Registration

600.445 Computer-Integrated Surgery
Russell H. Taylor
What needs registering?

- **Preoperative Data**
  - 2D & 3D medical images
  - Models
  - Preoperative positions

- **Intraoperative Data**
  - 2D & 3D medical images
  - Models
  - Intraoperative positioning information

- **The Patient**
A typical registration problem

Preoperative Model

Intraoperative Reality
Framework

- Definition of coordinate system relations
- Segmentation of reference features
- Definition of disparity function between features
- Optimization of disparity function
Definitions

**Overall Goal:** Given two coordinate systems, $\text{Ref}_A$ & $\text{Ref}_B$ 

and coordinates $x_A$ & $x_B$

associated with homologous features in the two coordinate systems, the general goal is to determine a transformation function $T$ that transforms one set of coordinates into the other:

$$x_A = T(x_B)$$
Definitions

• **Rigid Transformation**: Essentially, our old friends 2D & 3D coordinate transformations:

\[ T(x) = R \cdot x + p \]

The key assumption is that deformations may be neglected.

• **Elastic Transformation**: Cases where must take deformations into account. Many different flavors, depending on what is being deformed
Uses of Rigid Transformations

- Register (approximately) multiple image data sets
- Transfer coordinates from preoperative data to reality (especially in orthopaedics & neurosurgery)
- Initialize non-rigid transformations
Uses of Elastic Transformations

• Register different patients to common data base (e.g., for statistical analysis)
• Overlay atlas information onto patient data
• Study time-varying deformations
• Assist segmentation
Typical Features

- Point fiducials
- Point anatomical landmarks
- Ridge curves
- Contours
- Surfaces
- Line fiducials
Distance Functions

Given two (possibly distributed) features $F_i$ and $F_j$, need to define a distance metric distance $(F_i, F_j)$ between them. Some choices include:

- Minimum distance between points
- Maximum of minimum distances
- Area between line features
- Volume between surface features
- Area between point and line
- etc.
Disparity Functions Between Feature Sets

Let $\mathcal{F}_A = \{ \ldots F_{Ai} \ldots \}$ and $\mathcal{F}_B = \{ \ldots F_{Bi} \ldots \}$ be corresponding sets of features in $\text{Ref}_A$ and $\text{Ref}_B$, respectively. We need to define an appropriate disparity function $D(\mathcal{F}_A, \mathcal{F}_B)$ between feature sets. Some typical choices include:

$$D = \sum_i w_i [\text{distance}(F_{Ai}, T(F_{Bi}))]^2$$

$$D = \max_i \text{distance}(F_{Ai}, T(F_{Bi}))$$

$$D = \text{median distance}(F_{Ai}, T(F_{Bi}))$$

$$D = \text{Cardinality}\{i | \text{distance}(F_{Ai}, T(F_{Bi})) > \text{threshold}\}$$
Optimization

- Global vs Local
- Numerical vs Direct Solution
- Local Minima
Sampled 3D data to surface models

Outline:

- Select large number of sample points
- Determine distance function $d_S(f, \mathcal{F})$ for a point $f$ to a surface feature $\mathcal{F}$.
- Use $d_S$ to develop disparity function $D$.

Examples

- Head-in-hat algorithm [Levin et al., 1988; Pelizzari et al., 1989]
- Distance maps [e.g., Lavalle et al]
- Iterative closest point [Besl and McKay, 1992]
A typical registration problem
What the computer knows
Find homologous points & pull!
Find homologous points & pull!
Find homologous points & pull!

Iterate this until converge

Find new point pairs every iteration

Key challenge is finding point pairs efficiently.
Head in Hat Algorithm

- Levin et al, 1988; Pelizzari et al, 1989
- Originally used for Pet-to-MRI/CT registration
- Given $f_i \in \mathcal{F}_A$, and a surface model $\mathcal{F}_B$, computes a rigid transformation $T$ that minimizes

$$D = \sum_i [d_S(\mathcal{F}_B, T \cdot f_i)]^2$$

where $d_S$ is defined below, given a good initial guess for $T$.

- Optimization uses standard numerical method (steepest gradient descent [Powell]) to find six parameters (3 rotations, 3 translations) defining $T$. 
Head in Hat Algorithm

Definition of \( d_S(\mathcal{F}_B, f_i) \)

1. Compute centroid \( g_B \) of surface \( \mathcal{F}_B \).

2. Determine a point \( q_i \) that lies on the intersection of the line \( g_B - f_i \) and \( \mathcal{F}_B \).

3. Then, \( d_S(\mathcal{F}_B, f_i) = \|q_i - f_i\| \)
Head-in-hat algorithm: step 0
Head-in-hat algorithm: step1
Head-in-hat algorithm: step1
Head-in-hat algorithm: step 2
Head-in-hat algorithm: step 2
Head-in-hat algorithm: step 3
Head in Hat Algorithm

• **Strengths**
  – Moderately straightforward to implement
  – Slow step is intersecting rays with surface model
  – Works reasonably well for original purpose (registration of skin of head) if have adequate initial guess

• **Weaknesses**
  – Local minima
  – Assumptions behind use of centroid
  – Requires good initial guess and close matches during convergence
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, the goal is to find \([R, p]\) that minimizes

\[
\eta = \sum_i e_i \cdot e_i
\]

where

\[
e_i = (R \cdot a_i + p) - b_i
\]

This is tricky, because of \(R\).
Minimizing Rigid Registration Errors

Step 1: Compute
\[
\bar{\mathbf{a}} = \frac{1}{N} \sum_{i=1}^{N} \tilde{\mathbf{a}}_i \\
\bar{\mathbf{b}} = \frac{1}{N} \sum_{i=1}^{N} \tilde{\mathbf{b}}_i \\
\tilde{\mathbf{a}}_i = \mathbf{a}_i - \bar{\mathbf{a}} \\
\tilde{\mathbf{b}}_i = \mathbf{b}_i - \bar{\mathbf{b}}
\]

Step 2: Find \( \mathbf{R} \) that minimizes
\[
\sum_i (\mathbf{R} \cdot \tilde{\mathbf{a}}_i - \tilde{\mathbf{b}}_i)^2
\]

Step 3: Find \( \mathbf{p} \)
\[
\mathbf{p} = \mathbf{b} - \mathbf{R} \cdot \bar{\mathbf{a}}
\]

Step 4: Desired transformation is
\[
\mathbf{F} = Frame(\mathbf{R}, \mathbf{p})
\]
Solving for R: iteration method

Given \( \{\ldots,(\tilde{a}_i,\tilde{b}_i),\ldots\} \), want to find \( R = \arg \min \sum_i (R\tilde{a}_i - \tilde{b}_i) \)

Step 0: Make an initial guess \( R_0 \)
Step 1: Given \( R_k \), compute \( \tilde{b}_i = R_k^{-1}\tilde{b}_i \)
Step 2: Compute \( \Delta R \) that minimizes

\[
\sum_i (\Delta R\tilde{a}_i - \tilde{b}_i)^2
\]

Step 3: Set \( R_{k+1} = R_k \Delta R \)
Step 4: Iterate Steps 1-3 until residual error is sufficiently small
(or other termination condition)
Iterative method: Solving for $\Delta R$

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

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

for any vector $v$. Then, our least squares problem becomes

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

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

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

\[
H = \sum_i \begin{bmatrix}
\bar{a}_{i,x} \bar{b}_{i,x} & \bar{a}_{i,x} \bar{b}_{i,y} & \bar{a}_{i,x} \bar{b}_{i,z} \\
\bar{a}_{i,y} \bar{b}_{i,x} & \bar{a}_{i,y} \bar{b}_{i,y} & \bar{a}_{i,y} \bar{b}_{i,z} \\
\bar{a}_{i,z} \bar{b}_{i,x} & \bar{a}_{i,z} \bar{b}_{i,y} & \bar{a}_{i,z} \bar{b}_{i,z}
\end{bmatrix}
\]

Step 2: Compute the SVD of \( H = USV^t \)

Step 3: \( R = VU^t \)

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

• Failure is rare, and mostly fixable. The paper has details.
Quarternion Technique to solve for R

- 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 as a way to express the ratio of vectors. Can be thought of as

4 elements: \[ q = [q_0, q_1, q_2, q_3] \]

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

\[ q = q_0 + q_1 \hat{i} + q_2 \hat{j} + q_3 \hat{k} \]

Properties:

Linearity: \[ \lambda q_1 + \mu q_2 = [\lambda s_1 + \mu s_2, \lambda \vec{v}_1 + \mu \vec{v}_2] \]

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

Product: \[ q_1 \circ q_2 = [s_1 s_2 - \vec{v}_1 \cdot \vec{v}_2, s_1 \vec{v}_2 + s_2 \vec{v}_1 + \vec{v}_1 \times \vec{v}_2] \]

Transform vector: \[ q \circ \vec{p} = q \circ [0, \vec{p}] \circ q^* \]

Norm: \[ ||q|| = \sqrt{s^2 + \vec{v} \cdot \vec{v}} = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} \]
Digression continued: unit quaternions

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

$$\text{Rot}(\vec{n}, \theta) \iff \begin{bmatrix} \cos \frac{\theta}{2}, & \sin \frac{\theta}{2} \vec{n} \end{bmatrix}$$

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

$$\text{Rot}((\vec{n}, \theta) \vec{p}) = \left[ \cos \frac{\theta}{2}, \sin \frac{\theta}{2} \vec{n} \right] \cdot [0, \vec{p}] \cdot \left[ \cos \frac{\theta}{2}, -\sin \frac{\theta}{2} \vec{n} \right]$$
Rotation matrix from unit quaternion

\[ q = [q_0, q_1, q_2, q_3]; \quad \|q\| = 1 \]

\[
R(q) = \begin{bmatrix}
q_0^2 + q_1^2 - q_2^3 - q_3^3 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\
2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^3 - q_3^3 & 2(q_2q_3 - q_0q_1) \\
2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^3 + q_3^3
\end{bmatrix}
\]
**Unit quaternion from rotation matrix**

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

\[
a_2 = 1 - r_{xx} + r_{yy} - r_{zz}; \quad a_3 = 1 - r_{xx} - r_{yy} + r_{zz}
\]

<table>
<thead>
<tr>
<th>(a_0 = \max{a_k})</th>
<th>(a_1 = \max{a_k})</th>
<th>(a_2 = \max{a_k})</th>
<th>(a_3 = \max{a_k})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(q_0 = \frac{\sqrt{a_0}}{2})</td>
<td>(q_0 = \frac{r_{yx} - r_{zy}}{4q_1})</td>
<td>(q_0 = \frac{r_{zx} - r_{xz}}{4q_2})</td>
<td>(q_0 = \frac{r_{xy} - r_{yx}}{4q_3})</td>
</tr>
<tr>
<td>(q_1 = \frac{r_{xy} - r_{yx}}{4q_0})</td>
<td>(q_1 = \frac{\sqrt{a_1}}{2})</td>
<td>(q_1 = \frac{r_{xy} + r_{yx}}{4q_2})</td>
<td>(q_1 = \frac{r_{xz} + r_{zx}}{4q_3})</td>
</tr>
<tr>
<td>(q_2 = \frac{r_{zx} - r_{xz}}{4q_0})</td>
<td>(q_2 = \frac{r_{xy} + r_{yx}}{4q_1})</td>
<td>(q_2 = \frac{\sqrt{a_2}}{2})</td>
<td>(q_2 = \frac{r_{yz} + r_{zy}}{4q_3})</td>
</tr>
<tr>
<td>(q_3 = \frac{r_{yz} - r_{zy}}{4q_0})</td>
<td>(q_3 = \frac{r_{zx} + r_{xz}}{4q_1})</td>
<td>(q_3 = \frac{r_{yx} + r_{xy}}{4q_2})</td>
<td>(q_3 = \frac{\sqrt{a_3}}{2})</td>
</tr>
</tbody>
</table>
Quaternion method for R

Step 1: Compute

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

Step 2: Compute

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

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

Step 3: Compute eigen value decomposition of $G$

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

Step 4: The eigenvector $Q_k = [q_0, q_1, q_2, q_3]$ corresponding to the largest eigenvalue $\lambda_k$ is a unit quaternion corresponding to the rotation.
Iterative Closest Point

• Besl and McKay, 1992

• Start with an initial guess, $T_0$, for $T$.

• At iteration $k$:
  1. For each sampled point $f_i \in F_A$, find the point $v_i \in F_B$ that is closest to $T_k \cdot f_i$.
  2. Then compute $T_{k+1}$ as the transformation that minimizes

$$D_{k+1} = \sum_i \| v_i - T_{k+1} \cdot f_i \|^2$$

• Physical Analogy
Iterative Closest Point: step 0
Iterative Closest Point: step 1
Iterative Closest Point: step1
Iterative Closest Point: step 2
Iterative Closest Point: step 3
Iterative Closest Point: Discussion

- Minimization step can be fast
- Crucially requires fast finding of nearest points
- Local minima still an issue
- Data overlap still an issue
Outline of a practical ICP code

Given

1. Surface model $\mathcal{M}$ consisting of triangles $\{m_i\}$
2. Set of points $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_N\}$ known to be on $\mathcal{M}$.
3. Initial guess $\mathbf{F}_0$ for transformation $\mathbf{F}_0$ such that the points $\mathbf{F}_0\mathbf{q}_k$ lie on $\mathcal{M}$.
4. Initial threshold $\eta_0$ for match closeness
Outline of a practical ICP code

Temporary variables

\[ n \] Iteration number
\[ \mathbf{F}_n = [\mathbf{R}, \bar{\mathbf{p}}] \] Current estimate of transformation
\[ \eta_n \] Current match distance threshold
\[ C = \{\cdots, \bar{c}_k, \cdots\} \] Closest points on \( \mathcal{M} \) to \( Q \)
\[ D = \{\cdots, d_k, \cdots\} \] Distances \( d_k = \|\bar{c}_k - \mathbf{F}_n \cdot \bar{a}_k\| \)
\[ I = \{\cdots, i_k, \cdots\} \] Indices of triangles \( m_{i_k} \) corresp. to \( \bar{c}_k \)
\[ A = \{\cdots, \bar{a}_k, \cdots\} \] Subset of \( Q \) with valid matches
\[ B = \{\cdots, \bar{b}_k, \cdots\} \] Points on \( \mathcal{M} \) corresponding to \( A \)
\[ E = \{\cdots, \bar{e}_k, \cdots\} \] Residual errors \( \bar{b}_k - \mathbf{F} \cdot \bar{a}_k \)
\[ \sigma_n, (\varepsilon_{\text{max}})_n, \bar{\varepsilon}_n \] 
\[ \frac{\sum_k \bar{e}_k \cdot \bar{e}_k}{\text{NumElts}(E)} \]
\[ \max_k \sqrt{\bar{e}_k \cdot \bar{e}_k} \]
\[ \frac{\sum_k \sqrt{\bar{e}_k \cdot \bar{e}_k}}{\text{NumElts}(E)} \]
Outline of a practical ICP code

Step 0: (initialization)

Input surface model $M$ and points $Q$.
Build an appropriate data structure (e.g., octree, kD tree) $\mathcal{T}$ to facilitate finding the closest point matching search.

$n \leftarrow 0$
$I \leftarrow \{\ldots, 1, \ldots\}$
$C \leftarrow \{\ldots, \text{point on } m_1, \ldots\}$
$D \leftarrow \{\ldots, \| \mathbf{c}_k - \mathbf{F}_0 \mathbf{q}_k \|, \ldots\}$
Outline of a practical ICP code

Step 1: (matching)

\( A \leftarrow \emptyset; \quad B \leftarrow \emptyset \)

For \( k \leftarrow 1 \) step 1 to \( N \) do

begin

\( [\vec{c}_k, i_k, d_k] \leftarrow \text{FindClosestPoint}(F_n \vec{q}_k, i_k, d_k, T); \)

// Note: develop first with simple
// search. Later make more
// sophisticated, using \( T \)

if \( (d_k < \eta_n) \) then { put \( \vec{q}_k \) into \( A \); put \( \vec{c}_k \) into \( B \); };

// See also subsequent notes

end
Outline of a practical ICP code

Step 2 : (transformation update)

\[ n \leftarrow n + 1 \]
\[ \mathbf{F}_n \leftarrow \text{FindBestRigidTransformation}(A, B) \]
\[ \sigma_n \leftarrow \frac{\sqrt{\sum_k \bar{e}_k \cdot \bar{e}_k}}{\text{NumElts}(E)} ; \quad (\varepsilon_{\max})_n \leftarrow \max_k \sqrt{\bar{e}_k \cdot \bar{e}_k} ; \quad \bar{\varepsilon}_n \leftarrow \frac{\sum_k \sqrt{\bar{e}_k \cdot \bar{e}_k}}{\text{NumElts}(E)} \]

Step 3 : (adjustment)

Compute \( \eta_n \) from \( \{\eta_0, \ldots, \eta_{n-1}\} \) // see notes next page

// May also update \( \mathbf{F}_n \) from \( \{\mathbf{F}_0, \ldots, \mathbf{F}_n\} \) (see Besl & McKay)

Step 4 : (iteration)

if TerminationTest(\(\{\sigma_0, \ldots, \sigma_n\}, \{(\varepsilon_{\max})_0, \ldots, (\varepsilon_{\max})_n\}, \{\bar{\varepsilon}_0, \ldots, \bar{\varepsilon}_n\}\))
then stop. Otherwise, go back to step 1 // see notes
Outline of practical ICP code

Threshold $\eta_n$ update

The threshold $\eta_n$ can be used to restrict the influence of clearly wrong matches on the computation of $F_n$. Generally, it should start at a fairly large value and then decrease after a few iterations. One not unreasonable value might be something like $3(\varepsilon_{\text{max}})_n$. If the number of valid matches begins to fall significantly, one can increase it adaptively.

Also, if the mesh is incomplete, it may be advantageous to exclude any matches with triangles at the edge of the mesh.
Outline of practical ICP code

Termination test

There are no hard and fast rules for deciding when to terminate the procedure. One criterion might be to stop when $\sigma_n, \bar{\epsilon}_n$ and/or $\left( \epsilon_{\max} \right)_n$ are less than desired thresholds and $1 \leq \frac{\bar{\epsilon}_n}{\bar{\epsilon}_{n-1}} \leq \gamma$ for some value $\gamma$ (e.g., $\gamma \equiv .95$) for several iterations.
**Homework & Programming Assignments**

- **HW#3 (2002)** covers analysis of a registration method similar to ICP
- **PA#2 (2002)** requires implementation of point cloud to point cloud registration
- **PA#3 (2002)** will require implementation of closest point pairing and performing one step of ICP
- **PA#4 (2003)** will require putting it all together, together with some possible refinements
Distance Maps

- Many authors, e.g., Lavalle, Brunie, Malandain, Margin

- Basic idea is to use different distance metric:

\[ d_S(\mathcal{F}, f) = \min_{p \in \mathcal{F}} \| p - f \| \]

- But the problem is how to compute this quickly
Distance Maps (Continued)

- Approach is to precompute \( d_S(\mathcal{F}, v_j) \) for a lattice of points \( v_j \).

- Then, to compute \( d_S(\mathcal{F}, f_i) \):
  
  1. Determine the set \( \mathcal{V} \) of lattice points surrounding \( f_i \).
  2. Look up the distances \( \{d_j = d_S(\mathcal{F}, v_j)\} \) for \( v_j \in \mathcal{V} \).
  3. Estimate \( d_S \) from the \( d_j \), e.g., by trilinear interpolation

- Various techniques to do the optimization
Distance Maps: step 0
Distance Maps: step 1
Distance Maps: step 1
Distance Maps: step 2
Distance Maps: step 3
Interpolating Distance Maps
(2D Example)

\[ d_S = \mu \left[ \lambda d_{22} + (1 - \lambda) d_{21} \right] + (1 - \mu) \left[ \lambda d_{12} + (1 - \lambda) d_{11} \right] \]

\[ \nabla d_S = \begin{bmatrix}
\mu(d_{22} - d_{21}) + (1 - \mu)(d_{12} - d_{11}) \\
\lambda(d_{22} - d_{12} + (1 - \lambda)(d_{21} - d_{12})
\end{bmatrix} \]
Distance Maps: Iteration Step

1. Determine cell $\mathcal{V}_i$ for each $\mathbf{p}_i = \mathbf{T} \cdot \mathbf{f}_i$. Let $\lambda_i$ be the corresponding interpolation parameters for $\mathbf{p}_i$ within cell.

2. Determine small motion $\Delta \mathbf{T}$ that minimizes

\[
\sum_i [(\Delta \mathbf{T} \mathbf{p}_i - \mathbf{p}_i) \cdot \nabla d_S(\lambda_i, \mathcal{V}_i)]
\]

or

\[
\sum_i -[(\Delta \mathbf{T} \mathbf{p}_i - \mathbf{p}_i) \cdot \nabla d_S(\lambda_i, \mathcal{V}_i)]
\]

3. Update $\mathbf{T} \leftarrow \Delta \mathbf{T} \cdot \mathbf{T}$
2D-to-3D using Octree Splines

- Lavallee, et. al
- Registers 2D images (e.g., x-ray) to 3D models
- Select 2D sample points (defines 3D rays) on contours
- Distance map method
  - Store distance of line to surface
  - Use Octree to store distance map
- Least squares minimization
- When converge, 3D surface is tangent to irregularly shaped cone defined by rays
2D-to-3D

Source: Lavallee, CIS book
Octree Spline Distance Maps

• Select large number of points, $s_j$ on the surface.
• Construct standard octree subdivision of space to encompass the $s_j$
• Subdivide unoccupied cells near surface to be sure that adjacent cells differ in size from occupied cells by at most a factor of two.
• For each corner $c$ of each octree cell, compute the euclidean distance $d_e(c)$ to the closest point on the surface.
  – **Note**: Can do this with efficient tree walk of parents & neighbors of each cell
Octrees

Source: Lavallee, CIS book
Octree-spline

Source: Lavallee, CIS book
Octree Spline Distance Maps (con’d)

- Compute signed distances $d'(c)$ for each $c$
  - Note: assumes know something about the surface normals or can otherwise tell inside from outside
- Tweak the $d'(c)$ to assure continuity of interpolations (a rather technical correction)
- Compute a lower bound on interpolation function value. I.e., for each node $N$ of the octree, find a function $B_N(q)$ such that $B_N(q) \leq d'(q)$ for all $q$ in $N$.
  - This is done by a linear programming method defining $B_N(q) = a \cdot q + b$, and then finding $a$ and $b$ such that $a \cdot q + b \leq d'(q)$ for all $q$ in $B_N(q)$. 
Line-to-surface distances

- Represent lines by

\[ p_i(\eta, \lambda) = T_{reg}(\eta)[q_i + \lambda v_i] \]

where \( q \) is a point [on the contour], \( v \) is a direction, and \( \lambda \) is a free scalar variable. \( T_{reg}(\eta) \), is a transformation we are trying to find.

- \( d_i(p_i(\eta, \lambda) \) can have local minima along \( \lambda \), even if \( \eta \) is fixed.

**Caution** Lavallee uses \( p \) for parameters of \( T \) I find this confusing, so I have chosen to use \( \eta \).

- Global search (e.g., simulated annealing)
- Exhaustive search of all cells line intersects
- Tree-based search
Line-to-surface distances

Let $u(p)$ be the normalized coordinates of a point $p$ within a cell with corners $\{p_{000}, p_{001}, \ldots, p_{111}\}$:

$$u(p) = \frac{p - p_{000}}{p_{111} - p_{000}}$$

where $p_{000}$ and $p_{111}$ are opposite corners of the cell. Let $a$ and $b$ be points where a line enters and leaves cell. Then, the equation of the line has the form

$$p(\lambda) = \lambda b + (1 - \lambda)a$$

$$= interpolate(\lambda, a, b)$$

Problem is to find

$$\min_\lambda \tilde{d}(p(\lambda))$$

where

$$d(p(\lambda)) = interpolate(u(p(\lambda)), d_{000}, \ldots, d_{111})$$

$$d_{ijk} = \text{distance at } ijk'\text{th corner}$$

As a practical matter, can just take $\min(\tilde{d}(a), \tilde{d}(b))$. 

Note: $\tilde{d}$ is just $d'$ of the earlier slides.
Energy Minimization

- Use standard technique (Lavenberg-Marquardt).

- Energy is

\[ E(\eta) = \sum_i \frac{1}{\sigma_i^2} [e_i(\eta)]^2 \]

I.e.

\[ E(\eta) = \sum_i \frac{1}{\sigma_i^2} \left[ \min_{\lambda} \tilde{d}(T(\eta) \cdot (q_i + \lambda v_i)) \right]^2 \]

- Gradient is

\[ \frac{\partial e_i}{\partial \eta_i} = \left[ \nabla \tilde{d}(T(\eta) \cdot (q_i + \lambda_{\text{min}} v_i)) \right] \cdot \left[ \frac{\partial T(\eta)}{\partial \eta_j} (q_i + \lambda_{\text{min}} v_i) \right] \]
3D-to-3D with Octree Map

- Easy to adapt the mechanism of lines to points
- Use standard (unsigned) distance – saves step in computing map
- Energy function, given points $q_i$, is

$$E(\eta) = \sum_i \frac{1}{\sigma_i^2} \left[ e_i(\eta) \right]^2 = \sum_i \frac{1}{\sigma_i^2} \left[ \tilde{d}(T(\eta)q_i) \right]^2$$

- Gradient is

$$\frac{\partial e_i}{\partial \eta_i} = \left[ \nabla \tilde{d}(T(\eta)q_i) \right] \cdot \left[ \frac{\partial T(\eta)}{\partial \eta_j} q_i \right]$$
Figure 7.9: Convergence of algorithm for surface S, observed from three projection viewpoints. The external contours of the projected surfaces aid in verifying the final outcome.

(a) Initial configuration. (b) After two iterations. (c) After six iterations.
Sample Set Analysis

• **Question:** How good is a particular set of 3D sample points for the purpose of registration to a 3D surface?

• Long line of authors have looked at this question
• Next few slides are based on the work of David Simon, et al (1995)
Sample Set Analysis: Distance Estimates

Let

\[ F(x) = 0 \]

be the implicit equation of a surface, then one good estimate of the distance of a point \( x \) to the surface is

\[ D(x) = \frac{F(x)}{\|\nabla F(x)\|} \]
Sample set analysis: sensitivity

Let $x_s$ be a point on the surface, and let $T(\eta)$ represent a small perturbation with parameters $\eta$ with respect to the surface of point $x_s$:

$$x'_s = T(\eta)x_s$$

Then we define $V(x_s)$ to be

$$V(x_s) = \frac{\partial D(T(\eta)x_s)}{\partial \eta} = \begin{bmatrix} n_s \\ x_s \times n_s \end{bmatrix}$$

where $n_s$ is the unit normal to the surface at $x_s$. So,

$$D(T(\eta)x_s)) \simeq V^T(x_s)\eta$$

Squaring this gives

$$D^2(T(\eta)x_s)) \simeq \eta_T V(x_s)V^T(x_s)\eta$$

$$= \eta^T M(x_s)\eta$$

Note that $M$ is $6 \times 6$ positive, semi-definite, symmetric matrix.
Sample set analysis: sensitivity

For a region $\mathcal{R}$, define

$$E_R(\bar{\eta}) = \bar{\eta}^T \left[ \sum_{x_s \in \mathcal{R}} M(x_s) \right] \bar{\eta}$$

$$= \bar{\eta}^T \Psi \bar{\eta}$$

$$= \bar{\eta}^T Q \Lambda Q^T \bar{\eta}$$

$$= \sum_{1 \leq i \leq 6} \lambda_i \left( \bar{\eta}^T \cdot q_i \right)^2$$

- Note that the eigenvectors $q_i$ correspond to small differential transformations $T(q_i)$, and can sort eigenvalues so that

$$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_6$$

- Note that eigenvector $q_1$ corresponds to direction of greatest constraint.

- Similarly, can also think of $q_6$ as the least constrained direction.
Sample Set Analysis: Goodness Measures

- Magnitude of smallest eigenvalue (Simon)
- (Kim and Khosla)
  \[ \frac{\sqrt[6]{\lambda_1 \cdots \lambda_6}}{\lambda_1 + \ldots + \lambda_6} \]
- Nahvi
  \[ \frac{\lambda_6^2}{\lambda_1} \]
Sample Set Selection

• One blind search method (similar to Simon, 1995) is:
  – Randomly select sample points on surface
  – (prune for reachability)
  – evaluate goodness of sample set using some criterion
  – repeat many times and choose the best one found
Sample Set Selection

• Refinement of blind search (hill climbing):
  – Randomly select sample points on surface
  – (prune for reachability)
  – evaluate goodness of sample set using some criterion
  – replace a point from sample set with a randomly selected point
  – evaluate goodness
  – if better, keep it
  – else revert to original point and try again

• Variations include simulated annealing, “genetic” algorithms
Sample Set Selection: Another Alternative

- Select large number of random points $x_s$
- Prune for reachability
- For each point, compute constraint direction $V_s = V(x_s)$. To a first approximation, a measurement at $x_s$ with accuracy $\epsilon_s$ constrains $\bar{\eta}$ by

$$|V_s \bar{\eta}| \leq \epsilon_s$$

- Now select subset of the $x_s$ that minimizes, e.g.,

$$\min_{\delta_s} \max \eta^T S \eta$$

subject to

$$\delta_s \in \{0, 1\}$$

$$|\delta_s V_s \bar{\eta}| \leq \epsilon_s$$

$$\sum_s \delta_s \leq \text{subsetsize}$$

There are various ways to do this.
Sample Set Selection: Another Alternative (con’d)

- One can also minimize other forms, e.g.,

\[
\min \max_i |\sigma_i \eta_i|
\]

subject to similar constraints

- An alternative is to minimize the number of sample points required to ensure that some constraints on \( \eta \) are guaranteed to be met. E.g.,

\[
\min \sum \delta_s
\]

such that

\[
\delta_s \in \{0, 1\}
\]

\[
\xi \leq \xi_{limit}
\]

where

\[
\xi = \max \eta^T S \eta
\]

or some other form subject to

\[
|\delta_s V_s \eta| \leq \epsilon_s
\]
Deformable Atlas-based Registration

• Material that follows is derived from Jianhua Yao’s Ph.D. thesis work:

• A number of other authors, including
  – Cootes et al. 1999
  – Feldmar and Ayache 1994
  – Ferrant et al. 1999
  – Fleute and Lavallee 1999
  – Lowe 1991
  – Maurer et al. 1996
  – Shen and Davatzikos 2000
Deformable Registration Between Density Atlas and Patient CT

- Goal: Register and Deform the statistical density atlas to match patient anatomy
- Significance:
  - Building patient specific model with same topology (mesh structure) as the atlas
  - Automatic segmentation
  - Accumulatively building models for training set
  - Pathological diagnosis

Jianhua Yao
Deformable Registration Scheme

• Affine Transformation
  – Translation $T=(t_x, t_y, t_z)$
  – Rotation $R=(r_x, r_y, r_z)$
  – Scale $S=(s_x, s_y, s_z)$

• Global Deformation
  – Statistical deformation mode ($M_j$)

• Local Deformation
  – Adjustment of every vertex
Optimization Algorithm

- Direction Set (Powell’s) methods in multi-dimensions
  - Search the parameter space to minimize the cost functions
  - Advantage
    - Don’t need to compute derivative of cost functions
    - Much fewer evaluations than downhill simplex methods
Energy Function

- To measure the density and shape difference between model and image

\[
E(mdl, img) = w_s E^{(s)}(mdl, img) + w_d E^{(d)}(mdl, img)
\]

\[
E^{(s)}(mdl, img) = \sum_{i=1}^{N(v)} (\tilde{g}^{(mdl)}(v_i) \cdot \tilde{g}^{(img)}(v_i))
\]

\[
E^{(d)}(mdl, img) = \sum_{i=1}^{N(t)} \left( \int_{\mu} \left( \frac{d^{(mdl)}(t_i, \mu) - d^{(img)}(t_i, \mu)}{d^{(mdl)}(t_i, \mu)} \right)^2 \right)
\]
Local Deformation

• Motivation: Statistical deformation can’t capture all the variability due to the limited number of models in the training set
• Locally adjust the location of vertices to match the boundary of the bone and the interior density property
• Use multiple-layer flexible mesh template matching to find the correspondence between model vertices and image voxels
Multiple-layer Flexible Mesh Template

- Each vertex on the model defines a mesh template
- Template is in the form

\[(0, \text{Sphere } (v_1^{(1)} - v^{(0)}, r_1), \]
\[\text{Sphere } (v_2^{(1)} - v^{(0)}, r_1), \cdots, \]
\[\text{Sphere } (v_1^{(2)} - v^{(0)}, r_2), \]
\[\text{Sphere } (v_1^{(2)} - v^{(0)}, r_2), \cdots)\]
Results (Affine Transformation)

Jianhua Yao
Results (Global Deformation)

Initial

Intermediate

Final

Jianhua Yao
Results (Local Deformation)

Initial  Intermediate  Final

Jianhua Yao
Results (Deformable Registration)

Deformable Atlas/CT Registration

Energy Function

Iteration
Toolkits

Jianhua Yao
Deformable registration between density atlas and a set of 2D X-Rays

• Goal: Register and Deform the statistical density atlas to match intraoperative x-rays

• Significance:
  – Build virtual patient specific CT without real patient CT
  – Register pre-operative models and intra-operative images
  – Map predefined surgical procedure and anatomical landmarks into intra-operative images
2D/3D Registration Scheme

- Solve a Powell optimization problem to minimize the cost function between DRRs of model and x-rays

\[
\arg \min_{\Theta} \sum_{i=1}^{n} E(DRR_i(\Theta(mdl)), img_i)
\]

- Multiple resolutions and multiple step sizes
- \(E\) is the cost function
- \(\Theta\) is the transformation of model
  - Affine Transformation
    - Translation \(T=(t_x, t_y, t_z)\)
    - Rotation \(R=(r_x, r_y, r_z)\)
    - Scale \(S=(s_x, s_y, s_z)\)
  - Global Deformation
    - Statistical deformation mode \(\langle M_i \rangle\)
2D/3D Registration Experiment 1

Instance of Atlas

DRRs

Average Atlas

Deformable 2D/3D registration

Compare

Results

Patient specific model

Jianhua Yao
2D/3D Registration Experiment 2

CT data set

DRRs

Average Atlas

Deformable 2D/3D registration

Compare

Results

Patient specific model

Jianhua Yao
2D/3D Registration Experiment 3

CT data set

Average Atlas

Deformable 3D/3D registration

Patient specific model

Results

Deformable 2D/3D registration

Patient specific model

Compare

Jianhua Yao
2D/3D Registration Experiment 4

CT data set \rightarrow X Rays \rightarrow Deformable 2D/3D registration \rightarrow Compare \rightarrow Results

Average Atlas \rightarrow Patient specific model

Jianhua Yao
Results on Simulated Data

2D/3D Registration

Cost Function

Iteration

Jianhua Yao
Visual Results

Initial | Intermediate | Final

Jianhua Yao
Future work

• Enlarge atlas
  – More patients
  – More parts of body (knee, spine, …)

• Registration & segmentation:
  – Atlas-driven reconstruction
  – Registration to incomplete data
  – Fractures & abnormal anatomy

• Enriched information
  – Biomechanics and biomedical
  – Surgical information

Jianhua Yao & Russ Taylor