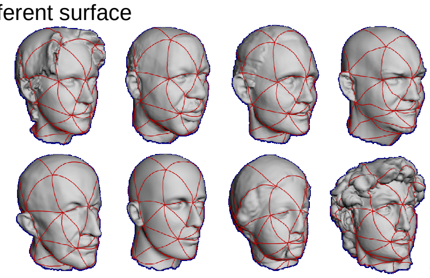
Surface Registration

Thomas Funkhouser
COS 526, Fall 2006

Goal

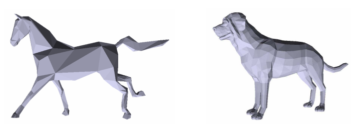
- Establish 1 to 1 mapping between points on one 3D surface and corresponding points on a different surface



Praun

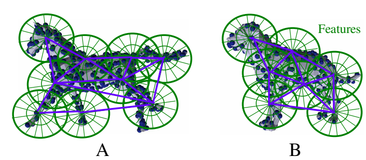
Motivation - Matching

- Determine geometric similarity of surfaces



Motivation - Matching

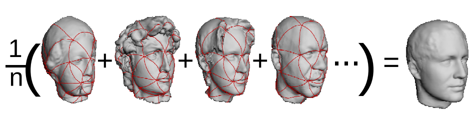
- Determine geometric similarity of surfaces



$$D(A,B) = \sum_{\text{Correspondences}} \Delta\text{FeatureShape} + \sum_{\substack{\text{Correspondence}\\ \text{Pairs}}} \Delta\text{SpatialConsistency}$$

Motivation – Common parameterization

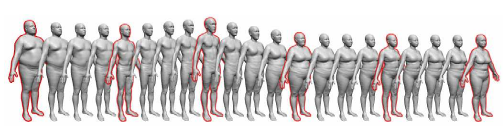
- Registration provides consistent parameterization
  - Allows for basic operations like matching, mean, signal processing, etc.



Praun

Motivation - Morphing

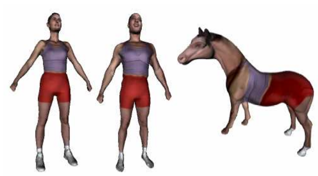
- Smoothly transition from one surface to another
  - When registered, simply use linear combinations of vertices



Allen03

Motivation – Attribute Transfer

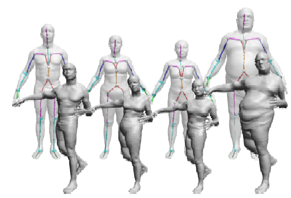
- Copy attributes from one surface to another
  - Texture transfer (below)
  - Deformation weight transfer
  - Segmentation transfer



Praun

Motivation – Attribute Transfer

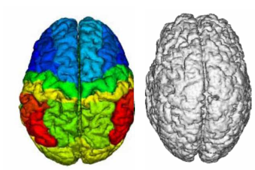
- Copy attributes from one surface to another
  - Texture transfer
  - Deformation weight transfer (below)
  - Segmentation transfer



Allen03

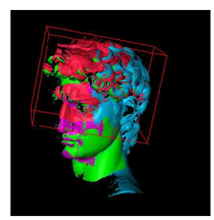
Motivation – Attribute Transfer

- Copy attributes from one surface to another
  - Texture transfer
  - Deformation weight transfer
  - Segmentation transfer (below)



Motivation –Scanning

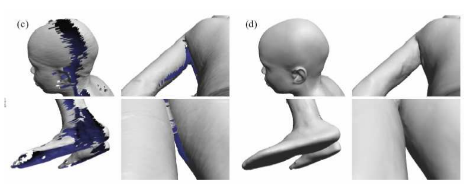
- Combine multiple scans to form complete surface
  - Must align scans from different views



Rusinkiewicz

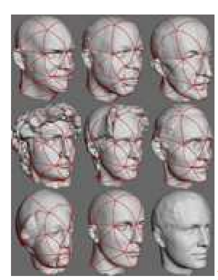
Motivation – Hole Filling

- Use surface of one model to fill holes of another
  - e.g., to fix surfaces captured with range scanners



Registration Goal

- Find minimal dissimilarity measure between surfaces over class of possible transformations



Praun

Registration Methods

- Underlying issues:
  - Transformation type
  - Surface representation
  - Dissimilarity measure
  - Algorithmic strategy

Registration Methods - Part 1

- Transformation Type
  - Rigid: mutual distances of points within a model are conserved during transformation

$$x_B = R_{AB} x_A + t_{AB}$$

    R is a rotation matrix and t is a translation vector

  - Non-rigid: account for surface deformations in the transformation
    - e.g., Affine transformation
    - e.g., Thin plate spline

Registration Methods - Part 2

- Surface Representation
  - Surface description
    - Points, mesh, splines, etc.
  - Surface features
    - Curvature extrema, saddle points, ridges, etc.
  - Shape descriptors
    - Harmonic shape descriptors, spin images

Registration Methods - Part 3

- Dissimilarity measure
  - Distance
    - Distances between corresponding points after alignment
  - Deformation
    - Amount of deformation implied by alignment

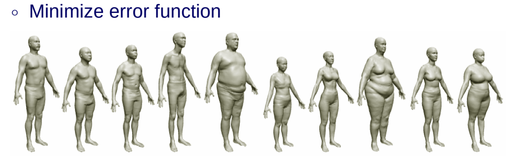
Registration Methods - Part 4

- Algorithmic strategy
  - Optimization
    - Iterative methods
    - Simulated annealing
  - Voting
    - Pose clustering
    - Geometric Hashing
    - Generalized Hough Transform

Example – Registering Human Bodies

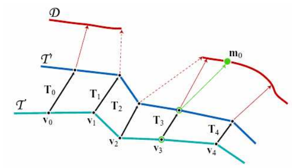
Allen et al.
The Space of Human Body Shapes
Siggraph 2003

- Algorithm Input
  - Set of human range images
  - Set of colored feature markers
- Algorithm Goal
  - Develop correspondence from template to target
  - Compute affine transform for each vertex
  - Minimize error function



Optimization Variables

- Algorithm viewed as optimization problem
  - Given an initial template surface with vertices v_i
  - Corresponding affine transformation matrices T_i
  - Current state is $T_i\ v_i$ for all i (see diagram)
  - Find values of T_i to minimize objective function
  - Attempts to find a “good fit” (blue) of template (cyan) to target (red)



Objective Function

- Objective Function has three weighted terms
  - Data error
  - Smoothness error
  - Marker error

$$E = \alpha E_d + \beta E_s + \gamma E_m$$

- Will use different weights in each phase of process
  - Multistep / Multi-resolution fitting process

Objective Function – Marker Error

- Measures distance between pre-labeled markers
  - Correspondences set up beforehand

$$E_m = \sum_{i=1}^{m} ||\mathbf{T}_{\kappa_i}\mathbf{v}_{\kappa_i} - \mathbf{m}_i||^2$$

Objective Function – Data Error

- Data error term requires current match to be close to target
  - Uses distance from each transformed vertex to the target surface
  - Weighted by confidence measure (from scanning)
  - Hole regions have weight = 0
  - Sums total error

$$E_d = \sum_{i=1}^{n} w_i \, \mathrm{dist}^2(\mathbf{T}_i\mathbf{v}_i, \mathcal{D})$$

- Distance function
  - Uses transformed template vertex
  - Takes minimum distance to “compatible” vertices in target

Objective Function – Smoothness

- Measures smoothness of deformation applied to template
  - E_s measures change in T_I between adjacent vertices
  - Encourages similarly-shaped features to be mapped to each other

$$E_s = \sum_{\{i,j|\{\mathbf{v}_i,\mathbf{v}_j\}\in edges(\mathcal{T})\}} ||\mathbf{T}_i - \mathbf{T}_j||_F^2$$

  - Uses Frobenius norm (vector L2 norm)

Algorithm Procedure

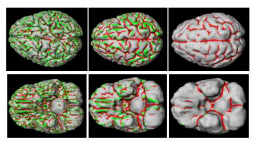
- Minimize error function using L-BFGS-B algorithm
  - Quasi-Newton method with limited memory usage

$$E = \alpha E_d + \beta E_s + \gamma E_m$$

- Make four passes over data (2 low res, 2 high res)
  - Fit markers (low res, $\alpha = 0$, $\beta = 1$, $\gamma = 10$)
  - Refit using data term (low res, $\alpha = 1$, $\beta = 1$, $\gamma = 10$)
  - Repeat in high resolution (hi res, $\alpha = 1$, $\beta = 1$, $\gamma = 10$)
  - Refit using predominantly data term (hi res, $\alpha = 10$, $\beta = 1$, $\gamma = 1$)

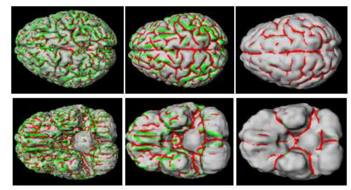
Example – Aligning Brains

- Algorithm Input
  - Set of human brain surfaces
  - Prelabeled reference brain mesh (low resolution)
- Algorithm Goal
  - Correspondence from template to target
  - Identify particular features in the brain (gyri, sulci)
  - Minimize error function



Transformation types

- In brains, we see homothetic deformation (local uniform stretch) when aligning features



Objective Function

- Minimizes error between vertex and feature point

$$O_{ij} = d_{ij} \cdot n_{ij} \cdot f_{ij}$$

  - Euclidean distance measure

$$d_{ij} = 1 + \sqrt{[x_i - x_j]^2 + [y_i - y_j]^2 + [z_i - z_j]^2}$$

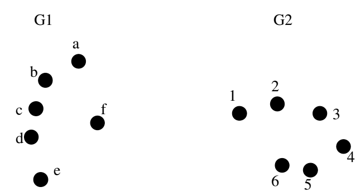
  - Surface Normal Match

$$n_{ij} = 2 - \vec{n}_i \cdot \vec{n}_j$$

  - Feature Match

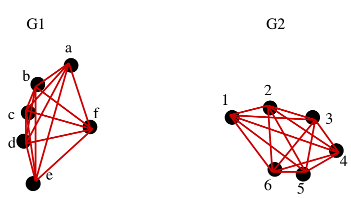
$$f_{ij} = \begin{cases} 1.0, & \text{if } (t_i, t_j) = \begin{cases} \text{(sulcus, sulcus)} \\ \text{(gyrus, gyrus)} \\ \text{(no feature, no feature)} \end{cases} \\ 2.8, & \text{if } (t_i, t_j) = \begin{cases} \text{(sulcus, no feature)} \\ \text{(gyrus, no feature)} \end{cases} \\ 3.0, & \text{if } (t_i, t_j) = \text{(sulcus, gyrus)} \end{cases}$$

Example – Aligning Point Sets



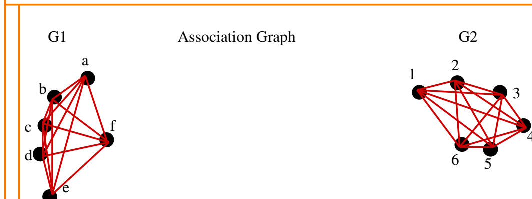
Consider rigid transformations

Association Graph



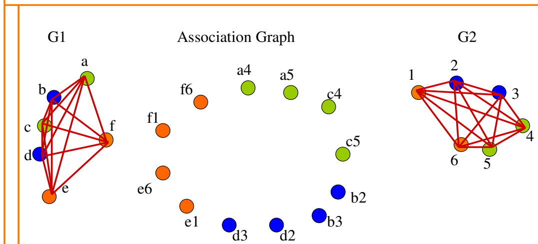
Represent both points sets as complete graphs (G1 and G2).
(edges connect all pairs of vertices within each point set)

Association Graph



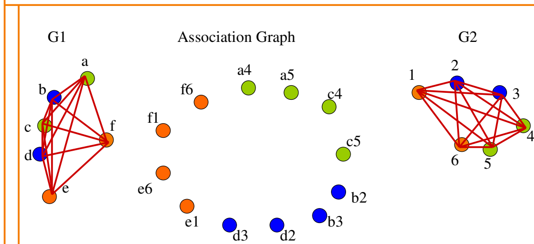
Create vertices in the association graph for all compatible pairs of vertices in the original graphs.
This can lead to a large number of vertices.

Association Graph



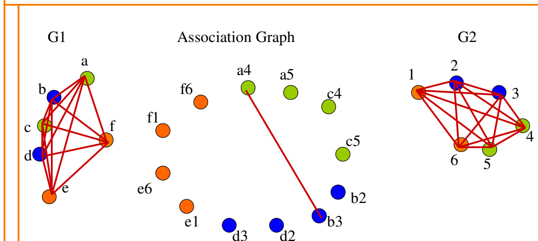
Create vertices in the association graph for all compatible pairs of vertices in the original graphs. Compatibility could refer to chemical properties.

Association Graph



Create edges between (uv) and (wx) if the edges between (u) and (w) as well as between (v) and (x) match.

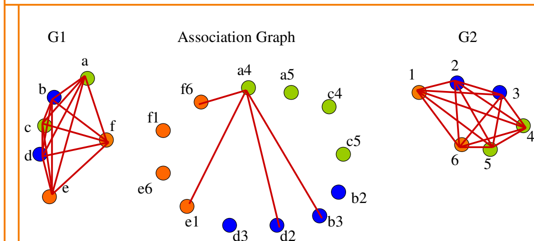
Association Graph



Create edges between (uv) and (wx) if the edges between (u) and (w) as well as between (v) and (x) match.

For this example, edge length is the only consideration

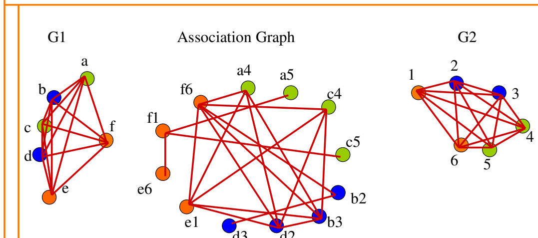
Association Graph



Create edges between (uv) and (wx) if the edges between (u) and (w) as well as between (v) and (x) match.

For this example, edge length is the only consideration

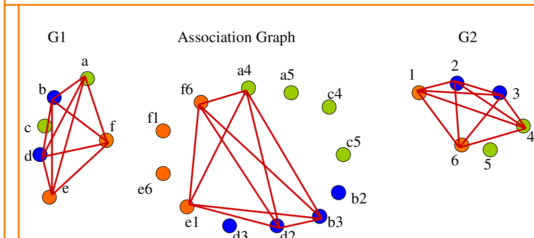
Association Graph



Create edges between (uv) and (wx) if the edges between (u) and (w) as well as between (v) and (x) match.

For this example, edge length is the only consideration

Association Graph



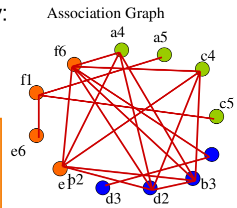
The the largest set of corresponding nodes in the same configuration is the maximal clique in the association graph (i.e., the largest subset of the association graph for which all nodes are connected to one another).

Association Graph

- Computational complexity:
  - $O(2^n)$ for n points
  - NP-complete

```
Find the Maximal Clique{
    return Cliques(empty, all nodes)
}

Cliques(X, Y){
    if (no node in Y-X is connected to all of X){
        return X;
    }else{
        y = node in Y connected to all of X;
        return Largest(Cliques(X union y, Y},
                       Cliques{X, Y-y});
    }
}
```



[Schmitt02, Brown82]

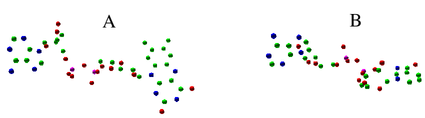
Iterative Closest Points (ICP)

- Assume closest points correspond
  - Avoid finding one-to-one correspondences
- Rigid body transformations
- Greedy optimization procedure
  - Start with rough guess for alignment
  - Iteratively refine transform

[Besl92]

Iterative Closest Point

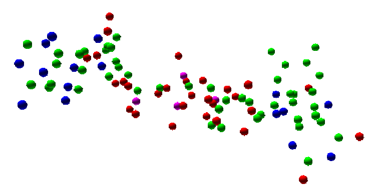
- Given two point sets



[Besl92]

Iterative Closest Point

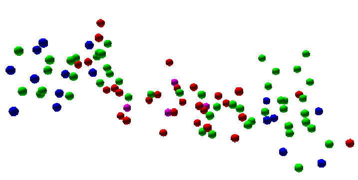
- Given two point sets and an initial guess for the transformation that aligns them



[Besl92]

Iterative Closest Point

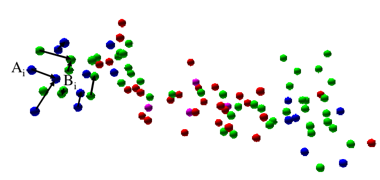
- Assume closest points correspond



[Besl92]

Iterative Closest Point

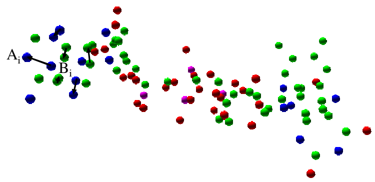
- Assume closest points correspond: A→B



[Besl92]

Iterative Closest Point

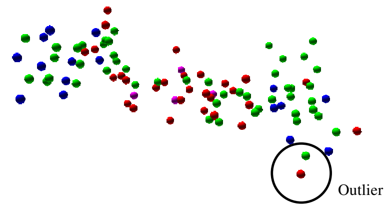
- Assume closest points correspond: A→B and B→A



[Besl92]

Iterative Closest Point

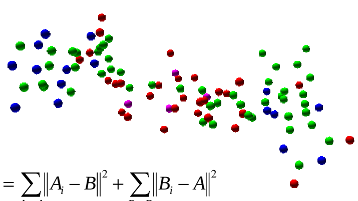
- Rejecting outliers



[Besl92]

Iterative Closest Point

- Find the transformation that optimally aligns proposed correspondences (superposition)



$$d(A,B) = \sum_{A_i \in A} \|A_i - B\|^2 + \sum_{B_i \in B} \|B_i - A\|^2$$

[Besl92]

Iterative Closest Point

- Iterate until convergence
  1. Select source points (from one or both surfaces)
  2. Match to points in the other molecule
  3. Weight the correspondences
  4. Reject outlier point pairs
  5. Compute an error metric for the current transform
  6. Minimize the error metric w.r.t. transformation

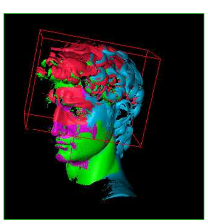
Computational complexity

- O(k * nlogn) for n points per set and k iterations
  - § k iterations * O(n) points * O(logn) to find closest point

Slide courtesy of Szymon Rusinkiewicz

ICP – Aligning Surfaces (Scans)

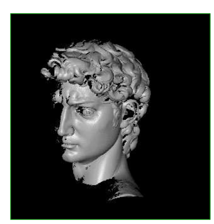
- Start with manual initial alignment



Slide courtesy of Szymon Rusinkiewicz

ICP - Aligning Surfaces (Scans)

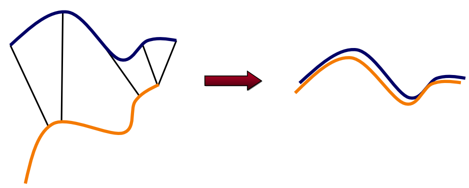
- Improve alignment using ICP



Slide courtesy of Szymon Rusinkiewicz

ICP - Aligning Surfaces (Scans)

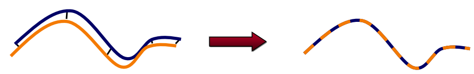
- Assume closest points correspond, compute the best transform…



Slide courtesy of Szymon Rusinkiewicz

ICP - Aligning Surfaces (Scans)

- … and iterate to find alignment
- Converges to some local minimum
- Correct if starting position "close enough"



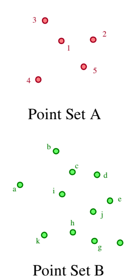
Slide courtesy of Szymon Rusinkiewicz

Pose Clustering

- General method
  - Enumerate possible transformations
  - Vote for best one
- Methods
  - Pose clustering
  - Geometric Hashing
  - Generalized Hough Transform

Pose Clustering

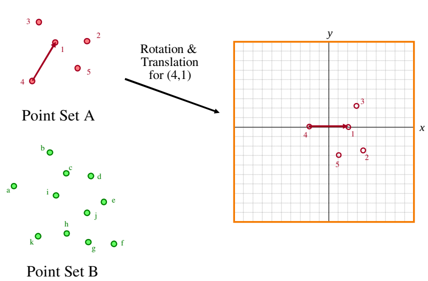
- Discretize transformations and scoring



[Wolfson97]

Pose Clustering

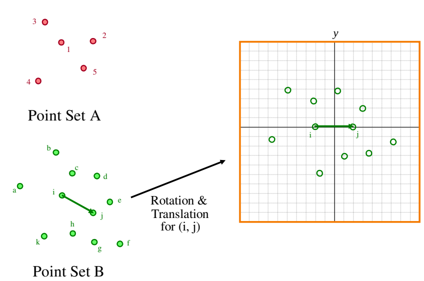
- Discretize transformations and scoring



[Wolfson97]

Pose Clustering

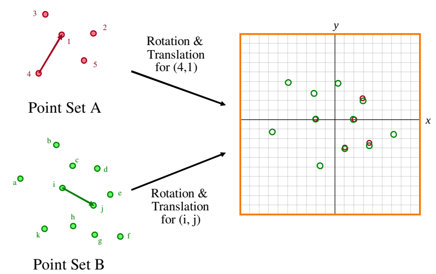
- Discretize transformations and scoring



[Wolfson97]

Pose Clustering

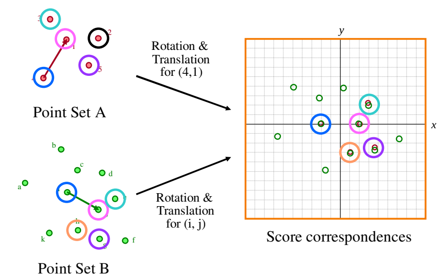
- Discretize transformations and scoring



[Wolfson97]

Pose Clustering

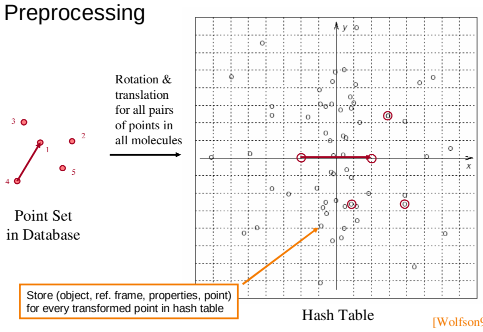
- Discretize transformations and scoring



[Wolfson97]

Geometric Hashing

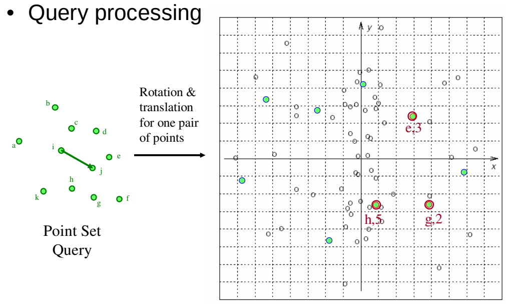
- Preprocessing



[Wolfson97]

Geometric Hashing

- Query processing



[Wolfson97]

Geometric Hashing

- Preprocessing
  - For each triple of points
  - Compute reference frame
  - For each point
    - Transform point into reference frame
    - Hash (molecule, ref. frame, properties, point)
- Query processing
  - Choose any triple of points
  - Compute reference frame
  - For each point
    - Transform point into reference frame
    - For each entry in hash bin for transformed point
    - Vote for (object, ref. frame)

Geometric Hashing

- Preprocessing complexity
  - $O(n^4)$ for n points per binding site
    - $O(n^3)$ possible triples * $O(n)$ transformations per triple
- Query complexity
  - $O(m)$ * binsize for m points in query binding site
    - 1 triple * $O(m)$ transformations per triple * binsize hash processing per transformation

[Wolfson97]

Summary

- Different methods for different …
    - Transformation types
    - Surface representations
    - Dissimilarity measures