Registration

CS5240 Theoretical Foundations in Multimedia

Leow Wee Kheng

Department of Computer Science
School of Computing
National University of Singapore
Motivation

Skulls can be defective for various reasons:

- patient’s skull
- patient’s skull
- victim’s skull

accident
congenital deformity
violence
To reconstruct the normal appearance of a defective model, need to know how the defective model differs from a normal skull.

Registration: align and match the two models.
Rigid Registration

Spatially align two 3D models without shape change.

reference  target  registration result
Let $p_i$, $i = 1, \ldots, n$, denote points on reference model $F$, and $q_i$ denote corresponding points on target model $T$.

Rigid registration means finding the transformation $T$, without shape change, that minimizes the error

$$E = \sum_{i=1}^{n} \| q_i - T(p_i) \|^2. \quad (1)$$

Possible transformations:

- **rigid transformation**: rotation, translation
- **similarity transformation**: scaling, rotation, translation
Let $p_i = (x_i, y_i, z_i)$ denote points on reference model $F$, and $q_i = (u_i, v_i, w_i)$ denote corresponding points on target model $T$.

Denote scaling $s$, rotation $R$, and translation $T$:

$$q_i = sR p_i + T, \quad \text{for } i = 1, \ldots, n.$$  \hfill (2)

Rotation matrix $R$ is orthogonal.
An $n \times n$ matrix $M$ is orthogonal if

$$M^\top M = MM^\top = I.$$  

(3)

Let $M = [m_1 \cdots m_n]$. Then, $M$ is orthogonal if $m_i$ are orthonormal:

$$m_i^\top m_j = \begin{cases} 
1 & \text{if } i = j, \\
0 & \text{if } i \neq j.
\end{cases}$$  

(4)

- $n = 1$: identity, reflection.
- $n = 2$: identity, 2-D rotation, 2-D reflection, permutation.

$$\begin{bmatrix}
\cos \theta & -\sin \theta \\
\sin \theta & \cos \theta
\end{bmatrix} \quad \begin{bmatrix}
\cos \theta & \sin \theta \\
\sin \theta & -\cos \theta
\end{bmatrix} \quad \begin{bmatrix}
0 & 1 \\
1 & 0
\end{bmatrix}$$

rotation reflection permutation

- $n = 3$: identity, 3-D rotation, 3-D reflection, permutation, etc.
For a point \( p = (x, y, z) \), its homogeneous coordinates \( \tilde{p} \) is \((cx, cy, cz, c)\) for \( c \neq 0 \).

Using homogeneous coordinates, Eq. 2 for 3-D case can be written as

\[
\tilde{q}_i = M \tilde{p}_i, \text{ for } i = 1, \ldots, n.
\] (5)

which is

\[
\begin{bmatrix}
  u_i \\
  v_i \\
  w_i \\
  1
\end{bmatrix} =
\begin{bmatrix}
  s r_{11} & s r_{12} & s r_{13} & t_x \\
  s r_{21} & s r_{22} & s r_{23} & t_y \\
  s r_{31} & s r_{32} & s r_{33} & t_z \\
  0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x_i \\
  y_i \\
  z_i \\
  1
\end{bmatrix}.
\] (6)

**Caution:**
Even though Eq. 6 is a linear equation, it cannot be solved using simple linear least squares. Why?
**Similarity Transformation Algorithm** [Horn88]

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

\[ \mathbf{r}_i = \mathbf{p}_i - \bar{\mathbf{p}}, \quad \mathbf{r}'_i = \mathbf{q}_i - \bar{\mathbf{q}}' \] (7)

where

\[ \bar{\mathbf{p}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{p}_i, \quad \bar{\mathbf{q}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{q}_i. \] (8)

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

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

\[ s^2 = \frac{\sum_{i=1}^{n} \| \mathbf{r}'_i \|^2}{\frac{1}{n} \sum_{i=1}^{n} \| \mathbf{r}_i \|^2} \] (9)
**Step 3:** Compute rotation matrix as follows:

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

$$\mathbf{M} = \sum_{i=1}^{n} \mathbf{r}_i' \mathbf{r}_i^\top.$$  \hspace{1cm} (10)

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

$$\mathbf{R} = \mathbf{M} \mathbf{Q}^{-1/2}$$ \hspace{1cm} (11)

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

Perform eigendecomposition of $\mathbf{Q}$ to obtain (Homework)

$$\mathbf{Q} = \lambda_1 \mathbf{v}_1 \mathbf{v}_1^\top + \lambda_2 \mathbf{v}_2 \mathbf{v}_2^\top + \lambda_3 \mathbf{v}_3 \mathbf{v}_3^\top,$$ \hspace{1cm} (12)

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

The inverse square root of eigensystem can be easily computed as

$$\mathbf{Q}^{-1/2} = \frac{1}{\sqrt{\lambda_1}} \mathbf{v}_1 \mathbf{v}_1^\top + \frac{1}{\sqrt{\lambda_2}} \mathbf{v}_2 \mathbf{v}_2^\top + \frac{1}{\sqrt{\lambda_3}} \mathbf{v}_3 \mathbf{v}_3^\top.$$ \hspace{1cm} (13)
**Step 4:** Given $s$ and $R$, we can now compute $T$.

$$T = \bar{q} - s R \bar{p}. \quad (14)$$

Notes:

- After obtaining $s$, $R$, and $T$, apply Eq. 2 to transform $p_i$ of reference model to new position $p_i'$:

$$p_i' = s R p_i + T. \quad (15)$$

Now, the transformed reference is aligned to the target.

- The $s$, $R$, and $T$ obtained minimize the error

$$E = \sum_{i=1}^{n} \| q_i - p_i' \|^2 = \sum_{i=1}^{n} \| q_i - s R p_i - T \|^2 \quad (16)$$

subject to $R$ is a rotation (orthogonal) matrix.
Notes:

- If the points $p_i$ or $q_i$ are degenerate, use [Umeyama91] instead.
  Examples:
  3-D points lie on 2-D plane,
  2-D points lie on 1-D line.
In general, correspondence between reference and target is **unknown**.

\[ p_i \] denote points on reference model \( F \),
\[ q_j \] denote points on target model \( T \).
Which \( p_i \) corresponds to which \( q_j \)?
Let $f : F \rightarrow T$ denote the correspondence function that maps a point $p_i$ in $F$ to a point $q_j$ in $T$.

Then, registration without known correspondence means finding the transformation $T(p_i)$ and correspondence function $f(p_i)$ that minimizes the error

$$E = \sum_{i=1}^{n} \| f(p_i) - T(p_i) \|^2.$$  \hspace{1cm} (17)

Caution: This problem definition admits a useless trivial solution. What is it?
Iterative Closest Point (ICP) [Besl92]:

- Make educated guess of $f(p_i)$, e.g., closest point.
- Iteratively update estimate.

**Iterative Closest Point**

Inputs: $p_i$, $i = 1, \ldots, n$ and target $T$.

1. Initialize $p'_i = p_i$ for each $i$.
2. Repeat until convergence:
   2.1 Find closest point $f(p'_i)$ on target $T$ for each $p'_i$.
   2.2 Compute $s$, $R$, $T$ from $p_i$ to $f(p'_i)$ for all $i$.
   2.3 Apply transformation: $p'_i = sR p_i + T$, for each $i$.

Outputs: $s$, $R$, $T$, $p'_i$, $i = 1, \ldots, n$.

What kind of optimization algorithm is this?
ICP can always converge [Besl92], typically to a local minimum.
Nonrigid Registration

Warp / deform reference to match target as best as possible.

reference  target  nonrigid registration  rigid registration
Nonrigid registration can produce either approximating surface or interpolating surface.
Nonrigid Registration

Given $p_i, i = 1, \ldots, n$, on reference model $F$ and known corresponding point $q_j$ on target model $T$, find the nonrigid transformation $T$ that minimizes the error

$$E = \sum_{i=1}^{n} \| q_i - T(p_i) \|^2$$

subject to appropriate constraints, e.g., smoothness.

- $E > 0$ for approximation; $E = 0$ for interpolation.
- For specific problem or algorithm, specify the specific error requirement and constraints.
- Revise the above appropriately if correspondence is unknown (Homework).
Affine Transformation

Affine transformation includes scaling, rotation, translation, reflection and shearing.

Affine transformation

- preserves straight lines and planes,
- but may not preserve distance between points, angle between lines.

So, it can produce shape change.
3D affine transformation $A$ maps homogeneous coordinates $\tilde{p}_i$ to corresponding point $\tilde{q}_i$:

$$\tilde{q}_i = A\tilde{p}_i$$

$$\begin{bmatrix} u_i \\ v_i \\ w_i \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$ (18)

Each component corresponds to one set of linear equations:

$$u = Da$$

$$\begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & z_n & 1 \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{14} \end{bmatrix}$$. (19)

So, $A$ can be solved using linear least squares one component at a time.
The three components can be combined in a single equation as

\[ U = D A^\top \]

\[ \begin{bmatrix} u_1^\top & 1 \\ \vdots & \vdots \\ u_n^\top & 1 \end{bmatrix} = \begin{bmatrix} p_i^\top & 1 \\ \vdots & \vdots \\ p_n^\top & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{21} & a_{31} & 0 \\ a_{12} & a_{22} & a_{32} & 0 \\ a_{13} & a_{23} & a_{33} & 0 \\ a_{14} & a_{24} & a_{34} & 1 \end{bmatrix}. \quad (20) \]

Then, the solution for \( A \) is given by (Homework)

\[ A^\top = (D^\top D)^{-1} D^\top U. \quad (21) \]

Solving for \( A \) minimize

\[ E = \sum_{i=1}^{n} \| \tilde{q}_i - A \tilde{p}_i \|^2 = \| DA^\top - U \|_F^2, \quad (22) \]

where \( \| \cdot \|_F \) is the Frobenius norm.
For $m \times n$ matrix $B = [b_{ij}]$, the Frobenius norm $\|B\|_F$ of $B$ is given by

$$\|B\|_F^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} b_{ij}^2 = \sum_{i=1}^{\min(m,n)} \sigma_i^2 \quad (23)$$

where $\sigma_i$ are the singular values of $B$.

The warped surface point $\tilde{p}_i' = A\tilde{p}_i \neq \tilde{q}_i$.

Therefore, affine transformation produces an approximating surface.
Nonrigid ICP

Nonrigid ICP [Amberg2007]

- Apply locally affine transformation.
- Minimize distance to landmark points.

\[ p_i \]: position of mesh vertex \( i \) on reference surface.
Each mesh vertex \( i \) has its own affine transformation \( A_i \).
As for ICP, find the closest point \( u_i \) on target surface to \( p_i \). So, the error in term of distance to the target surface is

\[
E_d = \sum_{i=1}^{n} w_i \| A_i p_i - u_i \|^2.
\]  

(24)

- \( p_i \) are homogeneous coordinates.
- Weight \( w_i = 1 \) for vertices with correspondence; 0, otherwise.
- Each \( A_i \) has 12 unknown parameters, but each pair of \( p_i \) and \( u_i \) gives only 3 equations. Not enough.
- Need additional constraints.

Let \( \mathcal{L} = \{ (p_i, q_i) \} \) contains known corresponding landmarks. Then, the error in terms of distances to corresponding landmarks is

\[
E_l = \sum_{(p_i, q_i) \in \mathcal{L}} \| A_i p_i - q_i \|^2.
\]  

(25)
Affine transformations of neighbours should not differ too much. Difference of affine transformations between connected neighbours is

$$E_s = \sum_{(i,j) \in \mathcal{E}} \| (A_i - A_j) G \|_F^2, \tag{26}$$

- $\mathcal{E}$ is the set of mesh edges.
- $G = \text{diag}(1, 1, 1, \gamma)$ is a weight matrix.
- $\| \cdot \|_F$ is the Frobenius norm.

Total error to be minimized is

$$E = E_d + \alpha E_s + \beta E_l. \tag{27}$$
To solve for the $A_i$, $i = 1, \ldots, n$, that minimize $E$, have to arrange $A_i$ in a similar way as in Eq. 20.

Let $W = \text{diag}(w_1, \ldots, w_n)$, $A = [A_1 \cdots A_n]^\top$, $U = [u_1 \cdots u_n]^\top$,

$$D = \begin{bmatrix}
p_1^\top & 0 & \cdots & 0 \\
0 & p_2^\top & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & p_n^\top
\end{bmatrix}.$$

Then, $E_d$ (Eq. 24) becomes (Homework)

$$E_d = \|W(DA - U)\|_F^2. \quad (28)$$

Similarly, let $Q = [q_1 \cdots q_n]^\top$, $D_l$ contain rows of $D$ with landmarks. Then, $E_l$ (Eq. 25) becomes (Homework)

$$E_l = \|D_lA - Q\|_F^2. \quad (29)$$
To rearrange $E_s$ (Eq. 26), introduce incidence matrix $M$:

- Each column corresponds to a mesh vertex.
- Each row corresponds to a mesh edge that connects two vertices.
- If edge $r$ connects vertices $i$ and $j$, where $i > j$, then set $M[r, i] = 1$, $M[r, j] = -1$.

Then, $E_s$ (Eq. 26) becomes (Homework)

$$E_s = \| (M \otimes G) A \|_F^2,$$  \hspace{1cm} (30)

where $\otimes$ is the Kronecker product.
Kronecker Product

Let $A$ be a $m \times n$ matrix and $B$ be a $p \times q$ matrix. Then, the Kronecker product of $A$ and $B$ is

$$A \otimes B = \begin{bmatrix}
    a_{11}B & \cdots & a_{1n}B \\
    \vdots & \ddots & \vdots \\
    a_{m1}B & \cdots & a_{mn}B
\end{bmatrix}.$$  \hspace{1cm} (31)

That is,

$$A \otimes B = \begin{bmatrix}
    a_{11}b_{11} & \cdots & a_{11}b_{1q} & \cdots & a_{1n}b_{11} & \cdots & a_{1n}b_{1q} \\
    \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\
    a_{11}b_{p1} & \cdots & a_{11}b_{pq} & \cdots & a_{1n}b_{p1} & \cdots & a_{1n}b_{pq} \\
    \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\
    a_{m1}b_{11} & \cdots & a_{m1}b_{1q} & \cdots & a_{mn}b_{11} & \cdots & a_{mn}b_{1q} \\
    \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\
    a_{m1}b_{p1} & \cdots & a_{m1}b_{pq} & \cdots & a_{mn}b_{p1} & \cdots & a_{mn}b_{pq}
\end{bmatrix}.  \hspace{1cm} (32)
Then, the total error $E$ (Eq. 27) becomes

$$E = \left\| \begin{bmatrix} WD \\ \alpha M \otimes G \\ \beta D_l \end{bmatrix} A - \begin{bmatrix} WU \\ 0 \\ Q \end{bmatrix} \right\|^2_F,$$

which can be written in the form

$$E = \| BA - C \|^2_F.$$  \hspace{1cm} (33)

Then, as for Eq. 20, the $A$ that minimizes $E$ is

$$A = (B^\top B)^{-1} B^\top C.$$  \hspace{1cm} (35)
Nonrigid ICP

Inputs: $\mathbf{p}_i$, $i = 1, \ldots, n$, target $T$.

1. Initialize $\alpha$ and $\mathbf{A}_i$ for each $i$.
2. Initialize $\mathbf{p}_i' = \mathbf{p}_i$ for each $i$.
3. Repeat:
   3.1 Repeat until convergence:
      3.1.1 Find closest point $\mathbf{u}_i$ on target $T$ for each $\mathbf{p}_i'$.
      3.1.2 Compute $\mathbf{A}_i$ from $\mathbf{p}_i$ to $\mathbf{u}_i$ for all $i$.
      3.1.3 Apply transformation: $\mathbf{p}_i' = \mathbf{A}_i \mathbf{p}_i$ for each $i$.
   3.2 Decrease $\alpha$.

Outputs: $\mathbf{A}_i$, $\mathbf{p}_i'$, $i = 1, \ldots, n$. 

Thin plate spline (TPS) warping [Bookstein89]

- Analogous to bending of a thin metal sheet.
- Impose smoothness by minimizing bending energy.

TPS maps points $\mathbf{p}_i = [x_{i1} \cdots x_{id}]^\top$ on reference surface to desired positions $\mathbf{q}_i = [v_{i1} \cdots v_{id}]^\top$ exactly.
Consider $j$-th component $v_{ij}$ of $q_i$, dropping subscript $j$ for notational simplicity.

TPS maps $p_i$ to $v_i$ through a mapping function $f(p_i) = v_i$ that minimizes a bending energy $E_d(f)$.

General form of bending energy with order-$m$ derivatives is [Wahba90]

$$E_d(f) = \sum_{k_1 + \cdots + k_d = m} \frac{m!}{k_1! \cdots k_d!} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \left( \frac{\partial^m f}{\partial x_1^{k_1} \cdots \partial x_d^{k_d}} \right)^2 \, dx_1 \cdots dx_d. \tag{36}$$

With $d = 2$, $m = 2$, and denoting $f_{x_1 x_2}^2 = \partial^2 f / \partial x_1 \partial x_2$,

$$E_2(f) = \int \int (f_{x_1 x_1}^2 + 2f_{x_1 x_2}^2 + f_{x_2 x_2}^2) \, dx_1 \, dx_2. \tag{37}$$
With $d = 3$, $m = 2$,

$$E_3(f) = \int \int \int (f^2 x_1 + f^2 x_2 + f^2 x_3 + 2f^2 x_1 x_2 + 2f^2 x_1 x_3 + 2f^2 x_2 x_3) \, dx_1 \, dx_2 \, dx_3. \quad (38)$$

Denote the homogeneous coordinates of a general point $x$ as $	ilde{x} = [1 \, x_1 \, \cdots \, x_d]^\top$.

The function $f$ that minimizes Eq. 36 takes the form [Bookstein89]

$$f(\tilde{x}) = a^\top \tilde{x} + \sum_{i=1}^{n} w_i U(\|x - p_i\|). \quad (39)$$

- $a = [a_0 \, a_1 \, \cdots \, a_d]^\top$ are affine parameters.
- $w = [w_1 \, \cdots \, w_n]^\top$ are weights.
- $U(r)$ is increasing function of distance $r$.
- 1st term is affine (linear) transformation.
- 2nd term is nonlinear warping.
$U(r)$ can take several forms, e.g., $U(r) = r^2 \log r^2$, $U(r) = r^2 \log r$.

![Graph showing two curves: $r^2 \log r^2$ and $r^2 \log r$.]

For interpolation, want to find $\mathbf{a}$ and $\mathbf{w}$ such that, for each $\mathbf{p}_i$,

$$v_i = f(\mathbf{\tilde{p}}_i) = \mathbf{a}^\top \mathbf{\tilde{p}}_i + \sum_{j=1}^{n} w_j U(\|\mathbf{p}_i - \mathbf{p}_j\|).$$

(40)
Define \( n \times (d + 1) \) matrix \( P \) and \( n \times n \) matrix \( K \), with \( r_{ij} = \| p_i - p_j \| \), as

\[
P = \begin{bmatrix} \tilde{p}_1^\top \\ \vdots \\ \tilde{p}_n^\top \end{bmatrix}, \quad K = \begin{bmatrix} 0 & U(r_{12}) & \cdots & U(r_{1n}) \\ U(r_{21}) & 0 & \cdots & U(r_{2n}) \\ \vdots & \vdots & \ddots & \vdots \\ U(r_{n1}) & U(r_{n2}) & \cdots & 0 \end{bmatrix}, \quad (41)
\]

and \( v = [v_1 \cdots v_n]^\top \).

Then, Eq. 40 can be written in matrix form (Homework)

\[
\begin{bmatrix} K & P \\ P^\top & 0 \end{bmatrix} \begin{bmatrix} w \\ a \end{bmatrix} = \begin{bmatrix} v \\ 0 \end{bmatrix}. \quad (42)
\]

The bottom rows of Eq. 42 impose the constraints

\[
\sum_{i=1}^{n} w_i = 0, \quad \sum_{i=1}^{n} w_i p_i = 0. \quad (43)
\]
Let
\[
L = \begin{bmatrix} K & P \\ P^\top & 0 \end{bmatrix}.
\] (44)

[Powell95] shows that, if the points are in general position, then \(P\) has independent columns and \(L\) is non-singular. So,

\[
\begin{bmatrix} w \\ a \end{bmatrix} = L^{-1} \begin{bmatrix} v \\ 0 \end{bmatrix}.
\] (45)

If \(L\) is singular, then use pseudo-inverse of \(L\) instead of \(L^{-1}\) in Eq. 45.

Notes:

- Eq. 45 is computed for each dimension of \(q_i\) with the corresponding \(v\); same \(L\).
- After obtaining \(w\) and \(a\) for each dimension, any point \(x\) can be mapped to \(f(\tilde{x})\) by applying Eq. 39 to each dimension.
General Position

A set of at least $d + 1$ points in $d$-D Euclidean space is in general position if no hyperplane of $(d − 1)$-D contains more than $d$ points.

Examples:

- In 2-D space, 2 points define a 1-D line.
  If 3 points lie on a 1-D line, then they are not in general position.
  3 or more points are in general position if no 1-D line contains more than 2 points.

- In 3-D space, 3 points define a 2-D plane.
  If 4 points lie on a 2-D plane, then they are not in general position.
  4 or more points are in general position if no 2-D plane contains more than 3 points.
It is possible to solve for all dimensions at the same time:
Pack $w_j$, $a_j$ and $v_j$ for all dimensions $j = 1, \ldots, d$ together:
\[
    w = [w_1 \cdots w_d], \quad a = [a_1 \cdots a_d], \quad v = [v_1 \cdots v_d]. \tag{46}
\]
and apply Eq. 45.

Notes:

- Previous method solves different dimensions separately.
- This method solves all dimensions simultaneously.
- This method has no advantage over previous method because different dimensions are independent.
Notes:

- If $L$ is non-singular, TPS satisfies positional constraints exactly; positional constraints are **hard constraints**, $p_i$ is mapped to $q_i$ exactly for each $i$.

- If $L$ is singular, TPS satisfies positional constraints approximately; positional constraints are **soft constraints**, $p_i$ is mapped to $q_i$ approximately for each $i$.

- $L$ is an $(n + d + 1) \times (n + d + 1)$ matrix. More positional constraints (larger $n$) lead to more computation.

- If $w = 0$, then $p_i$ is mapped to $q_i$ by affine transformation only.
Examples of TPS registration.

reference  target 1  target 2  target 3

result 1  result 2  result 3
Laplacian Deformation [Sorkine2004, Masuda2006]

- Preserve shape: local surface curvature and surface normal.

Laplacian deformation maps points $p_i$ on reference surface to desired positions $d_i$ exactly.
Discrete Laplacian operator (or Laplace-Beltrami operator) $L(p_i)$ estimates surface curvature and normal at vertex $i$ as:

$$L(p_i) = \sum_{j \in N_i} w_{ij} (p_i - p_j) \quad (47)$$

where $N_i$ is the set of connected neighbours of vertex $i$.

The weight $w_{ij}$ can be cotangent weight [Masuda2006] or equal weight $w_{ij} = \frac{1}{|N_i|}$.  \(48\)
With equal weight, Laplacian operator becomes

\[
L(p_i) = p_i - \frac{1}{N_i} \sum_{j \in N_i} p_j.
\]  
(49)

Shape is preserved by minimizing the difference of Laplacian operators \(L(p_i^0)\) before and \(L(p_i)\) after deformation:

\[
\|L(p_i) - L(p_i^0)\|^2,
\]  
(50)

This difference is organized into a matrix form for all mesh vertices:

\[
\|Ax - b\|^2.
\]  
(51)

- **A** is a \(3n \times 3n\) matrix of Laplacian constraints.
- **x** is a \(3n \times 1\) vector of unknown positions \(x_i\) of mesh vertices, \(x = [x_1^T \cdots x_n^T]^T\).
- **b** is a \(3n \times 1\) vector that contains \(L(p_i^0)\) before deformation.
Example: Suppose vertex 2 is a connected neighbour of vertex 1. Then, \( Ax - b \) is

\[
\begin{bmatrix}
1 & 0 & 0 & -1/N_1 & 0 & 0 & 0 & \cdots & 0 \\
0 & 1 & 0 & 0 & -1/N_1 & 0 & 0 & \cdots & 0 \\
0 & 0 & 1 & 0 & 0 & -1/N_1 & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots
\end{bmatrix}
\begin{bmatrix}
x_1 \\
y_1 \\
z_1 \\
x_2 \\
y_2 \\
z_2 \\
\vdots
\end{bmatrix}
- \begin{bmatrix}
l_{1x} \\
l_{1y} \\
l_{1z} \\
\vdots
\end{bmatrix}
\]

where \( l_{1x}, l_{1y}, l_{1z} \) are the components of \( L(p_0^1) \).
The positional constraints are organised into a matrix form

$$C \mathbf{x} = \mathbf{d}$$ \hfill (52)

Without loss of generality, we can arrange the mesh vertices with positional constraints as vertices 1 to $m < n$.

- $C$ is a $3m \times 3n$ matrix of positional constraints.
- Top $3m$ elements of $\mathbf{x}$ are vertices with positional constraints, bottom $3(n - m)$ elements are those without positional constraints.
- $\mathbf{d}$ is a $3m \times 1$ vector that contains desired positions $\mathbf{d}_i$ of vertices with positional constraints, $\mathbf{d} = [\mathbf{d}_1^\top \cdots \mathbf{d}_m^\top]^\top$. 
Example: Suppose only vertices 1 and 2 have positional constraints $d_1$ and $d_2$. Then, $C \mathbf{x} = \mathbf{d}$ is

\[
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\
\end{bmatrix}
\begin{bmatrix}
x_1 \\
y_1 \\
z_1 \\
x_2 \\
y_2 \\
z_2 \\
\vdots \\
x_n \\
y_n \\
z_n \\
\end{bmatrix}
= 
\begin{bmatrix}
d_{1x} \\
d_{1y} \\
d_{1z} \\
d_{2x} \\
d_{2y} \\
d_{2z} \\
\end{bmatrix}.
\]

Laplacian deformation minimizes $\|A \mathbf{x} - \mathbf{b}\|^2$ subject to the equality constraint $C \mathbf{x} = \mathbf{d}$. 
Equality constrained least squares problem takes the form:

$$\min_x \| Ax - b \|^2$$

subject to $Cx = d$.

There are two ways to solve this problem [Golub96]:

- Lagrange multiplier method: approximation method (Appendix)
- QR factorization method: exact method
QR Decomposition

QR Decomposition (QR factorization) decomposes square matrix $A$ as

$$A = QR,$$

where $Q$ is an orthogonal matrix and $R$ in an upper triangular matrix.

If $A$ is invertible and we make the diagonal elements of $R$ positive, then $Q$ and $R$ are unique.

If $A$ is $m \times n$, with $m \geq n$, then $Q$ and $R$ can be written as

$$Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}, \quad R = \begin{bmatrix} R_1 \\ 0 \end{bmatrix}. \quad (55)$$

- $Q_1$ is $m \times n$, with orthogonal columns.
- $Q_2$ is $m \times (m - n)$, with orthogonal columns.
- $R_1$ is $n \times n$ upper triangular.
- $0$ is $(m - n) \times n$ zero matrix.
Then,

\[ A = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = Q_1 R_1. \] (56)

This is called \textit{thin QR factorization} or \textit{reduced QR factorization}.

If \( A \) is of rank \( n \) and the diagonal elements of \( R_1 \) are positive, then \( Q_1 \) and \( R_1 \) are unique.
C is an $m \times n$ matrix, with $m$ constraints and $n$ data points, $m < n$. Then, $C^\top$ has QR factorization

$$C^\top = QR,$$  \hspace{1cm} (57)

where $Q$ is orthogonal, $R$ is upper triangular, and

$$Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}, \quad R = \begin{bmatrix} R_1 \\ 0 \end{bmatrix}. \hspace{1cm} (58)$$

Define vectors $u$ and $v$ such that

$$x = Q \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}. \hspace{1cm} (59)$$

Then, the objective function becomes

$$\|Ax - b\|^2 = \|AQ_1 u + AQ_2 v - b\|^2. \hspace{1cm} (60)$$
The left-hand side of the equality constraints is

\[ Cx = R^\top Q^\top x = R^\top \begin{bmatrix} u \\ v \end{bmatrix} = R_1^\top u. \]  

(61)

So, \( u \) can be obtained by solving

\[ R_1^\top u = d. \]  

(62)

Substituting the solution \( u \) into Function 60 gives

\[ \| AQ_2v - (b - AQ_1u) \|^2, \]  

(63)

which can be solved for \( v \) by unconstrained linear least squares.

Finally, \( x \) can be obtained by substituting \( u \) and \( v \) into Eq. 59.
Notes:

- This method requires solving a QR decomposition and two unconstrained linear least squares.
- If $\mathbf{R}_1$ is full rank, then the equality constraint is satisfied exactly.
- So, this method can produce exact solution.
Back to Laplacian Deformation

The equality constraint $\mathbf{C} \mathbf{x} = \mathbf{d}$ is

$$
\begin{bmatrix}
\mathbf{I}_{3m} & 0 \\
\mathbf{x}_1 \\
\vdots \\
\mathbf{x}_m \\
\mathbf{x}_{m+1} \\
\vdots \\
\mathbf{x}_n
\end{bmatrix}
= 
\begin{bmatrix}
\mathbf{d}_1 \\
\vdots \\
\mathbf{d}_m
\end{bmatrix}.
$$

That is,

$$
\mathbf{C} = 
\begin{bmatrix}
\mathbf{I}_{3m} & 0 \\
\end{bmatrix}.
$$

Then, the QR decomposition of $\mathbf{C}^\top$ is (Homework)

$$
\mathbf{C}^\top = 
\begin{bmatrix}
\mathbf{I}_{3m} & 0 \\
0 & \mathbf{I}_{3(n-m)}
\end{bmatrix}
\begin{bmatrix}
\mathbf{I}_{3m} \\
0
\end{bmatrix}.
$$
So, Eq. 62 becomes

\[ \mathbf{I}_{3m}^\top \mathbf{u} = \mathbf{u} = \mathbf{d}. \] (67)

Moreover,

\[ \mathbf{Q}_1 \mathbf{u} = \begin{bmatrix} \mathbf{I}_{3m} \\ \mathbf{0} \end{bmatrix} \mathbf{d} = \begin{bmatrix} \mathbf{d} \\ \mathbf{0} \end{bmatrix}, \] (68)

and

\[ \mathbf{Q}_2 \mathbf{v} = \begin{bmatrix} \mathbf{0} \\ \mathbf{I}_{3(n-m)} \end{bmatrix} \mathbf{v} = \begin{bmatrix} \mathbf{0} \\ \mathbf{v} \end{bmatrix}, \] (69)

Organize \( \mathbf{A} \) as

\[ \mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 \end{bmatrix}, \] (70)

where \( \mathbf{A}_1 \) is a \( 3n \times 3m \) matrix and \( \mathbf{A}_2 \) is a \( 3n \times 3(n - m) \) matrix. Then,

\[ \mathbf{A} \mathbf{Q}_1 \mathbf{u} = \mathbf{A}_1 \mathbf{d}, \quad \mathbf{A} \mathbf{Q}_2 \mathbf{v} = \mathbf{A}_2 \mathbf{v}. \] (71)

So, Function 63 can be simplified as

\[ \| \mathbf{A}_2 \mathbf{v} - (\mathbf{b} - \mathbf{A}_1 \mathbf{d}) \|^2. \] (72)
Minimization of Function 72 with linear least squares yields

\[ v = (A_2^\top A_2)^{-1} A_2^\top (b - A_1 d). \] (73)

Finally,

\[ x = Q_1 u + Q_2 v = \begin{bmatrix} d \\ v \end{bmatrix}. \] (74)

Notes:

- Computation of QR decomposition of \( C^\top \) is not needed.
- Many matrix multiplications are avoided because of the special \( C \).
- Positions of vertices with positional constraints are given as \( d \); do not require computation.
- Eq. 73 is evaluated only for vertices without positional constraints.
- So, more positional constraints lead to less computation.
Examples of registration by Laplacian deformation.
Examples of Laplacian deformation of blood vessel model.

(a) With ordinary Laplacian operator.
(b) With rotation-invariant Laplacian operator [Masuda2006].
### Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Registration</th>
<th>Surface</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICP</td>
<td>rigid</td>
<td>approximation</td>
</tr>
<tr>
<td>Nonrigid ICP</td>
<td>nonrigid</td>
<td>approximation</td>
</tr>
<tr>
<td>TPS</td>
<td>nonrigid</td>
<td>interpolation</td>
</tr>
<tr>
<td>Laplacian deformation</td>
<td>nonrigid</td>
<td>interpolation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICP</td>
<td>alternating op. with eigendecomposition</td>
</tr>
<tr>
<td>Nonrigid ICP</td>
<td>alternating op. with linear least squares</td>
</tr>
<tr>
<td>TPS</td>
<td>linear least squares</td>
</tr>
<tr>
<td>Laplacian deformation</td>
<td>equality constrained least squares</td>
</tr>
</tbody>
</table>
Probing Questions

- Is it possible to extend nonrigid ICP to produce interpolating surface. Is yes, how? If no, why?
- Nonrigid ICP minimizes distances of mesh vertices to target surface and to known landmarks. Consider any mesh vertex $i$. If its closest target surface point $u_i$ and its corresponding landmark $q_i$ are very different, what will be the registration result?
- Is it possible to extend TPS or Laplacian deformation to produce approximating surface? Is yes, how? If no, why?
- Laplacian deformation uses equality constrained least squares to achieve surface interpolation. Nonrigid ICP uses alternating optimization with linear least squares but achieve surface approximation instead of interpolation. Why is it that TPS can achieve surface interpolation using simple linear least squares? How does TPS satisfies the positional hard constraints?
1. Describe the essence of ICP, nonrigid ICP, TPS and Laplacian deformation each in one sentence.

2. Define the problem of nonrigid registration without correspondence.

3. Define the problems solved by nonrigid ICP, TPS and Laplacian deformation.

4. Suppose the eigendecomposition of an $m \times m$ matrix $A$ is $V \Lambda V^\top$, where $V$ contains the eigenvectors $v_j$ and $\Lambda$ contains the eigenvalues $\lambda_j$, $j = 1, \ldots, m$. Show that

$$A = \sum_{j=1}^{m} \lambda_j v_j v_j^\top.$$
5. What is the solution of $\mathbf{u} = \mathbf{D} \mathbf{a}$ (Eq. 19)? Based on this solution, show that the solution of $\mathbf{U} = \mathbf{D} \mathbf{A}^\top$ (Eq. 20) is

$$
\mathbf{A}^\top = (\mathbf{D}^\top \mathbf{D})^{-1} \mathbf{D}^\top \mathbf{U}.
$$

6. Show that Eq. 24 can be written as

$$
E_d = \| \mathbf{W}(\mathbf{D} \mathbf{A} - \mathbf{U}) \|_F^2.
$$

7. Show that Eq. 25 can be written as

$$
E_l = \| \mathbf{D}_l \mathbf{A} - \mathbf{Q} \|_F^2.
$$
8. Show that Eq. 26 can be written as

\[ E_s = \|(M \otimes G)A\|_F^2, \]

where \( \otimes \) is the Kronecker product.

9. Show that Eq. 40 can be written as

\[
\begin{bmatrix}
K & P \\
P^\top & 0
\end{bmatrix}
\begin{bmatrix}
w \\
a
\end{bmatrix} =
\begin{bmatrix}
v \\
0
\end{bmatrix}.
\]

(75)

10. Given a 3\(m\)×3\(n\) matrix \(C\) with \(m < n\),

\[ C = \begin{bmatrix} I_{3m} & 0 \end{bmatrix}. \]

show that the QR decomposition of \(C^\top\) is

\[
C^\top = \begin{bmatrix}
I_{3m} & 0 \\
0 & I_{3(n-m)}
\end{bmatrix}
\begin{bmatrix}
I_{3m} \\
0
\end{bmatrix}.
\]
Appendix

There are two ways to apply Lagrange multiplier method to solve equality constrained least squares problem:

\[
\min_x \| \mathbf{Ax} - \mathbf{b} \|^2 \\
\text{subject to } \mathbf{C} \mathbf{x} = \mathbf{d}.
\]
Lagrange Multiplier Method 1

Introduce Lagrange multiplier $\lambda$ to combine the objective function and the constraint as

$$f(x) = \|Ax - b\|^2 + \lambda\|Cx - d\|^2.$$  \hspace{1cm} (76)

The minimizer of $f(x)$ is attained at

$$\frac{df(x)}{dx} = 2x^\top A^\top A - 2b^\top A + 2\lambda x^\top C^\top C - 2\lambda d^\top C = 0.$$  \hspace{1cm} (77)

Transposing both sides of Eq. 77 yields

$$\left(A^\top A + \lambda C^\top C\right)x = A^\top b + \lambda C^\top d,$$  \hspace{1cm} (78)

which has the form

$$Bx = v.$$  \hspace{1cm} (79)

So, the solution is

$$x = (B^\top B)^{-1}B^\top v.$$  \hspace{1cm} (80)
Notes:

- $\lambda$ needs to be set to an appropriate value.
- When $\lambda = 0$, equality constraint is ignored.
- When $\lambda \rightarrow \infty$, equality constraint is satisfied, but objective function is ignored.
- For $0 < \lambda < \infty$, objective function is minimized but equality constraint is not strictly satisfied.
- So, this method produces an approximate solution.
Lagrange Multiplier Method 2

Let $C_j^\top$ denote the $j$th row of $C$, and $d_j$ denote the $j$th element of $d$, for $j = 1, \ldots, m$.

Introduce Lagrange multipliers $\lambda = [\lambda_1 \cdots \lambda_m]^\top$ to combine the objective function and the constraint as

$$f(x) = \|Ax - b\|^2 + \sum_{j=1}^{m} \lambda_j (C_j^\top x - d_j). \quad (81)$$

The minimizer of $f(x)$ is attained at

$$\frac{df(x)}{dx} = 2x^\top A^\top A - 2b^\top A + \sum_{j=1}^{m} \lambda_i C_j^\top = 0. \quad (82)$$

That is,

$$2x^\top A^\top A - 2b^\top A + \lambda^\top C = 0. \quad (83)$$
Transposing both sides of Eq. 83 yields

\[ 2A^\top Ax + C^\top \lambda = 2A^\top b. \] (84)

Combine Eq. 84 and equality constraint into a single matrix equation:

\[
\begin{bmatrix}
2A^\top A & C^\top \\
C & 0
\end{bmatrix}
\begin{bmatrix}
x \\
\lambda
\end{bmatrix}
=
\begin{bmatrix}
2A^\top b \\
d
\end{bmatrix},
\] (85)

which has the form

\[ My = v. \] (86)

Then,

\[
\begin{bmatrix}
x \\
\lambda
\end{bmatrix}
=
y
= (M^\top M)^{-1}M^\top v.
\] (87)
Notes:

- \( \lambda \) are solved along with \( x \).
- When \( \lambda = 0 \), equality constraints are ignored.
- When \( \lambda_j \to \infty \) for each \( j \), equality constraints are satisfied, but objective function is ignored.
- For \( 0 < \lambda_j < \infty \), objective function is minimized but equality constraints are not strictly satisfied.
- So, this method produces an approximate solution.
References


References II


References III