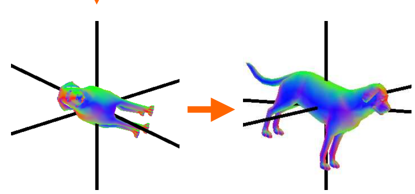
Rigid-Body Registration

COS 597D
Mike Burns
Paul Calamia
September 30, 2003

Registration

- What is registration?
  - Finding a one-to-one mapping between two or more coordinate systems such that corresponding features of models in the different systems are mapped to each other
  - Using the mapping to align a model(s)
    - Pair-wise model alignment
    - Transformation to a canonical pose/coordinate system

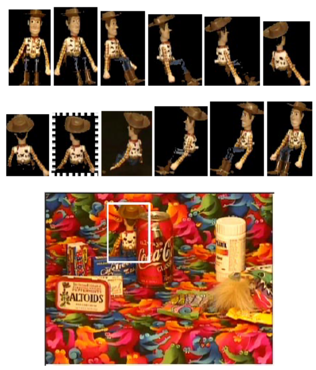


Audette 2000

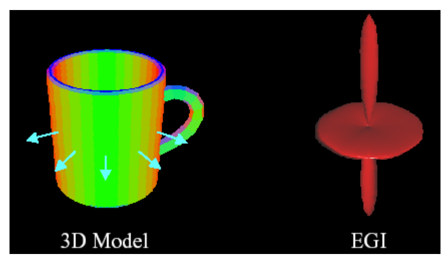
M. Kazhdan

Registration

- What is the resulting alignment/pose used for?
  - Object recognition in scenes
  - Stitching together parts of a model captured from different views
  - Alignment for pose-dependent shape descriptors



OR



Chang and Krumm, 1999

Funkhouser, COS 597D Class Notes

Lecture Overview

- Sub-problems within registration (from Audette00)
- Placing models in a canonical pose or coordinate system
- Methods for pair-wise model registration
  - ICP
  - Generalized Hough Transform
  - Geometric Hashing

Lecture Overview

- Sub-problems within registration (from Audette00)
- Placing models in a canonical pose or coordinate system
- Methods for pair-wise model registration
  - ICP
  - Generalized Hough Transform
  - Geometric Hashing

General Registration

- Partition the process into three underlying issues:
  - Transformation(s)
  - Surface Information/Representation and Similarity Criterion
  - Matching and optimization

Audette 2000

Registration Part 1

- Choice of Transformation
  - 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
    - Affine transformation, e.g.
    - Global polynomial function (low order polynomial to map one surface to another)
    - Chris will talk about these on Thursday

Registration Part 2

- Surface Representation and Similarity Criterion
  - Local surface information
    - Points or specific features, e.g. curvature extrema, saddle points, ridges, etc.
  - Global surface information
    - Spin maps, e.g.
- Choice of surface representation should allow for a discriminating similarity criterion

Registration Part 3

- Matching and Optimization: How should we use the (local or global) shape/surface information to align or register models?
  - Use discrete feature matching to compute a transformation, e.g. Generalized Hough Transform or Geometric Hashing
  - Iterative minimization of a distance function, e.g. Iterative Closest Points (ICP)

Audette 2000

Overview

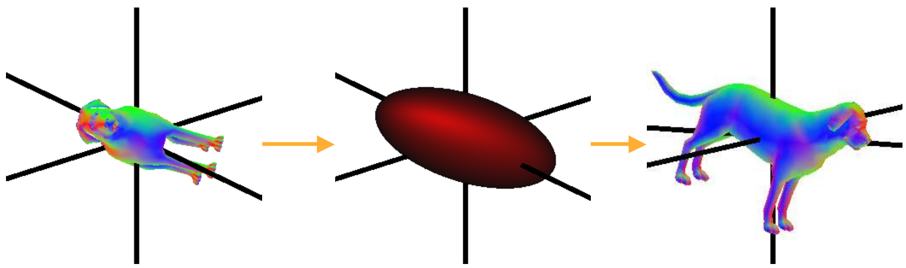
- Sub-problems within registration
- Placing models in a canonical pose or coordinate system
- Methods for pair-wise model registration
  - ICP
  - Generalized Hough Transform
  - Geometric Hashing

Normalization

- Use PCA to place models into a canonical coordinate frame

Covariance Matrix Computation

Principal Axis Alignment



Steps for finding principal axes

- Translate point set $\{p_i\}$ to origin by center of mass:

$$\mathbf{c} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{p}_i$$

$$\mathbf{q}_i = \mathbf{p}_i - \mathbf{c}$$

- Result is new point set $\{q_i\}$

Steps for finding principal axes

- Calculate second-order covariance matrix:

$$\mathbf{M} = \frac{1}{n}\sum_{i=1}^{n}\begin{bmatrix} q_i^x q_i^x & q_i^x q_i^y & q_i^x q_i^z \\ q_i^y q_i^x & q_i^y q_i^y & q_i^y q_i^z \\ q_i^z q_i^x & q_i^z q_i^y & q_i^z q_i^z \end{bmatrix}$$

Steps for finding principal axes

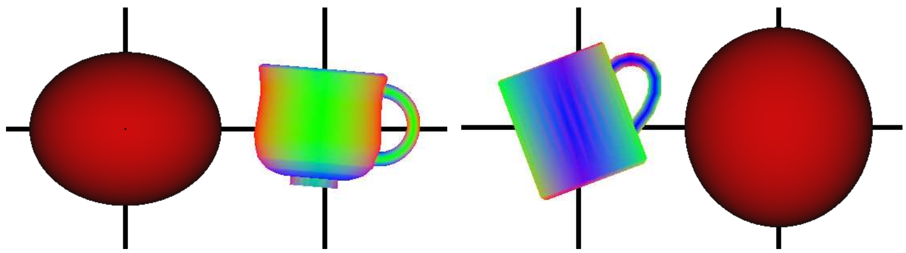
- Decompose symmetric covariance matrix:

$$\mathbf{M} = \mathbf{USU}^t \qquad \mathbf{U} = \begin{bmatrix} A_x & A_y & A_z \\ B_x & B_y & B_z \\ C_x & C_y & C_z \end{bmatrix} \quad \mathbf{S} = \begin{bmatrix} \lambda_a & 0 & 0 \\ 0 & \lambda_b & 0 \\ 0 & 0 & \lambda_c \end{bmatrix}$$

- Matrix U contains 3 principal axes (eigenvectors) as rows: A, B, C
- Matrix S contains eigenvalues

Problems with PCA

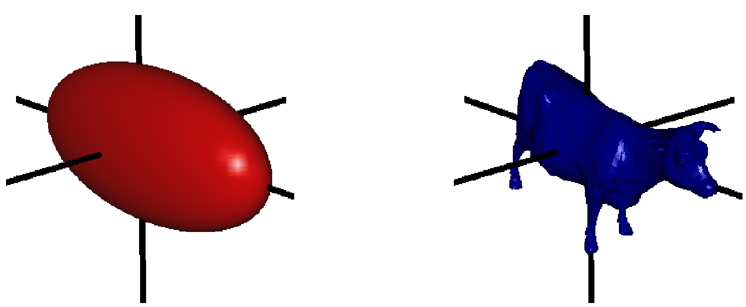
- Doesn't always work
  - Only second order information



M. Kazhdan

Problems with PCA

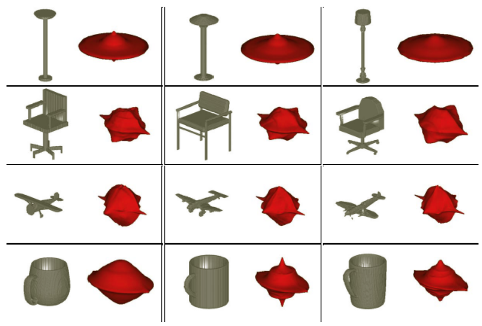
- Directions of principal axes are ambiguous



S. Rusinkiewicz

Reflective Symmetry Descriptors

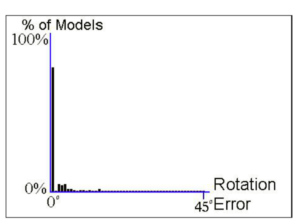
- Align to axes of symmetry rather than principal components



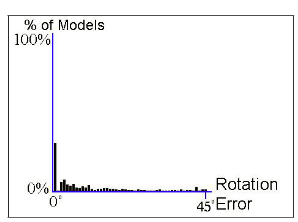
M. Kazhdan

Reflective Symmetry Descriptors

- Aligns objects more like humans
- Performs better than PCA in aligning objects within a class



Reflective Symmetry Descriptor



Principal Component Analysis

M. Kazhdan

Overview

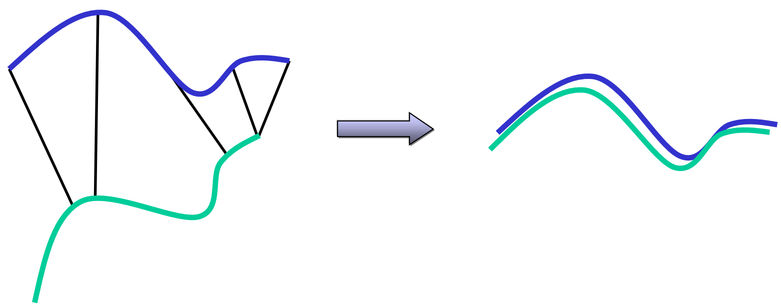
- Sub-problems within registration
- Placing models in a canonical pose or coordinate system
- Methods for pair-wise model registration
  - ICP
  - Generalized Hough Transform
  - Geometric Hashing

Iterative Closest Points (ICP)

- Besl & McKay, 1992
- Start with rough guess for alignment
- Iteratively refine transform

ICP

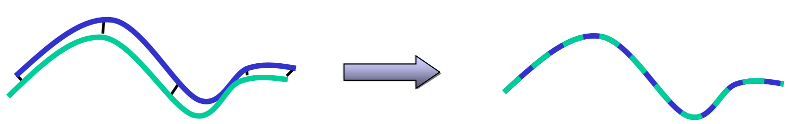
- Assume closest points correspond to each other, compute the best transform…



S. Rusinkiewicz

ICP

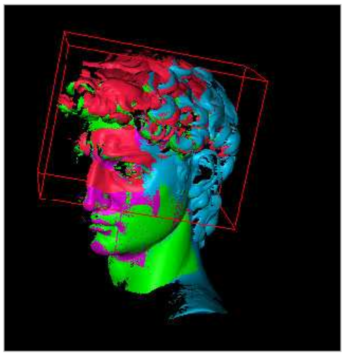
- … and iterate to find alignment
- Converges to some local minimum
- Correct if starting position “close enough“



S. Rusinkiewicz

Aligning Scans

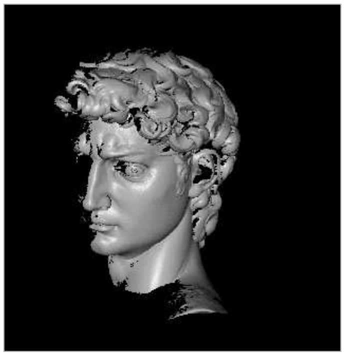
- Start with manual initial alignment



S. Rusinkiewicz [Pulli]

Aligning Scans

- Improve alignment using ICP algorithm



S. Rusinkiewicz [Pulli]

ICP Variants

- Variants on the following stages of ICP have been proposed:

1. Selecting source points (from one or both meshes)
2. Matching to points in the other mesh
3. Weighting the correspondences
4. Rejecting certain (outlier) point pairs
5. Assigning an error metric to the current transform
6. Minimizing the error metric w.r.t. transformation

S. Rusinkiewicz

Comparison of ICP Variants

Algorithm	Selection of Points	Matching Points	Weighting and Rejecting Pairs	Error Metric	Minimizing Error	Global Registration
[Faugeras 86]	Feature identification	Corresponding feature finding		Point-to-point		
[Hanson 81], [Arun 87], [Horn 87], [Horn 88], [Walker 91], [Eggert 97]	Assumed given	Assumed given	Constant or user-specified	Point-to-point	Closed-form solution for best transform given one set of correspondences	
[Chen 91]	Uniform subsampling, smooth regions	Normal shooting		Point-to-plane	Iterative minimization	New-> all previous
[Stein 92]						
[Besl 92]	All	Closest point	Constant	Point-to-point	Iterative minimization, accelerated by extrapolation	
[Szeliski 94]		Closest point		Point-to-point	Iterative minimization	
[Turk 94]	Uniform subsampling	Closest point	Distance threshold; reject edge points; weight is normal dot camera vector	Point-to-point	Iterative minimization, accelerated by extrapolation	Align all to cylindrical anchor scan
[Godin 94]	All in both meshes	Closest point with compatible color	Distance threshold; weighted by compatibility and distance	Point-to-point	Iterative minimization	
[Blais 95]	Uniform subsampling	Projection	Distance threshold	Point-to-point	Search in transform space using simulated annealing	Search for all transforms simultaneously
[Stoddart 96]	Assumed given	Assumed given		Point-to-point	Gradient descent	Find all transforms simultaneously
[Masuda 96]	Random sampling	Closest point, accelerated with k-d tree	Distance threshold	Point-to-point	Iterative minimization; find transform that minimizes median of squared distances after several random subsamplings	New to integration of all previous
[Bergevin 96]	Uniform subsampling, smooth regions	Normal shooting	Reject if dot product of normals is negative	Point-to-plane	Iterative minimization	Iterated all-to-all ICP
[Simon 96]	All	Closest point, accelerated with k-d tree and point cache	Distance threshold	Point-to-point	Iterative minimization, accelerated by separate extrapolation of rotation and translation	
[Dorai 96], [Dorai 98]	All	Normal shooting, accelerated by projection plus search	Rejection based on pair-to-pair compatibility	Point-to-plane	Iterative minimization	
[Dorai 97]	All	Normal shooting	Weighted based on effect of scanner noise on normal	Point-to-plane	Iterative minimization	
[Benjemaa 97]	All	Closest point, accelerated using z buffer search		Point-to-point	Iterative minimization	Iterated all-to-all ICP
[Johnson 97a]						
[Johnson 97b]	All	Closest point in shape+color space, accelerated with k-d tree		Point-to-point, shape+color	Iterative minimization	
[Neugebauer 97]	Uniform subsampling	Projection	Reject points with distance greater than 3 sigma	Point-to-plane	Iterative minimization via Levenberg-Marquardt	Align all scans simultaneously
[Weik 97]	Select points with high intensity gradient	Projection followed by search for sample with similar image intensity and gradient	Reject projected points that are occluded in the source mesh	Point-to-point	Iterative minimization	
[Pulli 97]	All	Like [Weik 97], but project complete images and do image alignment		Point-to-point	Iterative minimization	Scan-to-scan ICP, then global optimization of transforms using pre-computed point pairs
[Chen 98], [Chen 99]		Closest point		Number of corresponding points within threshold	points on P, and considering all possible xforms that map these to plausible corresponding points on Q	
[Pulli 99], [Levoy 00]	Random sampling in both meshes	Closest point with compatible normals	Distance threshold; reject edge points; reject some percentage of pairs with largest distances	Point-to-plane	Iterative minimization	Like [Pulli 97], but process scans in order of how many others they overlap
[Williams 00]				Point-to-point	Iterative minimization	Simultaneous alignment with error modeling

http://graphics.stanford.edu/~smr/ICP/comparison/

Comparison of ICP Variants

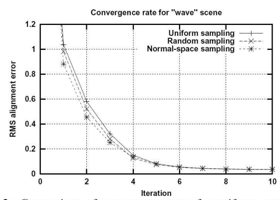


Figure 2: Comparison of convergence rates for uniform, random, and normal-space sampling for the "wave" meshes.

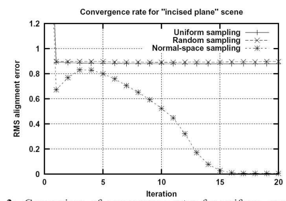


Figure 3: Comparison of convergence rates for uniform, random, and normal-space sampling for the "incised plane" meshes. Note that, on the lower curve, the ground truth error increases briefly in the early iterations. This illustrates the difference between the ground truth error and the algorithm's estimate of its own error.

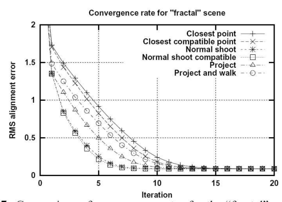


Figure 7: Comparison of convergence rates for the "fractal" meshes, for a variety of matching algorithms.

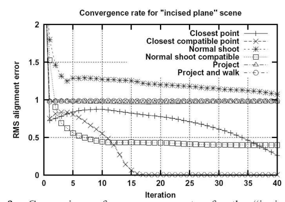
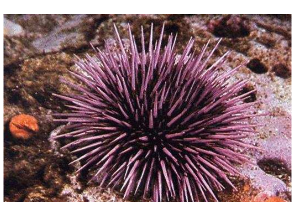


Figure 9: Comparison of convergence rates for the "incised plane" meshes, for a variety of matching algorithms. Normal-space-directed sampling was used for these measurements.

Rusinkiewicz and Levoy, *Efficient Variants of the ICP Algorithm*

One ICP Caveat

"It can safely be predicted that the proposed registration algorithm will have difficulty correctly registering 'sea urchins' and 'planets'."



Besl and McKay, *A Method for Registering 3-D Shapes*, 1992

Pair-wise Registration or Matching: Three Approaches (out of many)

- Generalized Hough transform
- "Curve" Geometric Hashing
- "Basis" Geometric Hashing

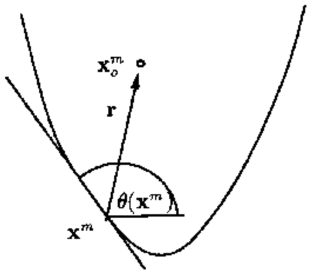
All are "model-based" approaches which use *a priori* knowledge about the models to populate a lookup table which is used to speed up the matching/registration process.

Generalized Hough Transform
(First for 2D Images)

- Every boundary point (of the object) in image votes
- Votes are cast for each object / transformation consistent with the presence of that point
- At the end, objects with most votes win

GHT: Preprocessing

- Simplified 2D case with translation only



  - For each point x_m, find angle of tangent $\theta(x_m)$ and vector r to reference point x_0
  - Form table indexed by $\theta(x_m)$, storing r and object ID

- For rotation or 3D objects, table has many dimensions, each point x has many entries

S. Rusinkiewicz, image from Hecker and Bolle

GHT: Identification

- For each point:
  - – Compute angle of tangent
  - – Look up in table
  - – For each object found:
    - Compute origin of object consistent with this point
    - Vote for the object at that location
- At end:
  - – Find clusters of votes for the same object
  - – Position of cluster gives location of object

S. Rusinkiewicz

Curve Geometric Hashing

- Compute “footprints” of each subcurve – invariant under rotation, translation
  - For example, in 2D, arc-length vs. turning-angle
  - Boundary curves must be (heuristically) segmented into subcurves first
- Preprocessing:
  - Create a table indexed by footprint
  - Each entry contains object ID and location of footprint along curve

CGH: Identification

- Find footprints in image
- For each model:
  - Each footprint votes for a relative shift
  - Peaks in the histogram are identified
  - Second pass to confirm the presence of the object and find the location by least-squares

S. Rusinkiewicz

Basis Geometric Hashing

- Objects are represented as sets of local "features" which allow for matching or recognition with partial occlusion (features can be points, line segments, etc.)
- Features are indexed with a function that is invariant to the transformation(s) being considered
- Preprocessing:
  - For each tuple *b* of features, compute location (ξ,η) of all other features in basis defined by *b*
  - Create a quantized hash table indexed by (ξ,η)
  - Each entry contains *b* and object ID

BGH: Identification

- Find features in target image
- Choose an arbitrary basis *b*'
- For each feature:
  - Compute (ξ',η') in basis *b*'
  - Look up in table and vote for (Object, *b*)
- For each (Object, *b*) with many votes:
  - Compute transformation that maps *b* to *b*'
  - Confirm presence of object, using all available features

Basis Geometric Hashing

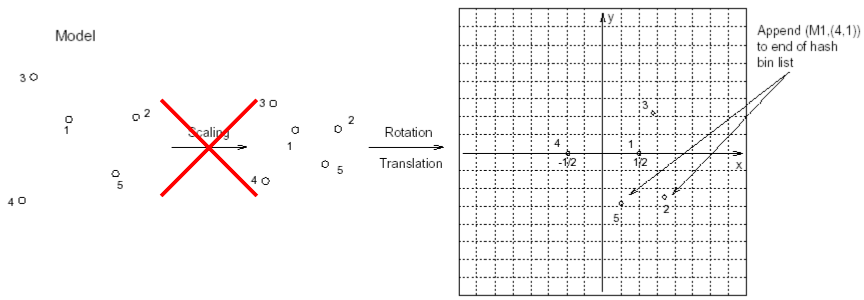


Figure 1. Determining the hash table entries when points 4 and 1 are used to define a basis. The models are allowed to undergo rotation, translation, and scaling. On the left of the figure, model M_1 comprises five points.

Wolfson and Rigoutsos, *Geometric Hashing, an Overview,* 1997

Basis Geometric Hashing

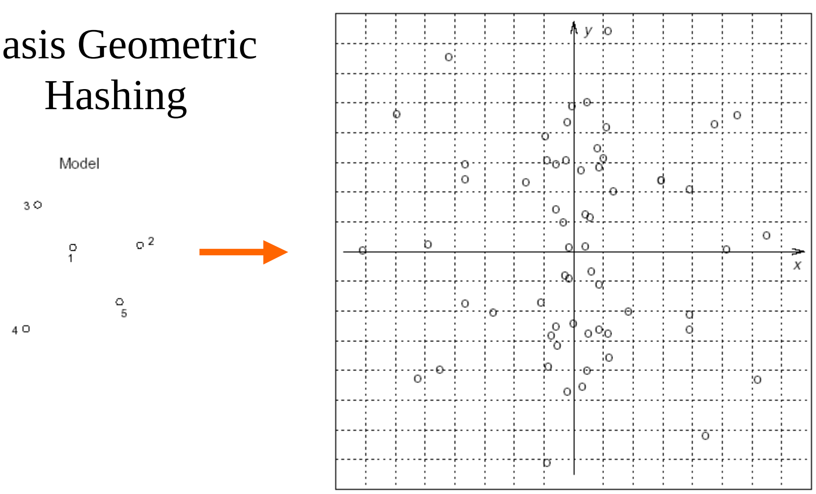


Figure 2. The locations of the hash table entries for model M_1. Each entry is labeled with the information "model M_1" and the basis pair (i, j) used to generate the entry. The models are allowed to undergo rotation, translation, and scaling.

Wolfson and Rigoutsos, *Geometric Hashing, an Overview,* 1997

Basis Geometric Hashing

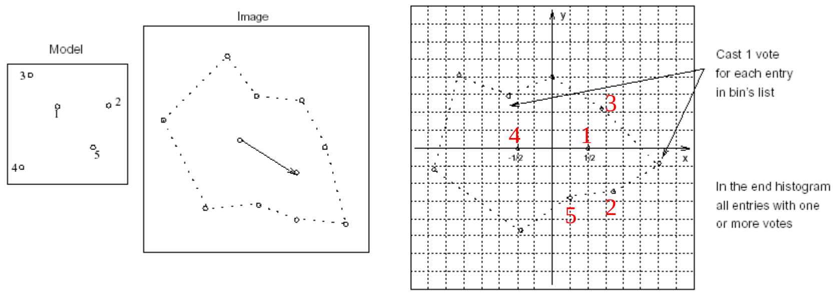
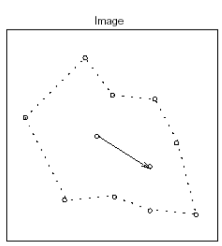
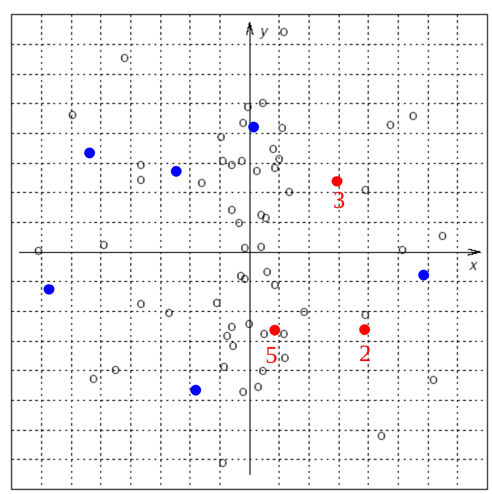


Figure 3. Determining the hash table bins that are to be notified when two arbitrary image points are selected as a basis. Similarity transformation is allowed.

Wolfson and Rigoutsos, *Geometric Hashing, an Overview*, 1997

Basis Geometric Hashing





- Hash table entries contain (M_1, (4,1)), a consistent match
- Hash table entries contain (M_k, (x,y)), k ≠ 1, (x,y) ≠ (4,1) (or nothing)

Figure 2. The locations of the hash table entries for model M_i. Each entry is labeled with the information "model M_i" and the basis pair (i, j) used to generate the entry. The models are allowed to undergo rotation, translation, and scaling.

Wolfson and Rigoutsos, *Geometric Hashing, an Overview*, 1997

BGH Complexity

With:

M models in the database (hash table),

n features per model

S features in a scene

C features needed to form a basis tuple

Preprocessing step is $O(Mn^{C+1})$

Matching/recognition is $O(HS^{C+1})$ where *H* is the complexity of processing a hash-table bin

Grimson and Huttenlocher, 1990

GHT and Geometric Hashing Comparison

- Similarities:
  - Image features "vote" for objects
  - Recognition time independent of size of database
- Differences:
  - Generalized Hough transform and curve geometric hashing need a clustering step because all features are used in the lookup process
  - Basis geometric hashing requires selecting "good" features which are the only ones used in the lookup process (more "good" features can be used for further iterations)

S. Rusinkiewicz

Algorithm Sensitivities

- Geometric Hashing
  - A relatively sparse hash table is critical for good performance
  - Method is not robust for cluttered scenes (full hash table) or noisy data (uncertainty in hash values)
- Generalized Hough Transform
  - Does not scale well to multi-object complex scenes
  - Also suffers from matching uncertainty with noisy data

Grimson and Huttenlocher, 1990

Acknowledgements

Tom, Szymon, and Misha