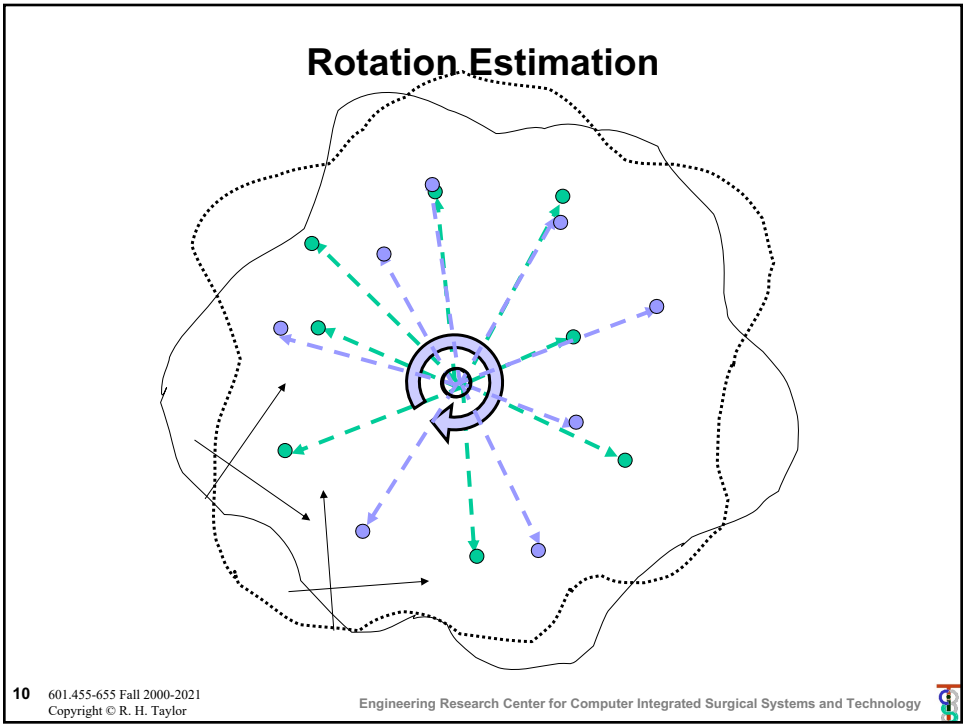
7



8

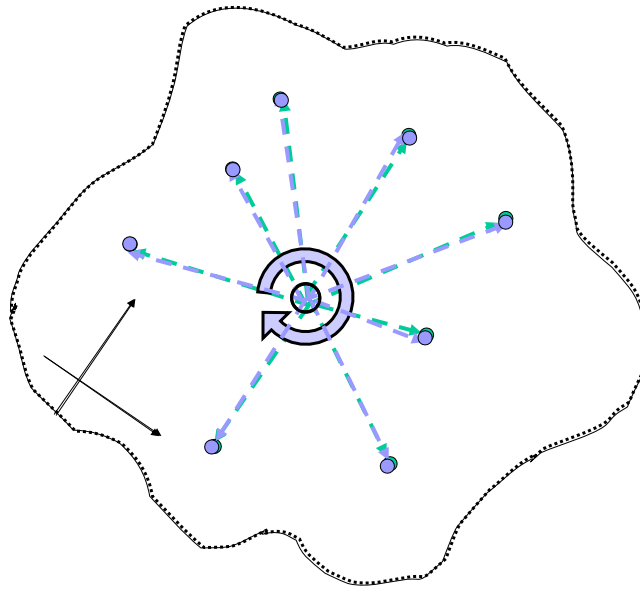


9



10

## Point cloud to point cloud registration



11 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



11

## Solving for R: iteration method

Given  $\{\dots, (\tilde{\mathbf{a}}_i, \tilde{\mathbf{b}}_i), \dots\}$ , want to find  $\mathbf{R} = \arg \min \sum_i \|\mathbf{R}\tilde{\mathbf{a}}_i - \tilde{\mathbf{b}}_i\|^2$

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

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

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

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

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)

12 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



12

## Iterative method: Getting Initial Guess

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

$$\mathbf{R}_0 \cdot [\dots \tilde{\mathbf{a}}_i \dots] \approx [\dots \tilde{\mathbf{b}}_i \dots]$$

One way to do this is as follows. Form matrices

$$\mathbf{A} = [\dots \tilde{\mathbf{a}}_i \dots] \quad \mathbf{B} = [\dots \tilde{\mathbf{b}}_i \dots]$$

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

**Note** : You may find it easier to solve  $\mathbf{A}_{3 \times N}^T \mathbf{M}_{3 \times 3}^T \approx \mathbf{B}_{3 \times N}^T$

Set  $\mathbf{R}_0 = \text{orthogonalize}(\mathbf{M}_{3 \times 3})$ . 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 $\Delta \mathbf{R}$

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

$$\Delta \mathbf{R} \cdot \mathbf{v} \approx \mathbf{v} + \bar{\alpha} \times \mathbf{v}$$

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

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

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

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



**Note: Use trigonometric formulas to compute this**



## Direct Iterative approach for Rigid Frame

Given  $\{\dots, (\vec{\mathbf{a}}_i, \vec{\mathbf{b}}_i), \dots\}$ , want to find  $\mathbf{F} = \arg \min \sum_i \|\mathbf{F}\vec{\mathbf{a}}_i - \vec{\mathbf{b}}_i\|^2$

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

Step 1: Given  $\mathbf{F}_k$ , compute  $\vec{\mathbf{a}}_i^k = \mathbf{F}_k \vec{\mathbf{a}}_i$

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

$$\sum_i \|\Delta \mathbf{F} \vec{\mathbf{a}}_i^k - \vec{\mathbf{b}}_i\|^2$$

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

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



## Direct Iterative approach for Rigid Frame

To solve for  $\Delta \mathbf{F} = \arg \min \sum_i \|\Delta \mathbf{F} \vec{\mathbf{a}}_i^k - \vec{\mathbf{b}}_i\|^2$

$$\Delta \mathbf{F} \vec{\mathbf{a}}_i^k - \vec{\mathbf{b}}_i \approx \vec{\alpha} \times \vec{\mathbf{a}}_i^k + \vec{\epsilon} + \vec{\mathbf{a}}_i^k - \vec{\mathbf{b}}_i$$

$$\vec{\alpha} \times \vec{\mathbf{a}}_i^k + \vec{\epsilon} \approx \vec{\mathbf{b}}_i - \vec{\mathbf{a}}_i^k$$

$$sk(-\vec{\mathbf{a}}_i^k) \vec{\alpha} + \vec{\epsilon} \approx \vec{\mathbf{b}}_i - \vec{\mathbf{a}}_i^k$$

Solve the least-squares problem

$$\begin{bmatrix} \vdots & \vdots \\ sk(-\vec{\mathbf{a}}_i^k) & \mathbf{I} \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} \vec{\alpha} \\ \vec{\epsilon} \end{bmatrix} \approx \begin{bmatrix} \vdots \\ \vec{\mathbf{b}}_i - \vec{\mathbf{a}}_i^k \\ \vdots \end{bmatrix}$$

Now set  $\Delta \mathbf{F} = [\Delta \mathbf{R}(\vec{\alpha}), \vec{\epsilon}]$





## 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

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

NOTE well

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

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

Step 4: Verify  $\text{Det}(\mathbf{R}) = 1$ . 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

$$\begin{aligned} \text{4 elements:} \quad & \mathbf{q} = [q_0, q_1, q_2, q_3] \\ \text{scalar \& vector:} \quad & \mathbf{q} = s + \vec{\mathbf{v}} = [s, \vec{\mathbf{v}}] \\ \text{Complex number:} \quad & \mathbf{q} = q_0 + q_1 i + q_2 j + q_3 k \\ & \text{where } i^2 = j^2 = k^2 = i j k = -1 \end{aligned}$$

Properties:

$$\begin{aligned} \text{Linearity:} \quad & \lambda \mathbf{q}_1 + \mu \mathbf{q}_2 = [\lambda s_1 + \mu s_2, \lambda \vec{\mathbf{v}}_1 + \mu \vec{\mathbf{v}}_2] \\ \text{Conjugate:} \quad & \mathbf{q}^* = s - \vec{\mathbf{v}} = [s, -\vec{\mathbf{v}}] \\ \text{Product:} \quad & \mathbf{q}_1 \circ \mathbf{q}_2 = [s_1 s_2 - \vec{\mathbf{v}}_1 \bullet \vec{\mathbf{v}}_2, s_1 \vec{\mathbf{v}}_2 + s_2 \vec{\mathbf{v}}_1 + \vec{\mathbf{v}}_1 \times \vec{\mathbf{v}}_2] \\ \text{Transform vector:} \quad & \mathbf{q} \circ \vec{\mathbf{p}} = \mathbf{q} \circ [0, \vec{\mathbf{p}}] \circ \mathbf{q}^* \\ \text{Norm:} \quad & \|\mathbf{q}\| = \sqrt{s^2 + \vec{\mathbf{v}} \bullet \vec{\mathbf{v}}} = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} \end{aligned}$$



## Digression continued: unit quaternions

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

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

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

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

**Hint:** Substitute and reduce to see if you get Rodrigues' formula.



## 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://en.wikipedia.org/wiki/Quaternions_and_spatial_rotation)

<http://www.euclideanspace.com/maths/algebra/realNormedAlgebra/quaternions/index.htm>

**CAUTION:** Different software packages are not always consistent in the order of elements if a quaternion is represented as a 4 element vector. Some put the scalar part first, others (including cisst libraries) put it last.



## Rotation matrix from unit quaternion

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

$$\mathbf{R}(\mathbf{q}) = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$



## Unit quaternion from rotation matrix

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

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

23 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



23

## Rotation axis and angle from rotation matrix

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

```
[ $\hat{\mathbf{n}}, \theta$ ]  $\leftarrow$  ExtractAxisAngle( $\mathbf{R}$ )
{
    [ $s, \vec{\mathbf{v}}$ ]  $\leftarrow$  ConvertToQuaternion( $\mathbf{R}$ )
    return([ $\vec{\mathbf{v}} / \|\vec{\mathbf{v}}\|, 2\text{atan}(s / \|\vec{\mathbf{v}}\|)$ )
}
```

24 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



24

## 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

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

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

Step 3: Compute eigen value decomposition of G

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

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



## Another Quaternion Method for R

Let  $\mathbf{q} = \mathbf{s} + \bar{\mathbf{v}}$  be the unit quaternion corresponding to  $\mathbf{R}$ . Let  $\tilde{\mathbf{a}}$  and  $\tilde{\mathbf{b}}$  be vectors with  $\tilde{\mathbf{b}} = \mathbf{R} \cdot \tilde{\mathbf{a}}$  then we have the quaternion equation

$$(\mathbf{s} + \bar{\mathbf{v}}) \cdot (\mathbf{0} + \tilde{\mathbf{a}})(\mathbf{s} - \bar{\mathbf{v}}) = \mathbf{0} + \tilde{\mathbf{b}}$$

$$(\mathbf{s} + \bar{\mathbf{v}}) \cdot (\mathbf{0} + \tilde{\mathbf{a}}) = (\mathbf{0} + \tilde{\mathbf{b}}) \cdot (\mathbf{s} + \bar{\mathbf{v}}) \quad \text{since } (\mathbf{s} - \bar{\mathbf{v}})(\mathbf{s} + \bar{\mathbf{v}}) = \mathbf{1} + \bar{\mathbf{0}}$$

Expanding the scalar and vector parts gives

$$-\bar{\mathbf{v}} \cdot \tilde{\mathbf{a}} = -\bar{\mathbf{v}} \cdot \tilde{\mathbf{b}}$$

$$\mathbf{s}\tilde{\mathbf{a}} + \bar{\mathbf{v}} \times \tilde{\mathbf{a}} = \mathbf{s}\tilde{\mathbf{b}} + \bar{\mathbf{v}} \times \tilde{\mathbf{b}}$$

Rearranging ...

$$(\tilde{\mathbf{b}} - \tilde{\mathbf{a}}) \cdot \bar{\mathbf{v}} = 0$$

$$\mathbf{s}(\tilde{\mathbf{b}} - \tilde{\mathbf{a}}) + (\tilde{\mathbf{b}} + \tilde{\mathbf{a}}) \times \bar{\mathbf{v}} = \bar{\mathbf{0}}_3$$

**NOTE:** This method works for any set of vectors  $\tilde{\mathbf{a}}$  and  $\tilde{\mathbf{b}}$ . We are using the symbols  $\tilde{\mathbf{a}}$  and  $\tilde{\mathbf{b}}$  to maintain consistency with the discussion of the previous method.



## Another Quaternion Method for R

Expressing this as a matrix equation

$$\begin{bmatrix} 0 & (\tilde{\mathbf{b}} - \tilde{\mathbf{a}})^T \\ (\tilde{\mathbf{b}} - \tilde{\mathbf{a}}) & sk(\tilde{\mathbf{b}} + \tilde{\mathbf{a}}) \end{bmatrix} \begin{bmatrix} s \\ \vec{\mathbf{v}} \end{bmatrix} = \begin{bmatrix} 0 \\ \vec{\mathbf{0}}_3 \end{bmatrix}$$

If we now express the quaternion  $\mathbf{q}$  as a 4-vector  $\vec{\mathbf{q}} = [s, \vec{\mathbf{v}}]^T$ , we can express the rotation problem as the constrained linear system

$$\begin{aligned} \mathbf{M}(\vec{\mathbf{a}}, \vec{\mathbf{b}}) \vec{\mathbf{q}} &= \vec{\mathbf{0}}_4 \\ \|\vec{\mathbf{q}}\| &= 1 \end{aligned}$$

27 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



27

## Another Quaternion Method for R

In general, we have many observations, and we want to solve the problem in a least squares sense:

$$\min \|\mathbf{M}\vec{\mathbf{q}}\| \text{ subject to } \|\vec{\mathbf{q}}\| = 1$$

where

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}(\vec{\mathbf{a}}_1, \vec{\mathbf{b}}_1) \\ \vdots \\ \mathbf{M}(\vec{\mathbf{a}}_n, \vec{\mathbf{b}}_n) \end{bmatrix} \text{ and } n \text{ is the number of observations}$$

Taking the singular value decomposition of  $\mathbf{M} = \mathbf{U}\Sigma\mathbf{V}^T$  reduces this to the easier problem

$$\min \|\mathbf{U}\Sigma\mathbf{V}^T\vec{\mathbf{q}}_x\| = \|\mathbf{U}(\Sigma\vec{\mathbf{y}})\| = \|\Sigma\vec{\mathbf{y}}\| \text{ subject to } \|\vec{\mathbf{y}}\| = \|\mathbf{V}^T\vec{\mathbf{q}}\| = \|\vec{\mathbf{q}}\| = 1$$

28 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



28

## Another Quaternion Method for R

This problem is just

$$\min \|\Sigma \bar{\mathbf{y}}\| = \left\| \begin{bmatrix} \sigma_1 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 \\ 0 & 0 & 0 & \sigma_4 \end{bmatrix} \bar{\mathbf{y}} \right\| \quad \text{subject to } \|\bar{\mathbf{y}}\| = 1$$

where  $\sigma_i$  are the singular values. Recall that SVD routines typically return the  $\sigma_i \geq 0$  and sorted in decreasing magnitude. So  $\sigma_4$  is the smallest singular value and the value of  $\bar{\mathbf{y}}$  with  $\|\bar{\mathbf{y}}\| = 1$  that minimizes  $\|\Sigma \bar{\mathbf{y}}\|$  is  $\bar{\mathbf{y}} = [0, 0, 0, 1]^T$ . The corresponding value of  $\bar{\mathbf{q}}$  is given by  $\bar{\mathbf{q}} = \mathbf{V}\bar{\mathbf{y}} = \mathbf{V}_4$ . Where  $\mathbf{V}_4$  is the 4th column of  $\mathbf{V}$ .

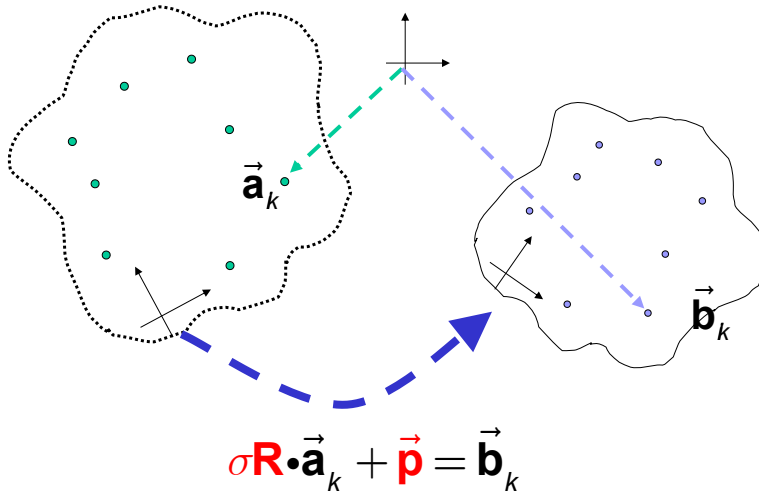
29 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



29

## Non-reflective spatial similarity (rigid+scale)



30 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



30

## Non-reflective spatial similarity

Step 1: Compute

$$\begin{aligned}\bar{\mathbf{a}} &= \frac{1}{N} \sum_{i=1}^N \bar{\mathbf{a}}_i & \bar{\mathbf{b}} &= \frac{1}{N} \sum_{i=1}^N \bar{\mathbf{b}}_i \\ \tilde{\mathbf{a}}_i &= \bar{\mathbf{a}}_i - \bar{\mathbf{a}} & \tilde{\mathbf{b}}_i &= \bar{\mathbf{b}}_i - \bar{\mathbf{b}}\end{aligned}$$

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  $\bar{\mathbf{p}}$

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

Step 5: Desired transformation is

$$\mathbf{F} = \text{SimilarityFrame}(\sigma, \mathbf{R}, \bar{\mathbf{p}})$$

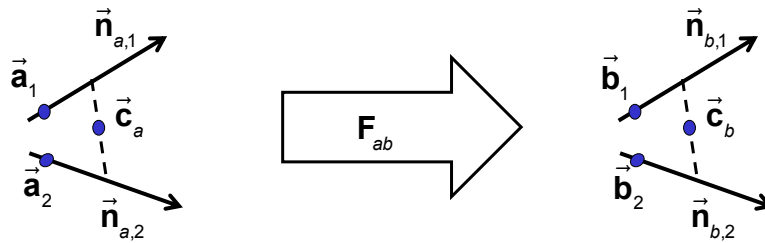
31 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



31

## Registration from line pairs



Approach 1:

Compute  $\mathbf{F}_a = [\mathbf{R}_a, \vec{\mathbf{c}}_a]$  from line pair  $a$

Compute  $\mathbf{F}_b = [\mathbf{R}_b, \vec{\mathbf{c}}_b]$  from line pair  $b$

$$\mathbf{F}_{ab} = \mathbf{F}_a^{-1} \mathbf{F}_b$$

32 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

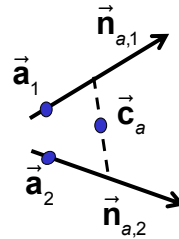
Engineering Research Center for Computer Integrated Surgical Systems and Technology



32



## Registration from line pairs



$$\mathbf{R}_a = \begin{bmatrix} \vec{x}_1 = \vec{n}_{a,1} & \vec{y}_1 = \frac{\vec{n}_{a,1} \times \vec{n}_{a,2}}{\|\vec{n}_{a,1} \times \vec{n}_{a,2}\|} & \vec{z}_1 = \vec{x}_1 \times \vec{y}_1 \end{bmatrix}$$

$\vec{c}_a$  = midpoint between the two lines

To get the midpoint:

$$\text{Solve } \begin{bmatrix} \vec{n}_{a,1} & -\vec{n}_{a,2} \end{bmatrix} \begin{bmatrix} \lambda \\ \nu \end{bmatrix} \approx [\vec{a}_2 - \vec{a}_1]$$

$$\text{Then } \vec{c}_a = \frac{(\vec{a}_1 + \lambda \vec{n}_{a,1}) + (\vec{a}_2 + \nu \vec{n}_{a,2})}{2}$$

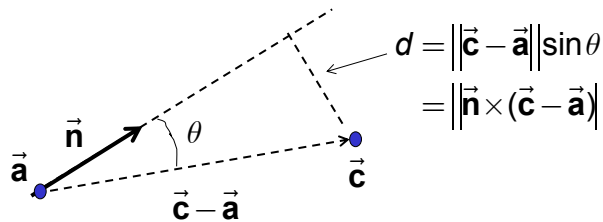
33 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



33

## Distance of a point from a line



So, to find the closest point to multiple lines

$$\vec{c} = \text{argmin} \sum_k d_k^2$$

Solve this problem in a least squares sense:

$$\vec{n}_k \times (\vec{c} - \vec{a}_k) \approx \vec{0} \text{ for } k = 1, \dots, n$$

Equivalently, solve

$$\vec{n}_k \times \vec{c} \approx \vec{n}_k \times \vec{a}_k \text{ for } k = 1, \dots, n$$

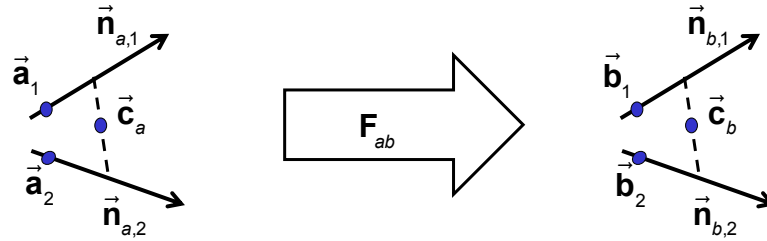
34 601.455-655 Fall 2000-2021  
Copyright © R. H. Taylor

Engineering Research Center for Computer Integrated Surgical Systems and Technology



34

## Registration from multiple corresponding lines



Approach 2:

$$\text{Solve } \mathbf{R}_{ab} \vec{n}_{b,k} \approx \vec{n}_{a,k} \text{ for } \mathbf{R}_{ab}$$

$$\text{Solve } \vec{n}_{a,k} \times \vec{c}_a \approx \vec{n}_{a,k} \times \vec{a}_k \text{ for } \vec{c}_a$$

$$\text{Solve } \vec{n}_{b,k} \times \vec{c}_b \approx \vec{n}_{b,k} \times \vec{b}_k \text{ for } \vec{c}_b$$

$$\vec{p}_{ab} = \vec{c}_a - \mathbf{R}_{ab} \vec{c}_b$$

