Geometric Hashing

- Originally, a Computer Vision technique for matching objects in a scene to models in a database.
- Based on the idea of storing invariant features of models in a hash table and finding matching features of the query object in this hash table.

Motivation

- Object recognition (ultimate goal of most computer vision research).
- Inputs:
  - A database of objects.
  - A scene or image to recognize.
- Problems:
  1. Objects in the scene undergo some transformations (e.g., translation, rotation. Other?).
  2. Objects may partially occlude each other.
  3. Computationally expensive to retrieve each object from database and compare it against the observed scene.

Problem Statement

- Recognition under Similarity Transformation:
- *"Is there a transformed (rotated, translated and scaled) subset of some model point-set which matches a subset of the scene point-set?"*

General Framework

- Two stage algorithm:

1. Preprocessing (for each model):
- For each feature points pair:
  - Define a local coordinate basis on this pair.
  - Compute and quantize all other feature points in this coordinate basis.
  - Record (model, basis) in a hash table.

General Framework

2. Online recognition (given a scene, extract feature points):

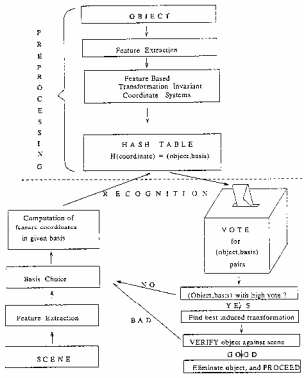
a) Pick arbitrary ordered pair:
   - Compute the other points using this pair as a basis.
   - For all the transformed points, vote all records (model, basis) appear in the corresponding entry in the hash table, and histogram them.

b) Matching candidates: (model, basis) pairs with large number of votes.

c) Recover the transformation that results in the best least-squares match between all corresponding feature points.

d) Transform the features, and verify against the input image features (if fails, repeat to 1).

Algorithm



Complexity

Assume m=n, and k is the number of points to define the basis.

- Preprocessing: $O(n^{k+1})$ for a single model.
- Recognition: $O(n^{k+1})$ against all objects in the database.

Bases for Various Transformations

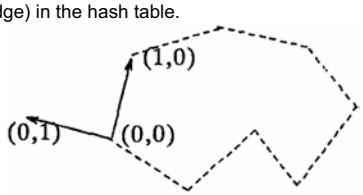
1. Translation in 2D and 3D.
   - 1-point basis.
   - $O(n^2)$.
2. Similarity transformation in 2D.
   - 2-point basis.
   - $O(n^3)$.
3. Similarity transformation in 3D.
   - 3-point basis.
   - $O(n^4)$.

Bases for Various Transformations

4. Affine transformation
   - 3-point basis.
   - $O(n^4)$
5. Projective transformation
   - 4-point basis.
   - $O(n^5)$

Recognition of Polyhedral Objects

- Polygonal objects
  - Choose an edge as the basis, record (model, basis edge) in the hash table.



  - Preprocessing and recognition is $O(n^2)$.

Summary

- Ability to recognize objects that have undergo an arbitrary transformation.
- Can perform partial matching.
- Efficient and can be parallelized easily.
- Use transformation-invariant access key to the hash table.
- Two phases (preprocessing and recognition).
- Require a large memory to store hash table.

Automated Multiple Structure Alignment and Detection of a Common Substructural Motifs

N. Leibowitz, Z.Y. Fligelman, R. Nussinov, H.J. Wolfson

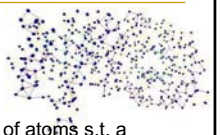
Motivation

- Finding structural core of a protein family
  - Clue to protein function & drug design
- Protein structure significantly more conserved than protein sequence
- Goal:
  - Given M proteins (by backbone Cα atoms)
  - Find a large congruent substructure

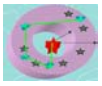
Pairwise Structural Comparison by Geometric Hashing

- Geometric hashing
  - Sequence independent
  - 3 stages:
    - Detection of seed matches
      - Model molecule & target molecules
      - Preprocessing: Cα atoms of model stored in hash table
      - Querying: hash table is queried for matching representation.
      - Large enough matching atoms serve as seed matches
    - Clustering of seeds of similar transformations
    - Extension of transformations by applying to entire proteins
      - Sufficiently close Cα-atom pairs are added to the match

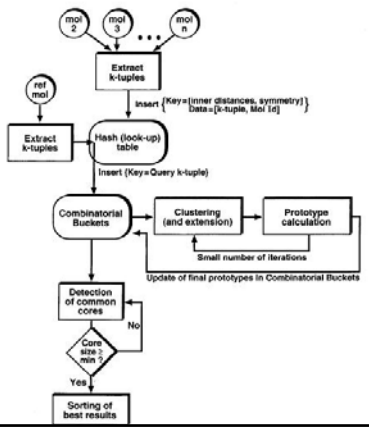
MUSTA: Definitions



- Input: 3D coordinates (Cα atoms) of M molecules (P_i, i = 1..M)
- Output: largest geometric configuration of atoms s.t. a congruent copy appears in all molecules (a.k.a. geometric core)
- One molecule (e.g. P_1) – ***reference molecule***
- Remaining (P_2..P_M) – ***source molecules***
- M-1 transformations (T_{12}, T_{13}, …, T_{1M}) - ***multidimensional transformation***
- k-tuple: ordered set of k (k>=3) atoms. (k-1) atoms within 3D spherical shell centered at one atom

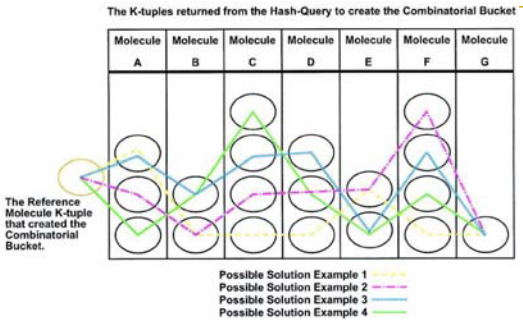


MUSTA Overview



Combinatorial buckets

- Combinatorial Buckets (CB)
  - Grouping all multidimensional transformations for the same set of atoms in reference protein
  - Each k-tuple of the reference molecule defines a bucket
  - The bucket members are all the congruent k-tuples
  - Each path through the CB forms a multi-transformation
    - At least k atoms as geometric core



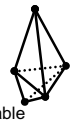
- Complexity
  - kt- reduced complexity: implicit multi-transformation & efficient processing
  - Fully preserved tuple: only keep reference seeds that have at least one congruent copy in each source

Multiple Structure Alignment Algorithm (MUSTA)

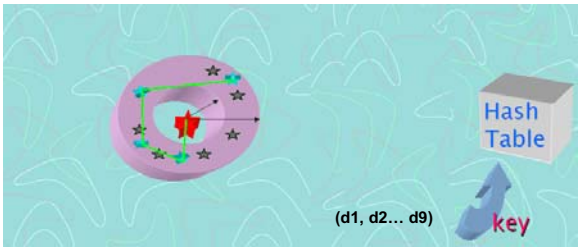
- 3 stages:
  - Detection of seed matches & forming combinatorial buckets
  - Clustering & extension of seed matches
  - Computation of highest scoring multi-transformation

MUSTA: Detection of seed matches & forming CBs

- Detect k-tuples appearing in all molecules
- Geometric hashing
  - Each k-tuple represented by a parameter vector (invariant to 3D rotation & translation)
  - Insert k-tuple of sources into hash table by its invariants
    - Congruent k-tuples reside together
    - k=5: 5-tuple has 9 invariants -> 9 dimensional hash table
  - Query the hash table with reference's k-tuple as key
    - Store results in the associated combinatorial bucket



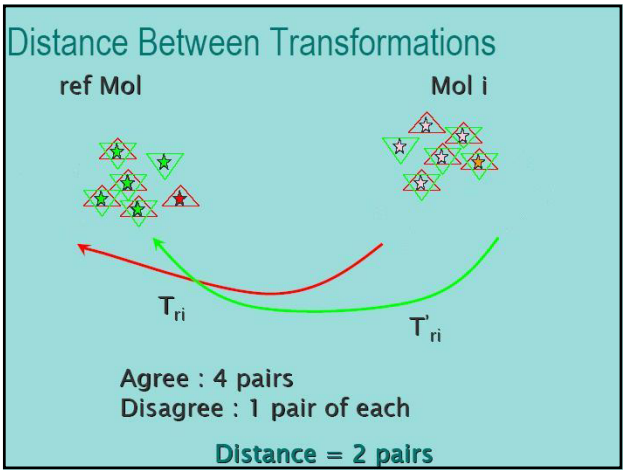
Hashing mechanism



MUSTA: seed extension & clustering

- Iterative seed extension
  - Compute & apply each transformation from CB
  - Enlarge seed match
    - Add source atoms that coincide with reference atoms
    - Only use atom pairs appearing in fully preserved CBs to bias toward good transformation
- Iterative clustering proximate transformations
  - Goal: as large as possible geometrically congruent core
  - Criteria: transformations from the same cluster map relevant 3D points to almost identical locations
  - Distance between transformation: number of atoms mapped to different locations

Distance Between Transformations



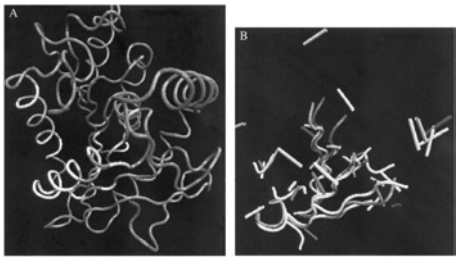
MUSTA: Cluster Prototype

- Prototype transformation
  - Representative of one cluster
  - Smallest RMSD of match list pairs
- M-1 sets of prototype transformations
  - M-1 sources vs. 1 reference
- Further reduce complexity by removing transformations with small maximal match list

MUSTA: Compute highest scoring multi-transformations

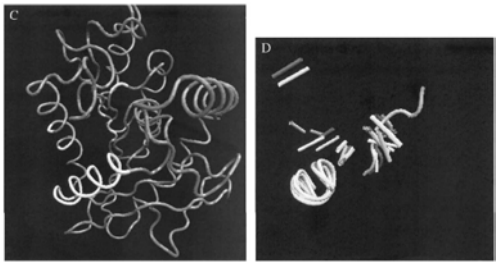
- Remove buckets which have not survived the clustering
- Combine all prototype transformations
- Core verification
  - Intersect M-1 match lists of all combinations by each CB
  - Keep ranked results that are above a threshold

Random set of proteins



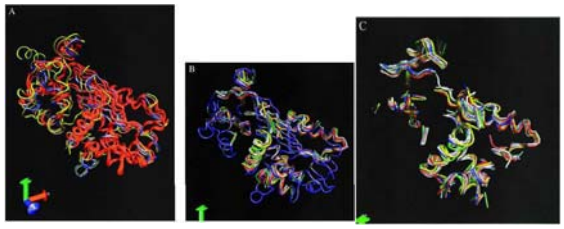
- Align random set of proteins (3, 5, 7, 9, 10 proteins)
- 1din, 1nba, 1tml, 1akz, 1nsy, 1dea, 3pgm, 1sbc, 1eaa, iula

Random set of proteins



- Align random set of proteins (3, 5, 7, 9, 10 proteins)
- 1din, 1nba, 1tml, 1akz, 1nsy, 1dea, 3pgm, 1sbc, 1eaa, iula

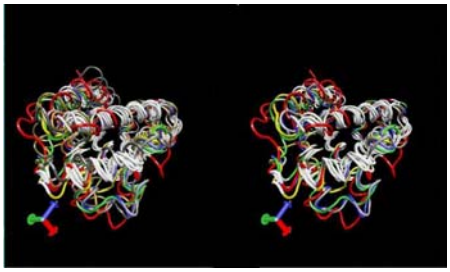
Serpins



First 6 molecules (core highlighted in red) | **11 molecules** | **Core alone of 11 molecules**

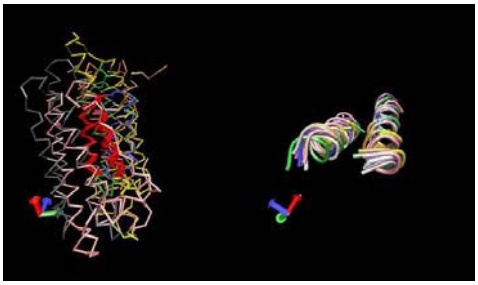
Running time = 10-12 seconds

Globins



Running time = 1min (average)

Cal-binding



Running time = 8 sec

MUSTA: Summary

- Sequence – independent structural alignment
  - No order at all on the aligned atoms
- Input: atomic coordinates
- Output: simultaneous structural alignment & core detection
- Geometric hashing of shape invariants to detect initial seeds
- Combinatorial buckets
  - Implicitly define and screen of exponential set of naïve combinations

MUSTA: Limitations

- Proteins in related families
  - Important to preserve sequence
- MUSTA aligns all input molecules
  - Preferable to recognize well-aligned subset
- MASS (Multiple Alignment by Secondary Structures)
  - Efficiency improvement & partial solutions
  - Two-level alignment & geometric hashing
    - Local secondary structure superposition
    - Global atomic superposition