

# Rigid Registration

CS6240 Multimedia Analysis

Leow Wee Kheng

Department of Computer Science  
School of Computing  
National University of Singapore

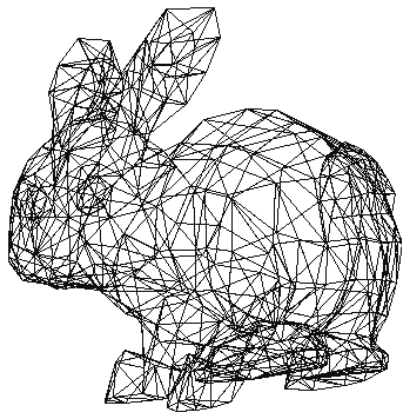


# Introduction

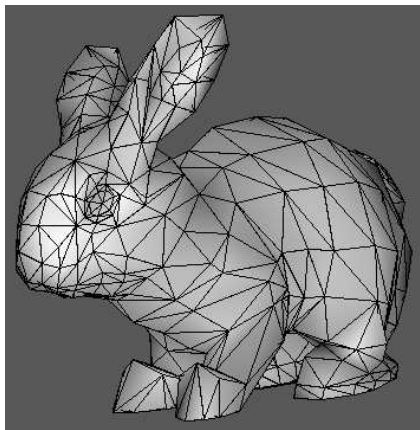
Rigid registration deals with the **spatial alignment** of **rigid objects**.

- Rigid objects are objects whose shapes are fixed.
- Examples: curves, planes, surfaces, cubes, spheres, cylinders, images.
- They can be represented by equations.
- They can also be represented by connected points.
- 3D free-form objects are typically represented by the **point-and-mesh** model:  
3D points connected into triangles forming the object's surface.

## Triangle Mesh Representation:



(a) wire frame



(b) with surface shading

A rigid object can undergo various transformations:

- **rigid transformation:**
  - rotation, translation
  - preserves distances between two points and angles made by three points
  - does not change shape
- **similarity transformation:**
  - rotation, scaling, translation
  - preserves angles made by three points
  - does not change shape

A special kind of linear transformation:

- **affine transformation:**
  - rotation, scaling, translation, reflection, shearing
  - may change shape
  - a simple form of **non-rigid transformation**.

These transformations are **linear transformations**.

## 3D Affine Transformation

Affine transformation includes rotation, scaling, translation, reflection and shearing. It may change shape.

So, it is a simple form of **non-rigid transformation**.

$$\mathbf{p}' = \mathbf{A}\mathbf{p}$$
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \quad (1)$$

# 3D Similarity Transformation

3D Similarity Transformation [FvDFH87, JKS95, SS01] includes 3D scaling, rotation, and translation.

**Scaling S:**

$$\mathbf{p}' = \mathbf{S} \mathbf{p}, \quad \mathbf{S} = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & s \end{bmatrix}. \quad (2)$$

**Translation T:**

$$\mathbf{p}' = \mathbf{p} + \mathbf{T}, \quad \mathbf{T} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}. \quad (3)$$

**Rotation** about  $x$ -axis  $\mathbf{R}_x(\omega)$ :

$$\mathbf{p}' = \mathbf{R}_x(\omega) \mathbf{p}, \quad \mathbf{R}_x(\omega) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \omega & -\sin \omega \\ 0 & \sin \omega & \cos \omega \end{bmatrix}. \quad (4)$$

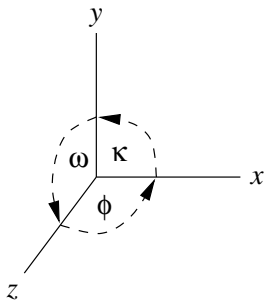
**Rotation** about  $y$ -axis  $\mathbf{R}_y(\phi)$ :

$$\mathbf{p}' = \mathbf{R}_y(\phi) \mathbf{p}, \quad \mathbf{R}_y(\phi) = \begin{bmatrix} \cos \phi & 0 & \sin \phi \\ 0 & 1 & 0 \\ -\sin \phi & 0 & \cos \phi \end{bmatrix}. \quad (5)$$

**Rotation** about  $z$ -axis  $\mathbf{R}_z(\kappa)$ :

$$\mathbf{p}' = \mathbf{R}_z(\kappa) \mathbf{p}, \quad \mathbf{R}_z(\kappa) = \begin{bmatrix} \cos \kappa & -\sin \kappa & 0 \\ \sin \kappa & \cos \kappa & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (6)$$

Rotation angles in right-handed coordinate system:



- $\omega$ : about  $x$ -axis, from positive  $y$ -axis to positive  $z$ -axis
- $\phi$ : about  $y$ -axis, from positive  $z$ -axis to positive  $x$ -axis
- $\kappa$ : about  $z$ -axis, from positive  $x$ -axis to positive  $y$ -axis

Combining the three rotations:

$$\mathbf{R} = \mathbf{R}_z(\kappa) \mathbf{R}_y(\phi) \mathbf{R}_x(\omega) = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (7)$$

where

$$\begin{aligned} r_{11} &= \cos \phi \cos \kappa \\ r_{12} &= \sin \omega \sin \phi \cos \kappa - \cos \omega \sin \kappa \\ r_{13} &= \cos \omega \sin \phi \cos \kappa + \sin \omega \sin \kappa \\ r_{21} &= \cos \phi \sin \kappa \\ r_{22} &= \sin \omega \sin \phi \sin \kappa + \cos \omega \cos \kappa \\ r_{23} &= \cos \omega \sin \phi \sin \kappa - \sin \omega \cos \kappa \\ r_{31} &= -\sin \phi \\ r_{32} &= \sin \omega \cos \phi \\ r_{33} &= \cos \omega \cos \phi. \end{aligned} \quad (8)$$

The rotation matrix is an orthonormal matrix (Exercise):

$$\mathbf{R}^\top \mathbf{R} = \mathbf{I}. \quad (9)$$

So,  $\mathbf{R}^{-1} = \mathbf{R}^\top$ . In program, should use  $\mathbf{R}^{-1}$  for accuracy.

In particular, let

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_1^\top \\ \mathbf{R}_2^\top \\ \mathbf{R}_3^\top \end{bmatrix}. \quad (10)$$

Then,

$$\mathbf{R}_i^\top \mathbf{R}_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \quad (11)$$

Again, can combine scaling, rotation, and translation to obtain

$$\mathbf{p}' = s \mathbf{R} \mathbf{p} + \mathbf{T}. \quad (12)$$

Using homogeneous coordinates yields

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s r_{11} & s r_{12} & s r_{13} & t_x \\ s r_{21} & s r_{22} & s r_{23} & t_y \\ s r_{31} & s r_{32} & s r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \quad (13)$$

Transformation matrix has 16 elements but only 7 independent variables.

## Notes:

- 3D rotation can also be represented using **quaternions** and **Rodrigues' Rotation Formula**.
- They are numerically more stable in programs.
- For details, refer to [JKS95, Kui99, Rod].

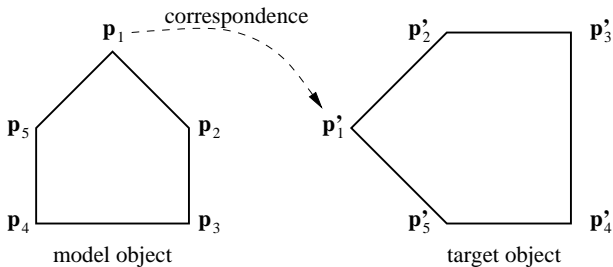
# Registration With Known Correspondence

General problem: Given two objects with **known correspondence** between their points,

- obtain the best-fitting transformation between the objects,
- **align** or **register** one object with the other.
- One of the objects can be regarded as the rigid model.

Notations:

- $\mathbf{p}_i$ ,  $i = 1, \dots, n$ , denote the  $N$ -D points of the model object.
- $\mathbf{p}'_i$  denote the **corresponding**  $N$ -D points of the target object, i.e.,  $\mathbf{p}'_i$  corresponds to  $\mathbf{p}_i$ .
- The two objects differ by a linear transformation  $\mathbf{T}$ .



Ideally,  $\mathbf{T} \mathbf{p}_i = \mathbf{p}'_i$ . So, can formulate the registration problem as follows:

*Find the linear transformation  $\mathbf{T}$  that minimizes the error  $E$ :*

$$E = \sum_{i=1}^n \|\mathbf{T} \mathbf{p}_i - \mathbf{p}'_i\|^2. \quad (14)$$

### Alternative Notation:

Denote linear transformation as a function  $T$ , i.e.,  $T(\mathbf{p}_i) = \mathbf{p}'_i$ . Then, the registration problem is to find the linear transformation  $T$  that minimizes the error  $E$ :

$$E = \sum_{i=1}^n \|T(\mathbf{p}_i) - \mathbf{p}'_i\|^2. \quad (15)$$

# Registration under Affine Transformation

Affine transformation is given by the equation

$$\mathbf{p}'_i = \mathbf{A} \mathbf{p}_i, \quad i = 1, \dots, n. \quad (16)$$

So, the problem is to find the matrix  $\mathbf{A}$  that minimizes the error  $E$ :

$$E = \sum_i \|\mathbf{p}'_i - \mathbf{A} \mathbf{p}_i\|^2. \quad (17)$$

This problem can be solved using **linear least square** method.

Reassemble Eq. 16 into the following form (Exercise):

$$\mathbf{M} \mathbf{a} = \mathbf{b}. \quad (18)$$

- $\mathbf{M}$  is a  $n \times 12$  matrix of known values from  $\mathbf{p}_i$  and  $\mathbf{p}'_i$ .
- $\mathbf{a}$  is a  $12 \times 1$  column matrix of affine parameters  $(a_{11}, a_{12}, \dots, a_{34})^\top$ .
- $\mathbf{b}$  is a  $n \times 1$  column matrix of known values from  $\mathbf{p}_i$  and  $\mathbf{p}'_i$ .
- Eq. 18 can be solved using **linear least square** method.  
Use software such as Matlab or Python.
- Linear least square method yields

$$\mathbf{a} = (\mathbf{M}^\top \mathbf{M})^{-1} \mathbf{M}^\top \mathbf{b}. \quad (19)$$

# Registration under Similarity Transformation

The linear transformation is a similarity transformation, which includes scaling  $s$ , rotation  $\mathbf{R}$ , and translation  $\mathbf{T}$ :

$$\mathbf{p}'_i = s \mathbf{R} \mathbf{p}_i + \mathbf{T}. \quad (20)$$

So, the registration problem is to find the  $s$ ,  $\mathbf{R}$ , and  $\mathbf{T}$  that minimize the error  $E$ :

$$E = \sum_{i=1}^n \|s \mathbf{R} \mathbf{p}_i + \mathbf{T} - \mathbf{p}'_i\|^2. \quad (21)$$

## Algorithm for Computing Similarity Transformation [HHN88]

**Step 1:** Remove translation by moving object's centroid to origin of coordinate system:

$$\mathbf{r}_i = \mathbf{p}_i - \bar{\mathbf{p}}, \quad \mathbf{r}'_i = \mathbf{p}'_i - \bar{\mathbf{p}}' \quad (22)$$

where

$$\bar{\mathbf{p}} = \frac{1}{n} \sum_{i=1}^n \mathbf{p}_i, \quad \bar{\mathbf{p}}' = \frac{1}{n} \sum_{i=1}^n \mathbf{p}'_i. \quad (23)$$

Now,  $\mathbf{r}_i$  and  $\mathbf{r}'_i$  are bundles of vectors.

**Step 2:** Determine scaling factor by comparing mean vector length:

$$s^2 = \frac{\sum_{i=1}^n \|\mathbf{r}'_i\|^2}{\sum_{i=1}^n \|\mathbf{r}_i\|^2} \quad (24)$$

For rigid transformation,  $s = 1$ , so this step can be skipped.

**Step 3:** Compute rotation matrix as follows:

Form matrix  $\mathbf{M}$  from sum of outer product:

$$\mathbf{M} = \sum_{i=1}^n \mathbf{r}'_i \mathbf{r}_i^\top. \quad (25)$$

The rotation matrix  $\mathbf{R}$  is given by

$$\mathbf{R} = \mathbf{M}\mathbf{Q}^{-1/2} \quad (26)$$

where  $\mathbf{Q} = \mathbf{M}^\top \mathbf{M}$ .

Perform **eigen decomposition** of  $\mathbf{Q}$  [PTVF92] to obtain

$$\mathbf{Q} = \lambda_1 \mathbf{v}_1 \mathbf{v}_1^\top + \lambda_2 \mathbf{v}_2 \mathbf{v}_2^\top + \lambda_3 \mathbf{v}_3 \mathbf{v}_3^\top \quad (27)$$

where  $\mathbf{v}_i$  and  $\lambda_i$  are the eigenvectors and eigenvalues.

The inverse square root of eigensystem can be easily computed as

$$\mathbf{Q}^{-1/2} = \frac{1}{\sqrt{\lambda_1}} \mathbf{v}_1 \mathbf{v}_1^\top + \frac{1}{\sqrt{\lambda_2}} \mathbf{v}_2 \mathbf{v}_2^\top + \frac{1}{\sqrt{\lambda_3}} \mathbf{v}_3 \mathbf{v}_3^\top. \quad (28)$$

Eq. 27 can also be written as

$$\mathbf{Q} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top \quad (29)$$

where

$$\mathbf{V} = [\mathbf{v}_1 \ \mathbf{v}_2 \ \mathbf{v}_3], \quad \mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \lambda_3). \quad (30)$$

Then,

$$\mathbf{Q}^{-1} = \mathbf{V} \text{diag} \left( \frac{1}{\sqrt{\lambda_1}}, \frac{1}{\sqrt{\lambda_2}}, \frac{1}{\sqrt{\lambda_3}} \right) \mathbf{V}^\top \quad (31)$$

**Step 4:** Given  $s$  and  $\mathbf{R}$ , we can now compute  $\mathbf{T}$ .

$$\mathbf{T} = \bar{\mathbf{p}}' - s \mathbf{R} \bar{\mathbf{p}}. \quad (32)$$

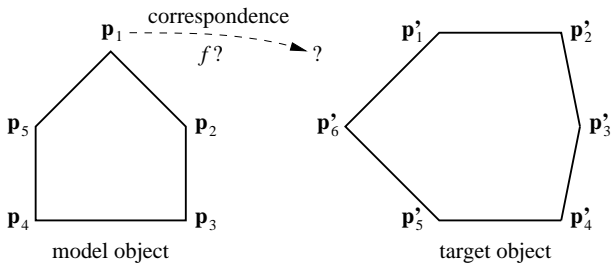
The  $s$ ,  $\mathbf{R}$ , and  $\mathbf{T}$  obtained are the best-fit solutions.

# Registration With Unknown Correspondence

General problem: Given two objects with **unknown correspondence** between their points,

- obtain the best-fitting transformation between the objects,
- **align** or **register** one object with the other.
- One of the objects can be regarded as the rigid model.

- $\mathbf{p}_i, i = 1, \dots, m$ , denote the  $N$ -D points of the model object  $M$ .
- $\mathbf{p}'_j, j = 1, \dots, n$ , denote the  $N$ -D points of the target object  $O$ .
- Correspondence between the points are unknown.



- Let  $f$  denote the **correspondence function** from the transformed model object to the target object.
- If  $f$  is known, then  $f(\mathbf{p}_i)$  is exactly one of the points in the target object. That is, there exist a  $\mathbf{p}'_j$  such that  $f(\mathbf{p}_i) = \mathbf{p}'_j$ .
- The two objects differ by a linear transformation  $T$ . So, ideally,  $f(\mathbf{p}_i) = T(\mathbf{p}_i)$ .

So, can formulate the registration problem as follows:

*Find the similarity transformation  $T$  and correspondence function  $f$  that minimize the error  $E$ :*

$$E = \sum_{i=1}^n \|T(\mathbf{p}_i) - f(\mathbf{p}_i)\|^2. \quad (33)$$

# Iterative Closest Point Algorithm

Main Ideas [BM92, Zha94]:

- Make **educated guess** about the correspondence function.
- When two objects are registered, the points on an object should be **close** to some points on the other object.

Given a point  $\mathbf{p}_i$  in  $M$ , the closest point  $\mathbf{p}'_k$  in  $O$  to  $\mathbf{p}_i$  satisfies

$$d(\mathbf{p}_i, \mathbf{p}'_k) = \min_{\mathbf{p}'_j \in O} d(\mathbf{p}_i, \mathbf{p}'_j) \quad (34)$$

where  $d(\cdot)$  is the Euclidean distance.

Let  $f(\mathbf{p}_i)$  denote the closest point of  $\mathbf{p}_i$  in  $O$ ,  
i.e.,  $f$  is the **closest point function**.

- Now, each point  $\mathbf{p}_i$  in  $M$  is matched to a point  $f(\mathbf{p}_i)$  in  $O$ .
- So, can apply the method of registration with point correspondence to find the best similarity transformation that register  $M$  with  $O$ .

## Iterative Closest Point Algorithm

Repeat for  $t = 0, 1, \dots$ ,

Find  $f(\mathbf{p}_i(t))$  for each  $\mathbf{p}_i(t)$  in  $M(t)$ .

Compute  $s(t)$ ,  $\mathbf{R}(t)$ , and  $\mathbf{T}(t)$  that minimize

$$E(t) = \sum_{\mathbf{p}_i(t) \in M(t)} \|s(t) \mathbf{R}(t) \mathbf{p}_i(t) + \mathbf{T}(t) - f(\mathbf{p}_i(t))\|^2.$$

Apply the transformation to points  $\mathbf{p}_i(t)$ :

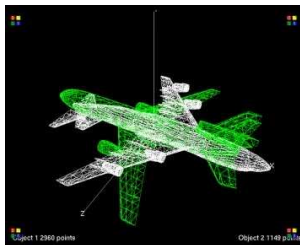
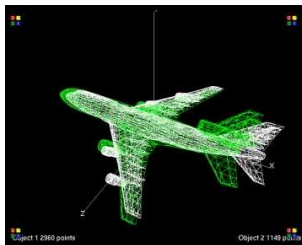
$$\mathbf{p}_i(t+1) = s(t) \mathbf{R}(t) \mathbf{p}_i(t) + \mathbf{T}(t).$$

until  $E(t)$  or  $E(t) - E(t-1)$  is small enough.

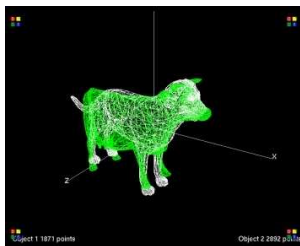
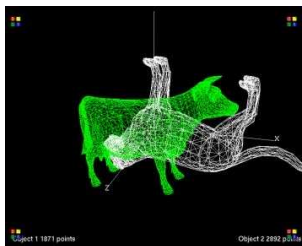
Notes:

- Like other iterative algorithms, the choice of initial parameters affects the registration results.
- The algorithm can also match lines, curves, surfaces, etc. instead of points (see [BM92]).

Examples:



demo 1



demo 2

(a) initial

(b) registered

# Summary

Problems and algorithms discussed:

problem	corresp.	transform.	algorithm
affine registration	known	affine	linear least square
relative orientation	known	similarity	eigen decomposition
relative orientation	unknown	similarity	Iterative Closest Point

- Registration with known correspondence is quite straightforward.
- Registration with unknown correspondence needs to guess possible correspondence and iteratively refines the solution.
- Iterative optimization algorithms typically need good initial conditions to yield good solutions.

# Exercise

- (1) Show that the 3D rotation matrix is an orthonormal matrix, i.e.,

$$\mathbf{R}^\top \mathbf{R} = \mathbf{I}.$$

In particular, if we write  $\mathbf{R} = [\mathbf{R}_1^\top \ \mathbf{R}_2^\top \ \mathbf{R}_3^\top]^\top$  where  $\mathbf{R}_1^\top$ ,  $\mathbf{R}_2^\top$ , and  $\mathbf{R}_3^\top$  are the row matrices of  $\mathbf{R}$ , then

$$\mathbf{R}_i^\top \mathbf{R}_j = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases}$$

- (2) Reassemble affine transformation Eq. 16 into Eq. 18.

# Further Readings

- Read Appendix A: Eigen Decomposition.

# A Eigen Decomposition

Eigenvalue equation has this form

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v} \quad (35)$$

- $\mathbf{A}$  is a matrix.
- $\mathbf{v}$  is a column vector called **eigenvector**.
- $\lambda$  is a scalar called **eigenvalue** of  $\mathbf{A}$  corresponding to  $\mathbf{v}$ .
- If  $\mathbf{A}$  is  $n \times n$  matrix, then there are  $n$  eigenvectors and eigenvalues.
- Solving for the  $n$   $\mathbf{v}_i$  and  $\lambda_i$  is called **eigen decomposition**.
- There are standard algorithms for eigen decomposition.

## Useful Properties

The eigenvectors  $\mathbf{v}_i$  are orthonormal vectors

$$\mathbf{v}_i^\top \mathbf{v}_j = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \quad (36)$$

Trace of  $\mathbf{A}$  is

$$\text{tr}(\mathbf{A}) = \sum_{i=1}^n \lambda_i. \quad (37)$$




Determinant of  $\mathbf{A}$  is

$$\det(\mathbf{A}) = \prod_{i=1}^n \lambda_i. \quad (38)$$





Eigenvalues of  $\mathbf{A}^k$  are  $\lambda_1^k, \dots, \lambda_n^k$ .

◀ return

# Reference I

-  P. J. Besl and N. D. McKay.  
A method for registration of 3-D shapes.  
*IEEE Trans. on Pattern Analysis and Machine Intelligence*,  
14(2):239–256, 1992.
-  J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes.  
*Computer Graphics: Principles and Practice*.  
Addison-Wesley, 2nd edition, 1987.
-  B. K. P. Horn, H. M. Hilden, and S. Negahdaripour.  
Closed-form solution of absolute orientation using orthonormal  
matrices.  
*J. Optical Society of America A*, 5(7):1127–1135, 1988.

## Reference II

-  R. Jain, R. Kasturi, and B. G. Schunck.  
*Machine Vision*.  
McGraw-Hill, 1995.
-  J. B. Kuipers.  
*Quaternions and Rotation Sequences*.  
Princeton University Press, 1999.
-  W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery.  
*Numerical Recipes in C*.  
Cambridge University Press, 2nd edition, 1992.
-  Rodrigues' rotation formula,  
[mathworld.wolfram.com/rodriguesrotationformula.html](http://mathworld.wolfram.com/rodriguesrotationformula.html).

## Reference III



L. G. Shapiro and G. C. Stockman.

*Computer Vision.*

Prentice-Hall, 2001.



Z. Y. Zhang.

Iterative point matching for registration of free-form curves and surfaces.

*Int. Journal of Computer Vision*, 13(2):119–152, 1994.