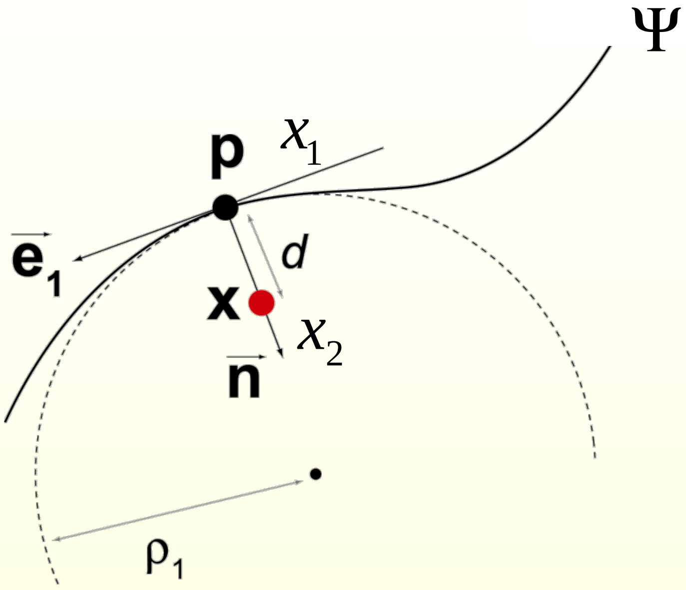
CS164: Shape Matching

Leonidas Guibas
Computer Science Dept.
Stanford University

1891

Distance Fields for Registration

Approximate Squared Distance



For a curve Ψ,

$$F(x,\Psi) = \frac{d}{d - \rho_1} x_1^2 + x_2^2 = \delta_1 x_1^2 + x_2^2$$

in the Frenet frame at *p*

[Pottmann and Hofer 2003]

Approximate Squared Distance

For a curve Ψ,

$$\mathbf{F}(\mathbf{x}, \Psi) = \frac{d}{d\text{-}\rho_1} x_1^2 + x_2^2 = \delta_1 x_1^2 + x_2^2$$

For a surface Φ,

$$\mathbf{F}(\mathbf{x}, \Phi) = \frac{d}{d\text{-}\rho_1} x_1^2 + \frac{d}{d\text{-}\rho_2} x_2^2 + x_3^2 = \delta_1 x_1^2 + \delta_2 x_2^2 + x_3^2$$

ρ_1 and ρ_2 are the principal curvatures

[Pottmann and Hofer 2003]

$\mathbf{F}(\mathbf{x}, \Phi_{\mathbf{P}})$ Using d2Tree

- A kd-tree like data structure for storing approximants of the squared distance function
- Each cell (c) stores a quadratic approximant as a matrix Q_c
- Efficient to query

[Leopoldseder et al. 2003]

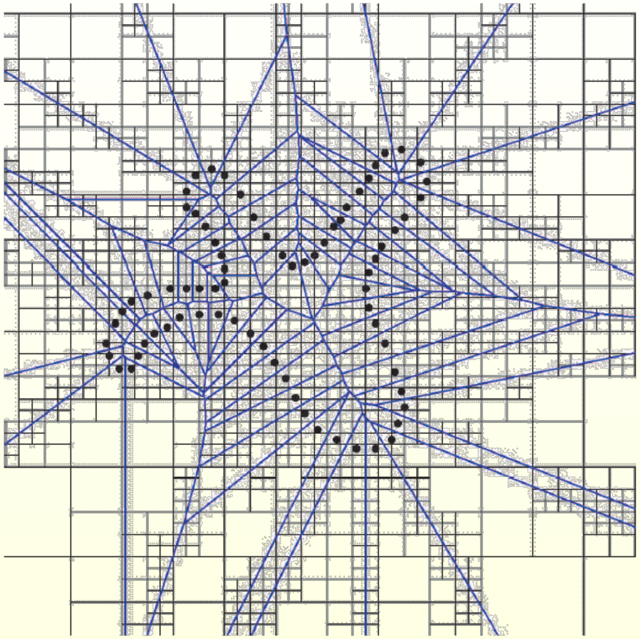
$F(x, \Phi_P)$ Using d2Tree

- A kd-tree like data structure for storing approximants of the squared distance function.
- Each cell (c) stores a quadratic approximant as a matrix Q_c.
- Efficient to query.
- Simple bottom-up construction
- Can be pre-computed for a given curve or surface.

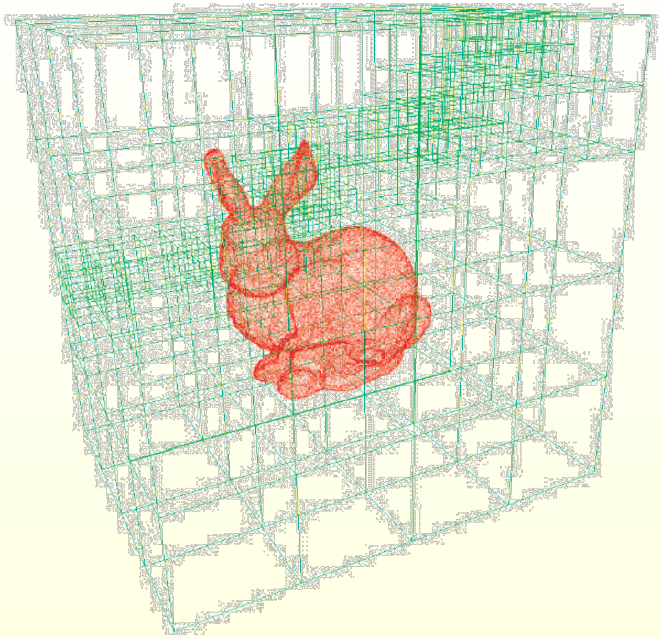
Closest point information implicitly embedded in the squared distance function.

Can therefore be used to implement an implicit ICP.

Example d2trees



2D



3D

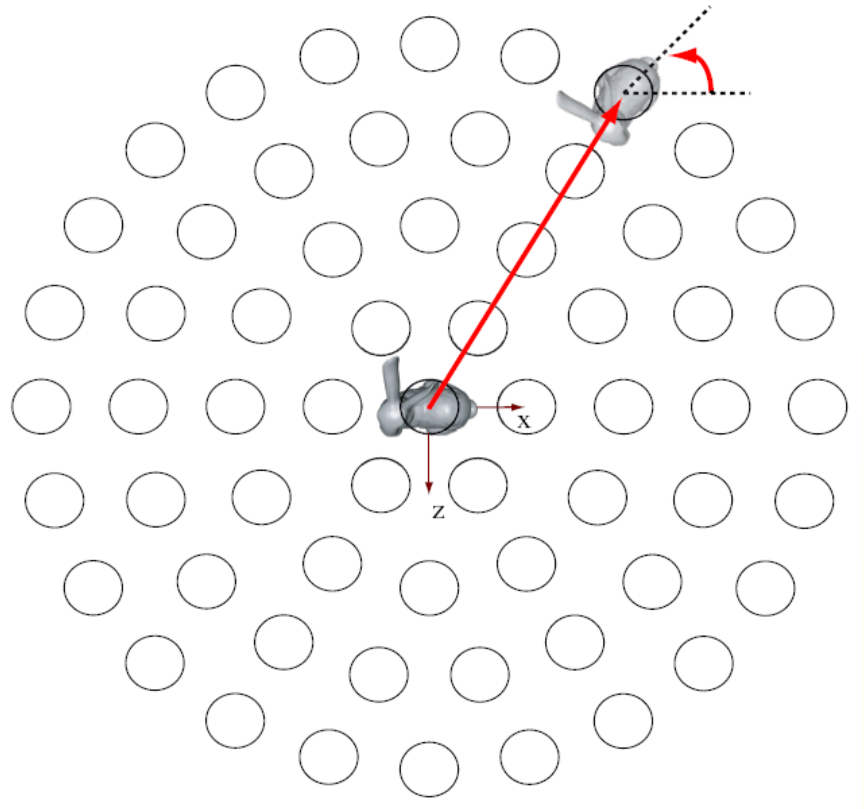
Convergence Funnel

Set of all initial poses of the data surface with respect to the model surface that is successfully aligned using the algorithm.

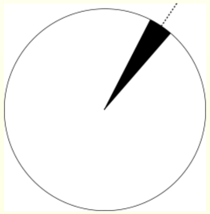
Desirable properties

- broad
- stable

Convergence Funnel



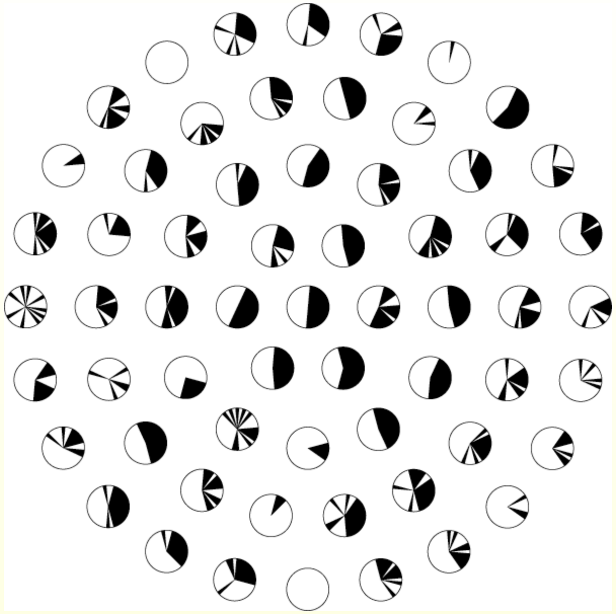
Translation in x-z plane.
Rotation about y-axis.



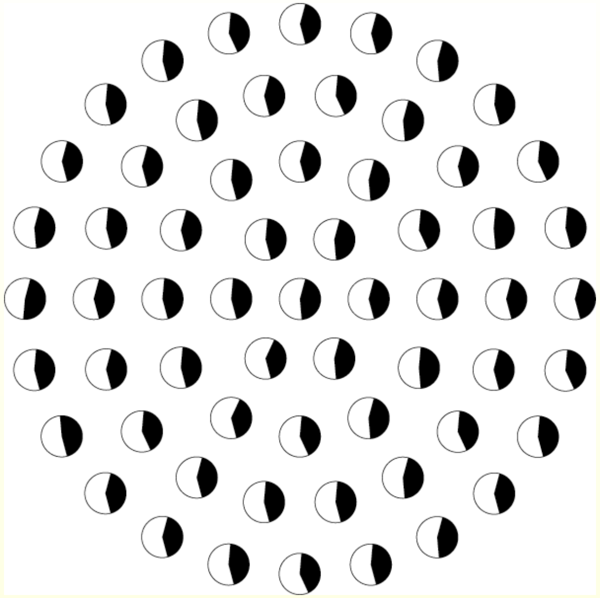
■ Converges

□ Does not converge

Convergence Funnel

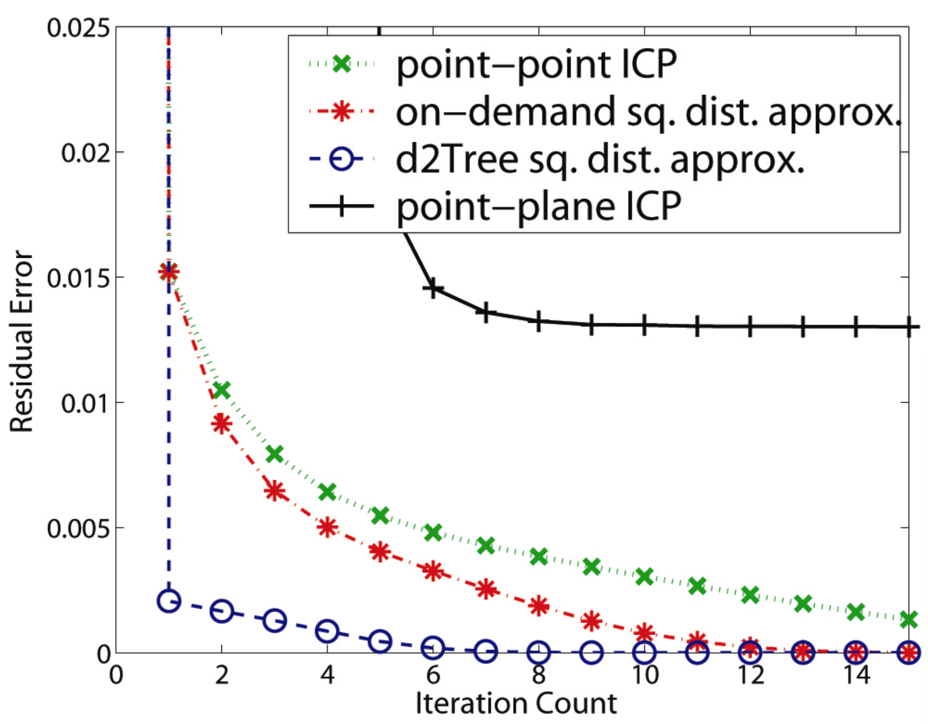


Point-to-plane ICP



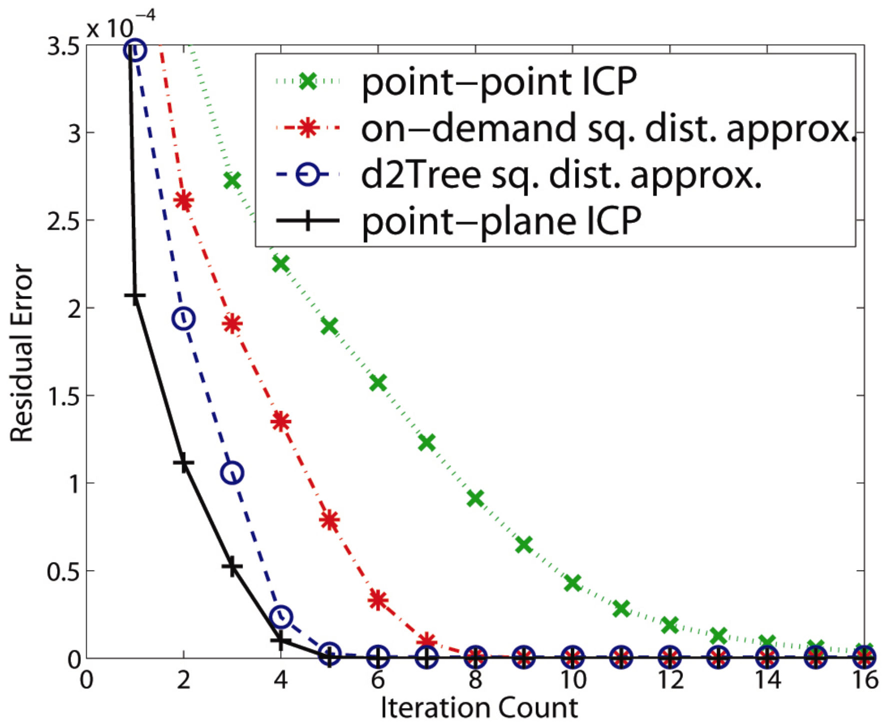
d2Tree

Convergence Rate I



Bad Initial Alignment

Convergence Rate II

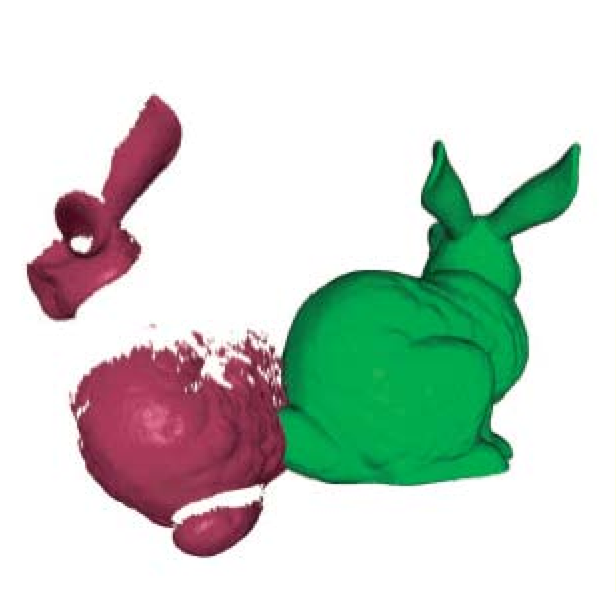


Good Initial Alignment

Partial Alignment

Starting Position

Partial Alignment



After 6 iterations

Partial Alignment

Different sampling density

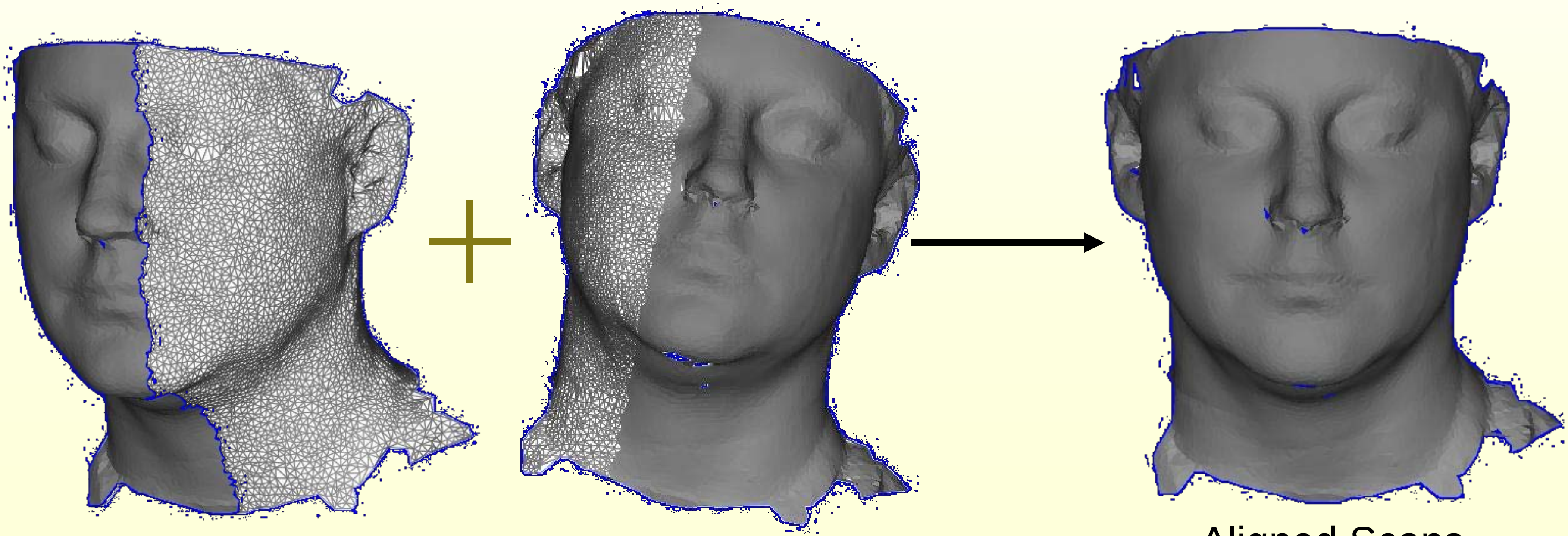
After 6 iterations

Finding Correspondences

[Slides from Michael Kazhdan]

Goal

Given two partially overlapping scans in different frames, compute the transformation that merges the two

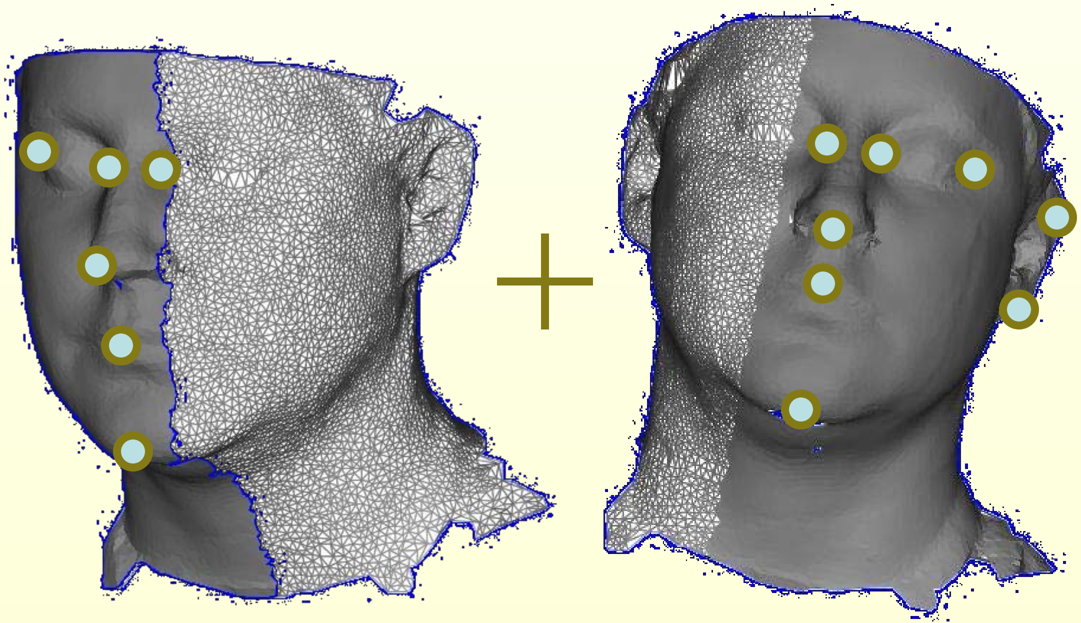


Partially Overlapping Scans

Aligned Scans

Matching Using Feature Points

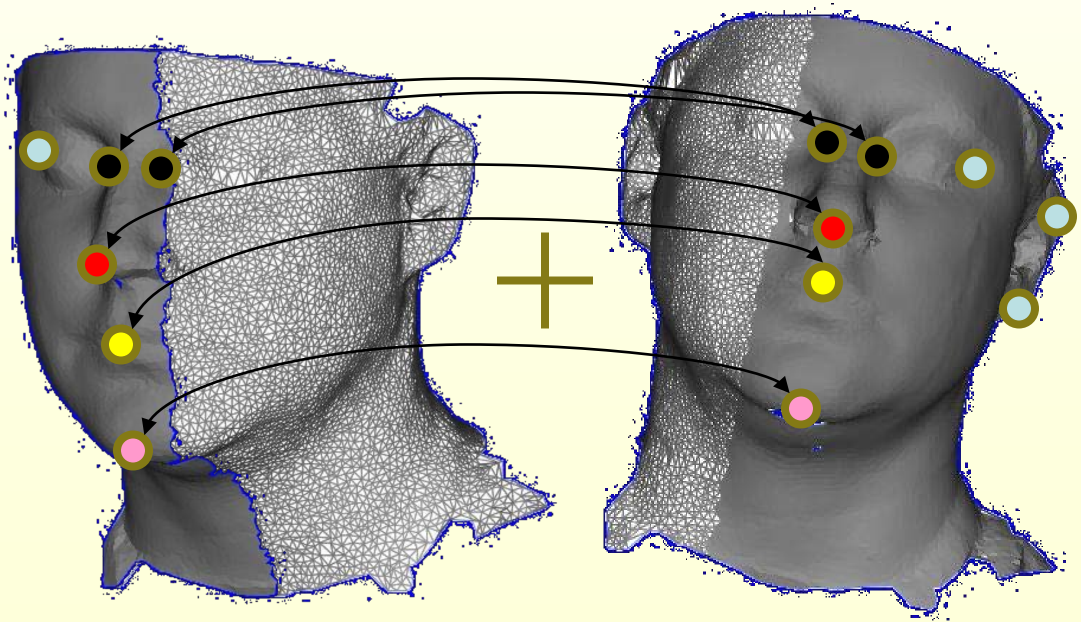
1. Find feature points on the two scans (we'll come back to that issue)



Partially Overlapping Scans

Approach

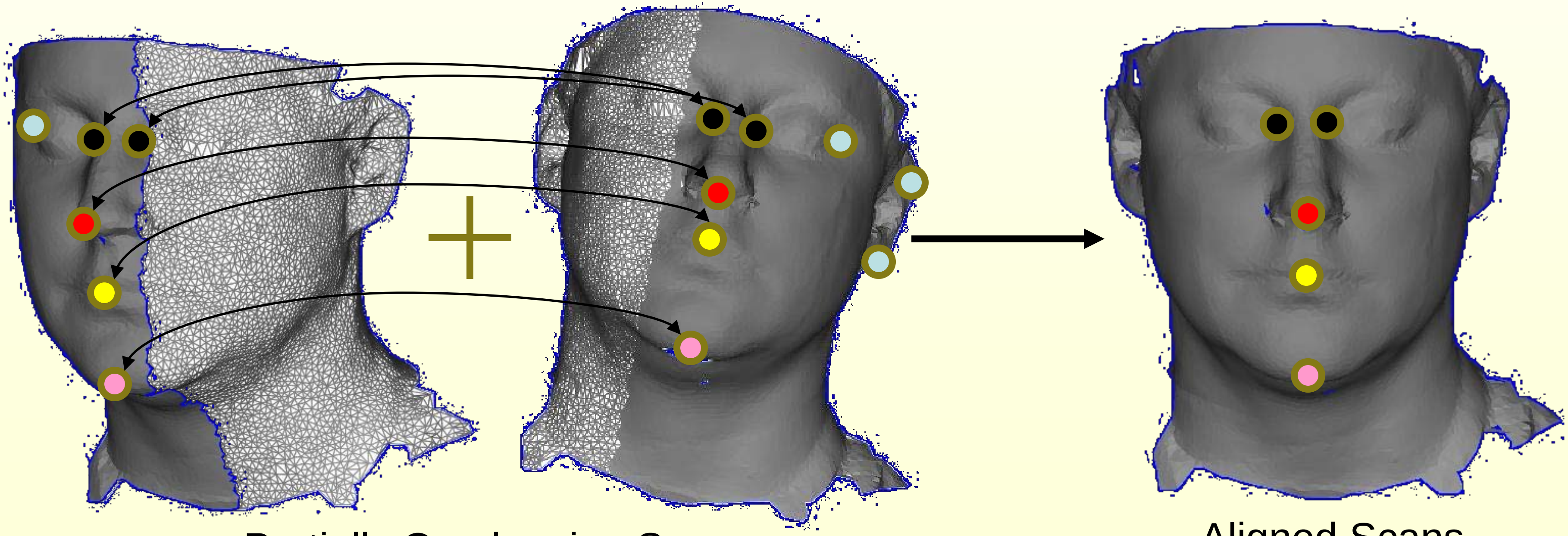
1. (Find feature points on the two scans)
2. Establish correspondences



Partially Overlapping Scans

Approach

1. (Find feature points on the two scans)
2. Establish correspondences
3. Compute the aligning transformation



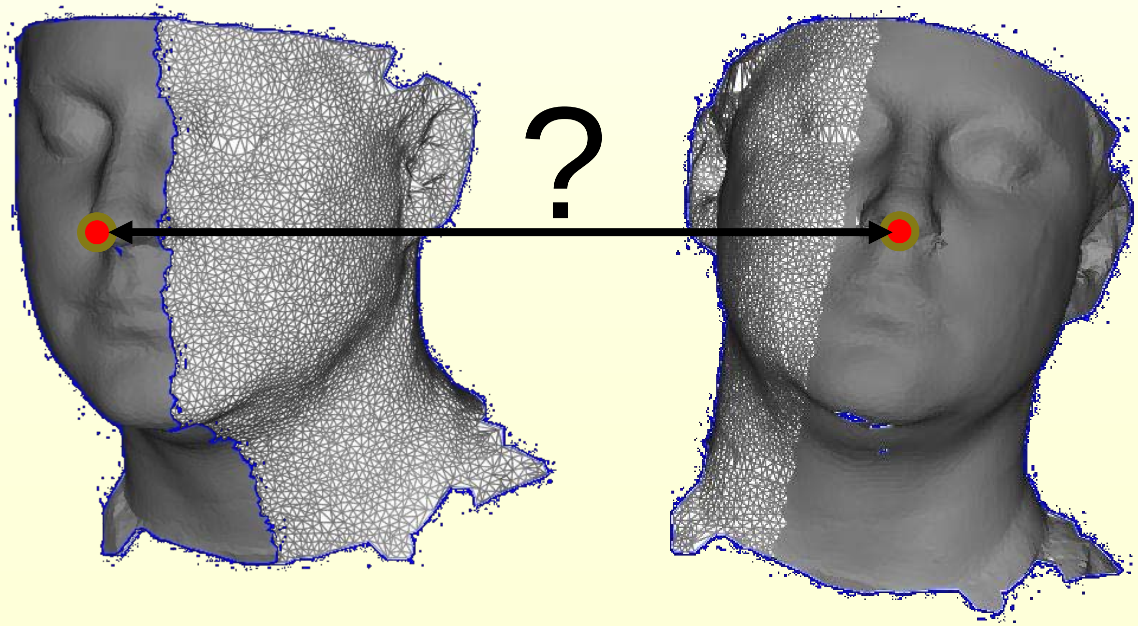
Partially Overlapping Scans

Aligned Scans

Correspondence

Goal:

Identify when two points on different scans represent the same feature

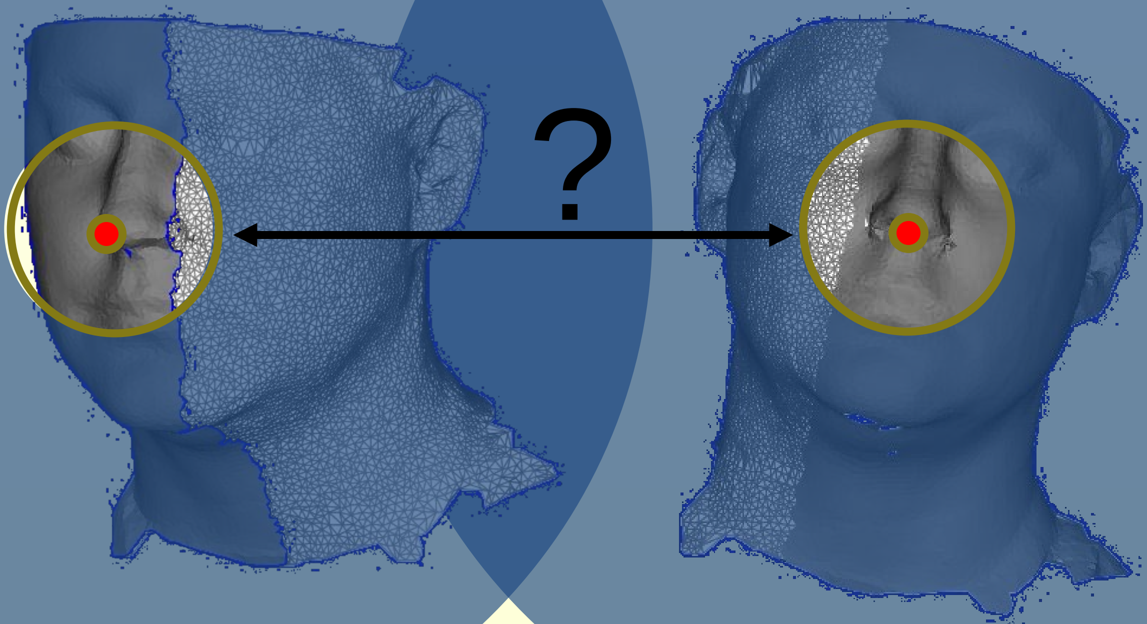


Correspondence

Goal:

Identify when two points on different scans represent the same feature:

Are the surrounding regions similar?



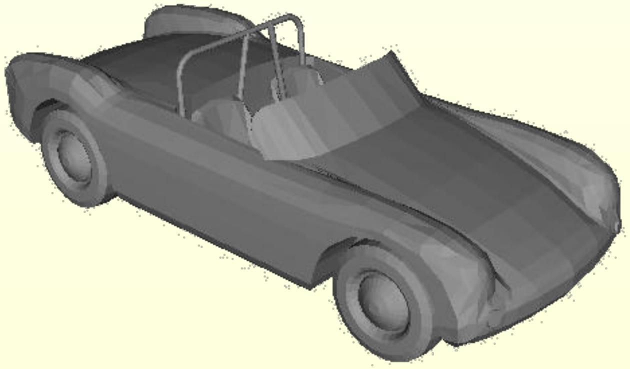
Global Shape Similarity

Global Similarity

More Generally:

Given two models, determine if they represent the same/similar shapes

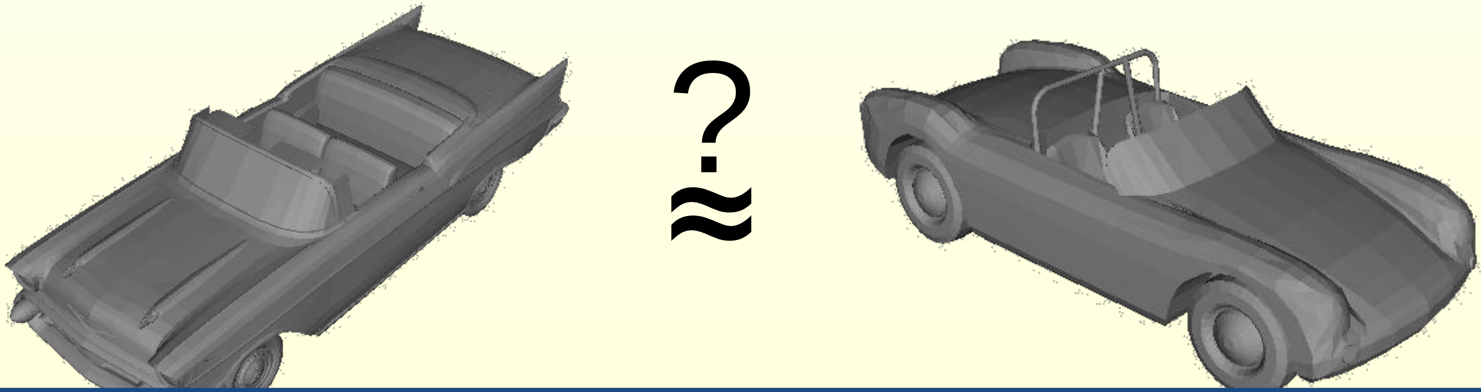
$\overset{?}{\approx}$



Global Correspondence

<u>**More Generally**</u>:

Given two models, determine if they represent the same/similar shapes.



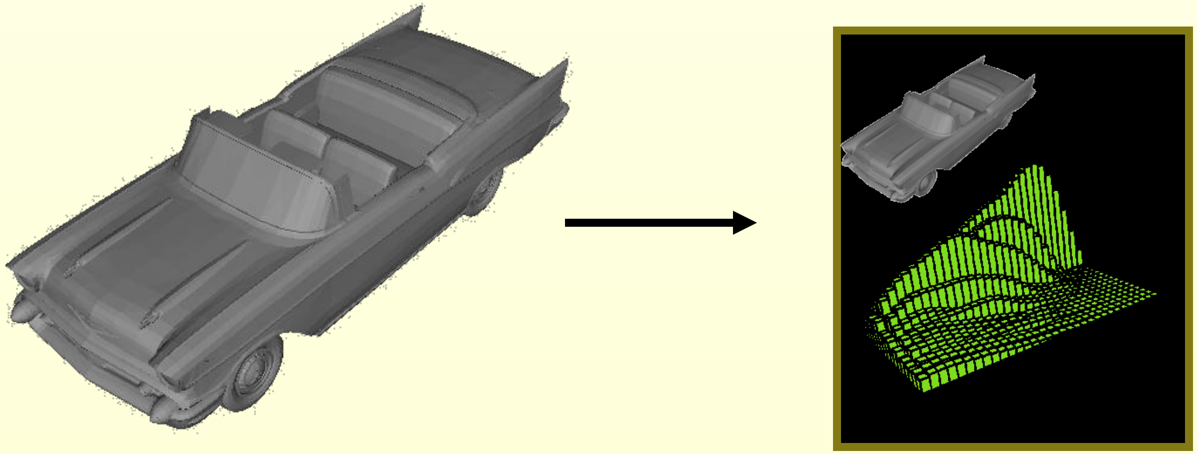
$\overset{?}{\approx}$

Models can have different:
representations, tessellations, topologies, etc.

Global Similarity

Approach:

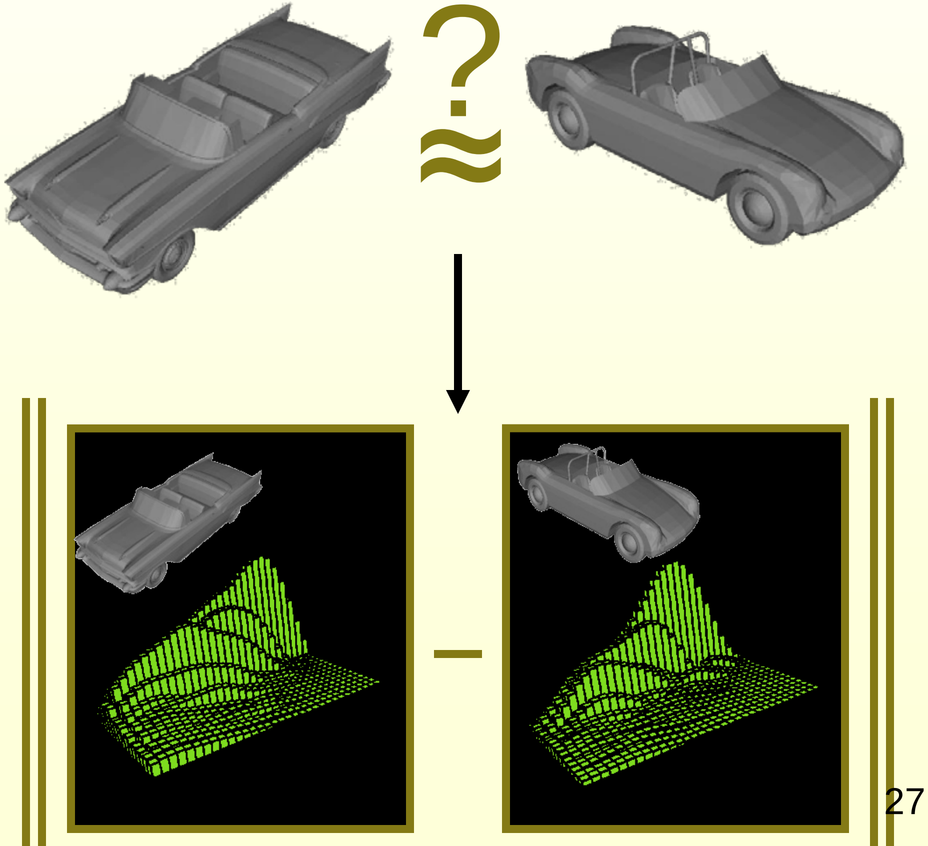
1. Represent each model by a shape descriptor:
   - A structured abstraction of a 3D model
   - That captures salient shape information



Global Similarity

Approach:

1. Represent each model by a shape descriptor
2. Compare shapes by comparing their shape descriptors.

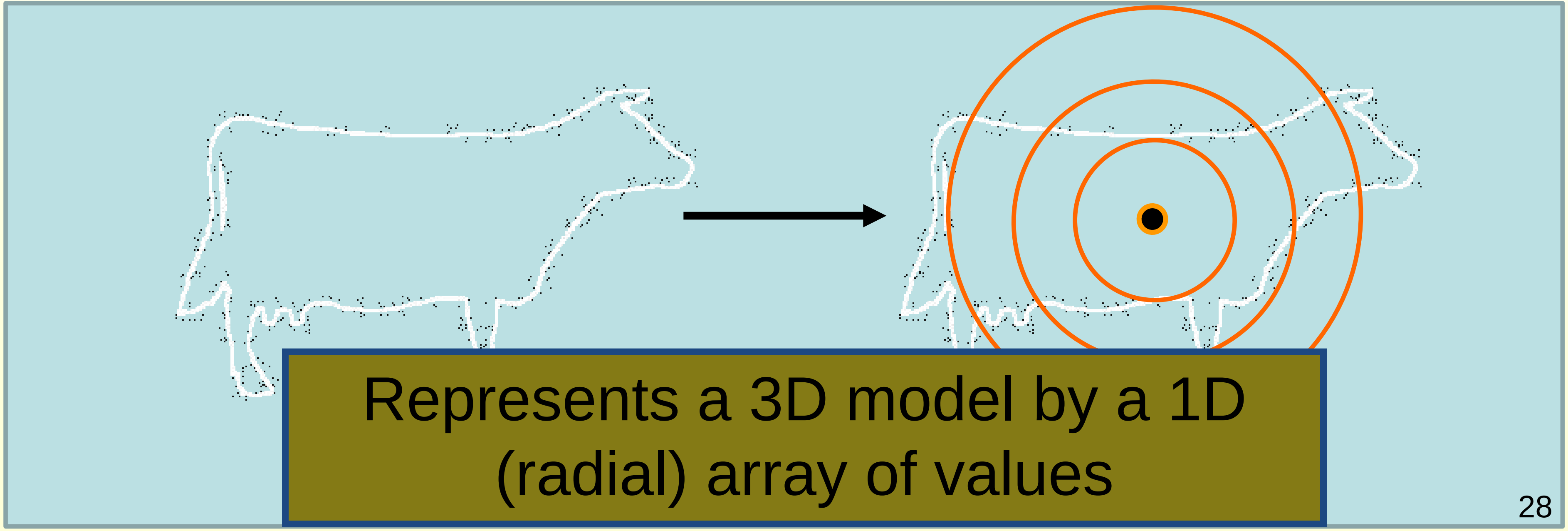


Shape Descriptors: Examples

Shape Histograms

[Ankerst *et al*. 1999]

Shape descriptor stores a histogram of how much surface area resides within different concentric shells in space



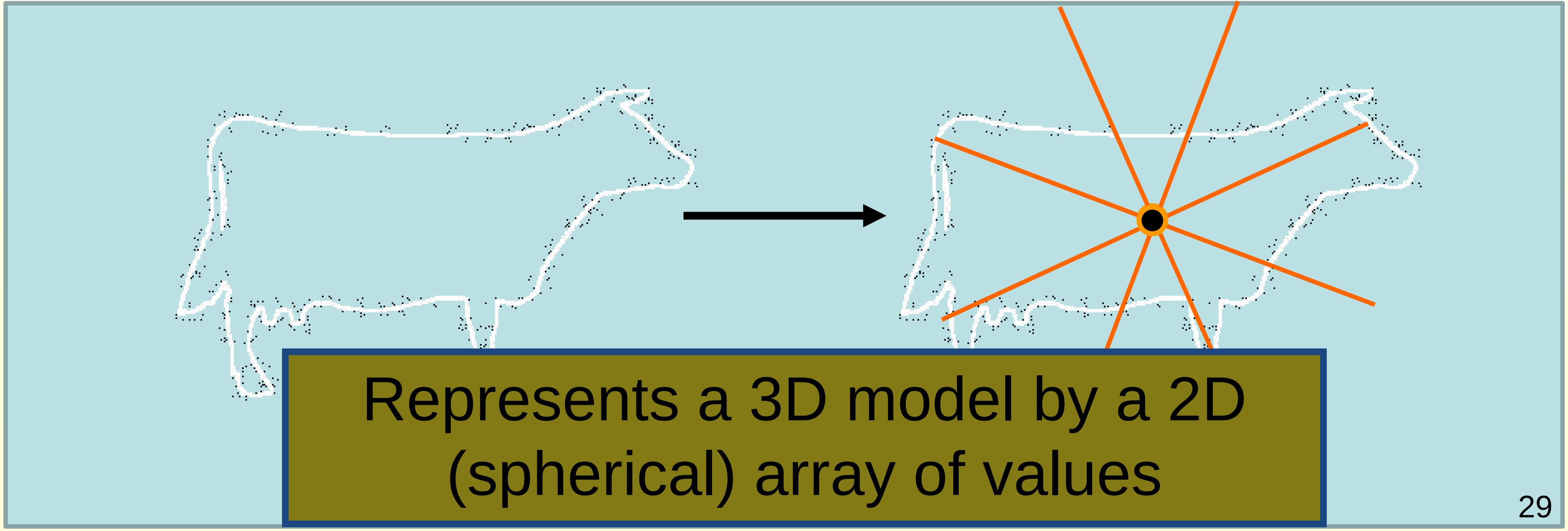
Represents a 3D model by a 1D (radial) array of values

Shape Descriptors: Examples

Shape Histograms

[Ankerst *et al*. 1999]

Shape descriptor stores a histogram of how much surface area residesm within different sectors in space



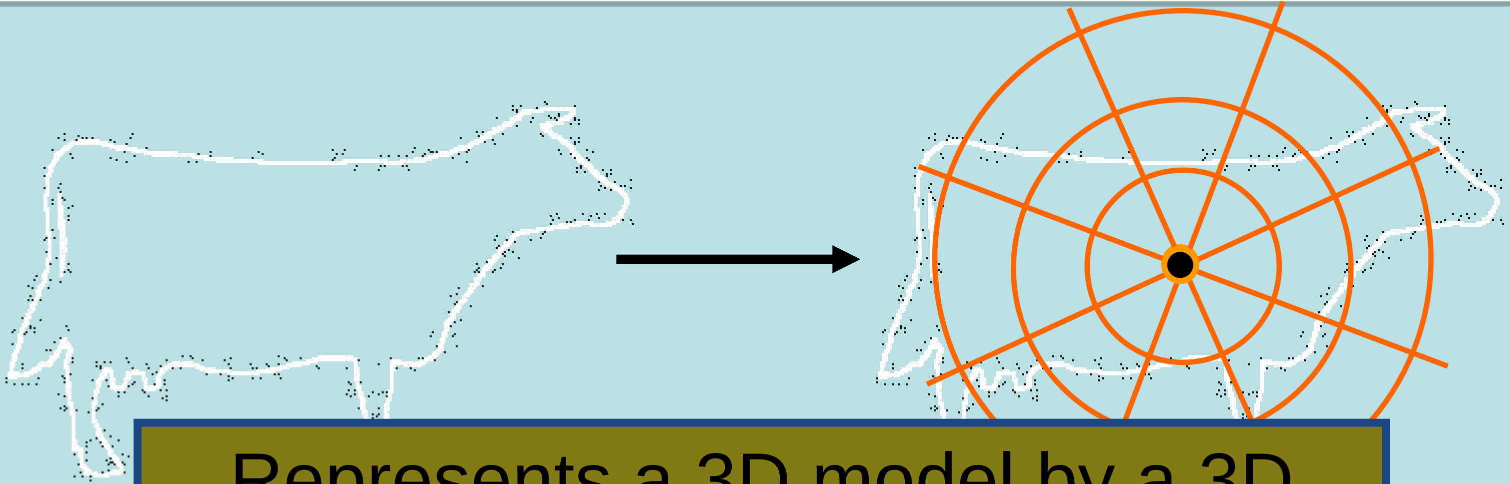
Represents a 3D model by a 2D (spherical) array of values

Shape Descriptors: Examples

Shape Histograms

[Ankerst *et al*. 1999]

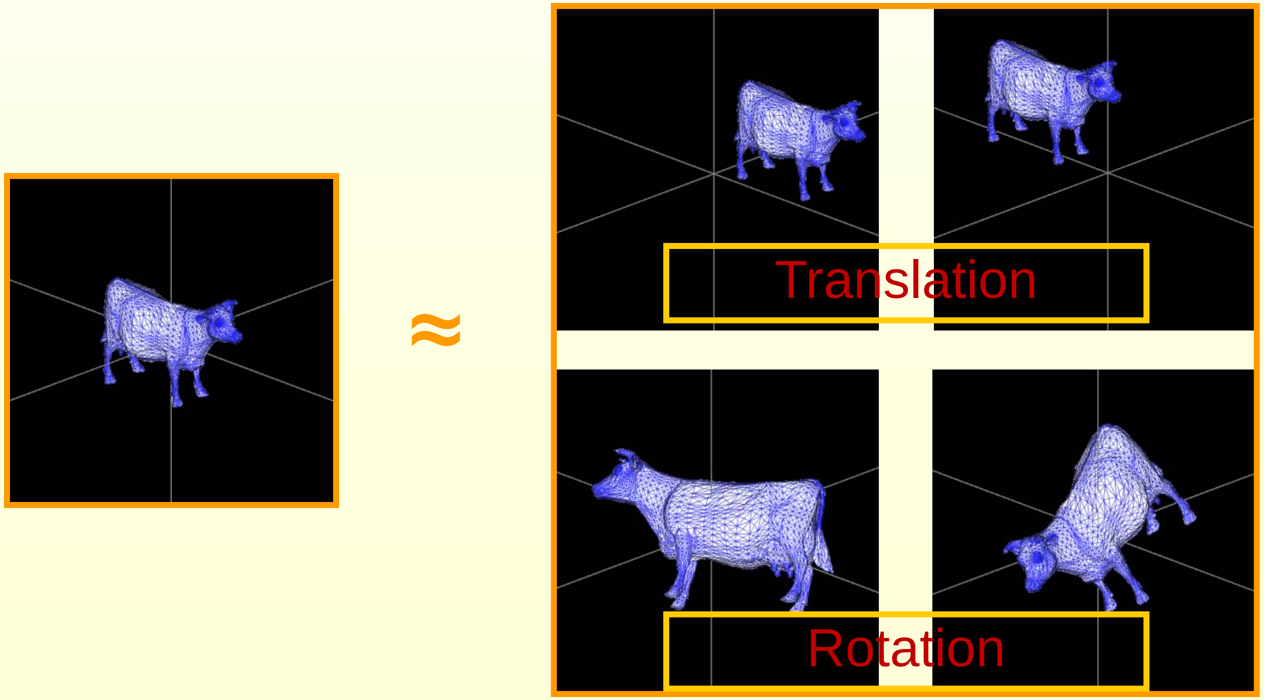
Shape descriptor stores a histogram of how much surface area resides within different shells and sectors in space



Represents a 3D model by a 3D (spherical x radial) array of values

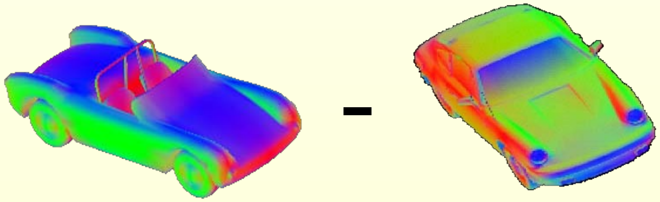
Shape Descriptors: Challenge

The descriptor must not change when a rigid body transformation is applied to the model



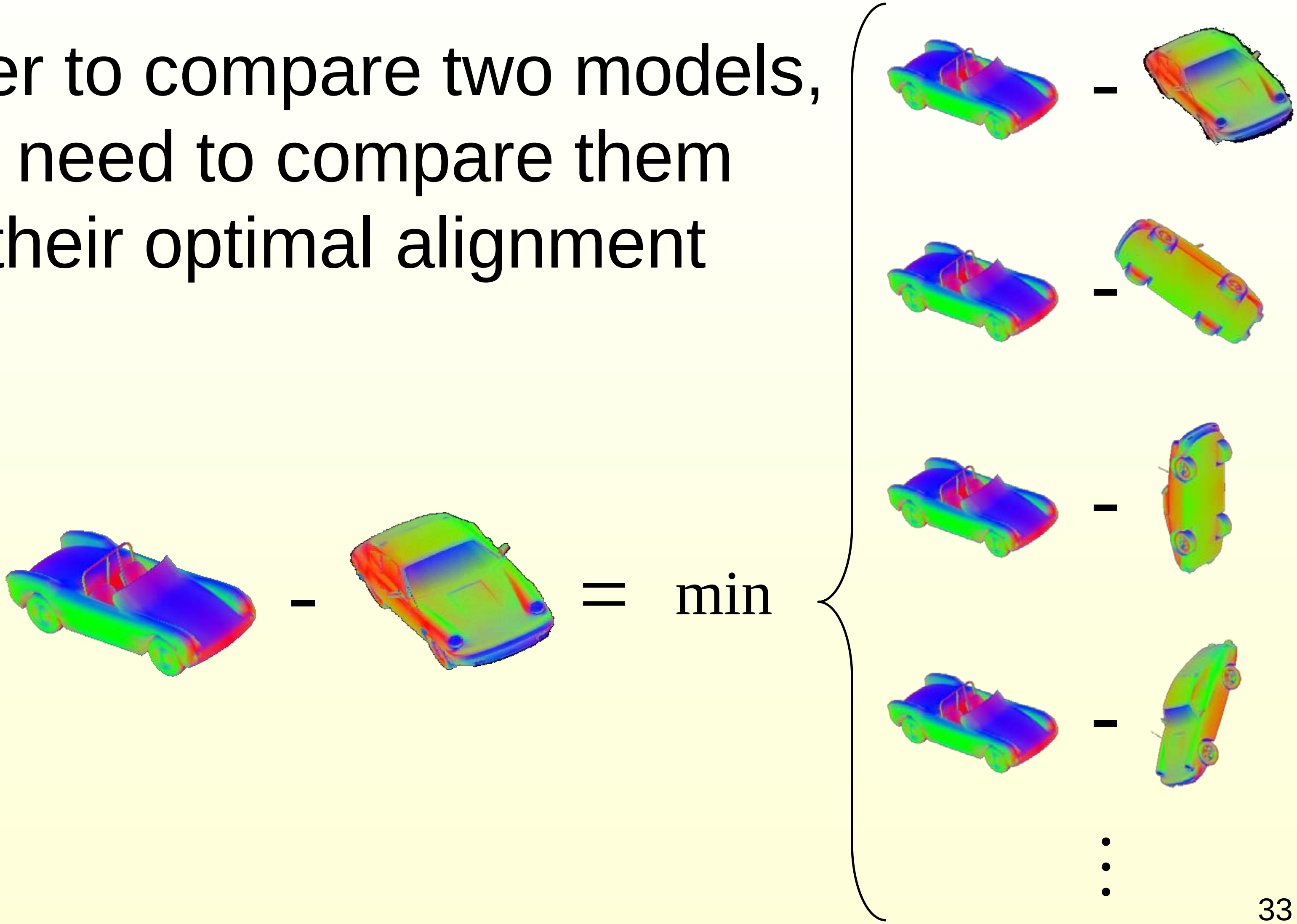
Shape Descriptors: Challenge

In order to compare two 3D models,
we need to compare them
at their optimal alignment



Shape Descriptors: Challenge

In order to compare two models,
we need to compare them
at their optimal alignment



Shape Descriptors: Alignment

Three general methods:

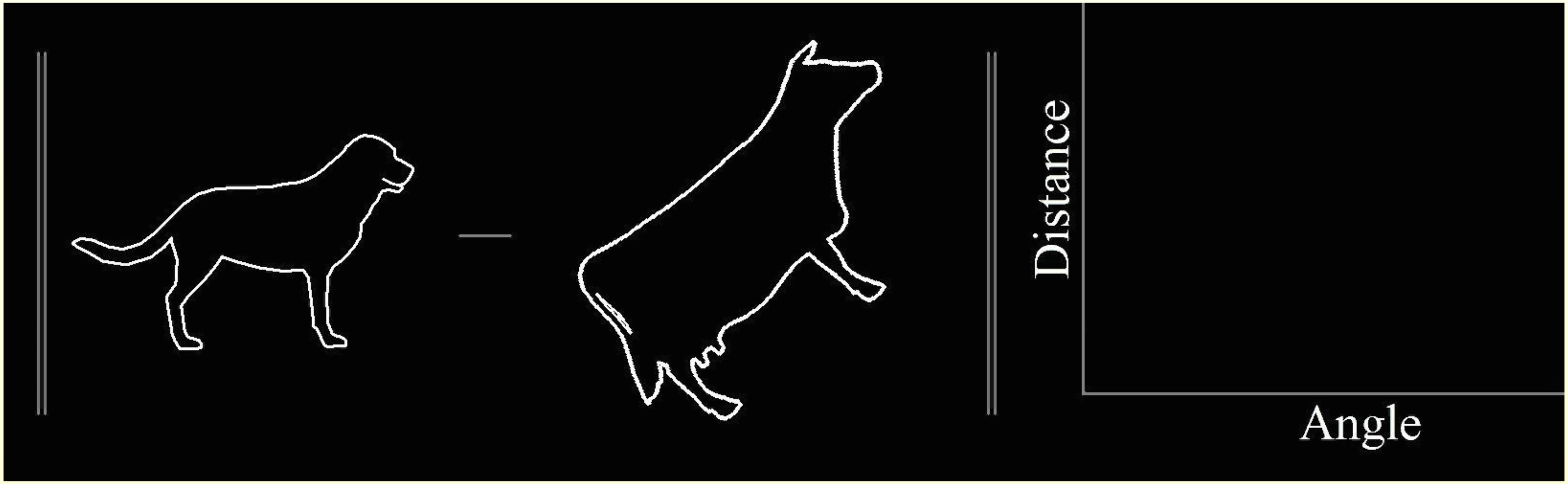
- Exhaustive Search
- Normalization
- Invariance

Shape Descriptors: Alignment

Exhaustive Search:

- Compare at all alignments

Very common in biology: e.g., protein docking

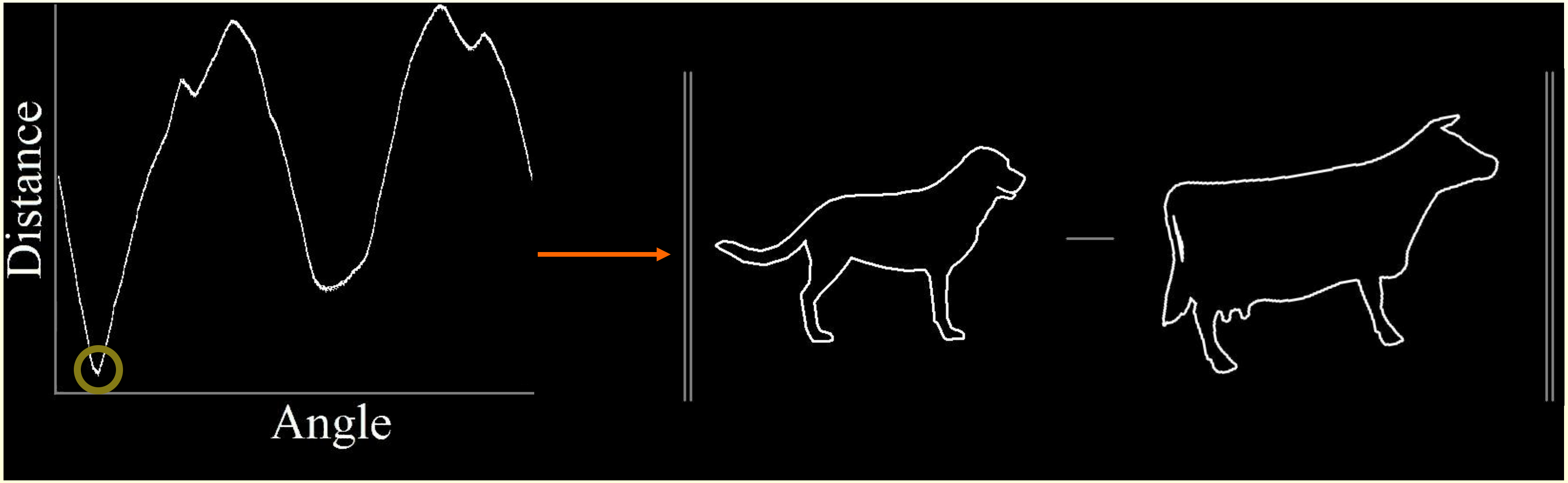


Exhaustive search for optimal rotation

Shape Descriptors: Alignment

Exhaustive Search:

- Compare at all alignments
- Correspondence is determined by the alignment at which models are closest



Exhaustive search for optimal rotation

Shape Descriptors: Alignment

Exhaustive Search:

- Compare at all alignments
- Correspondence is determined by the alignment at which models are closest

Properties:

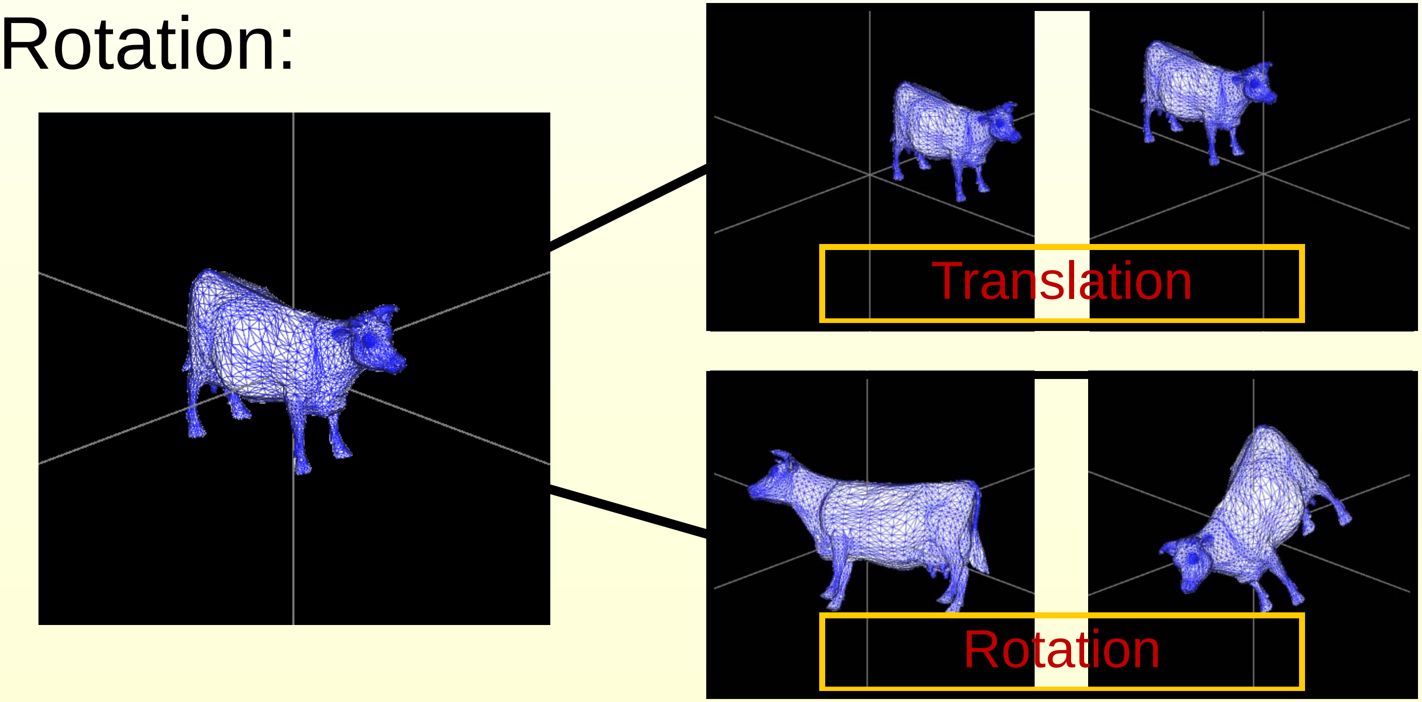
- Gives the correct answer
- Even with fast signal processing tools, it is hard to do efficiently

Shape Descriptors: Alignment

Normalization:

Put each model into a canonical frame

- Translation:
- Rotation:

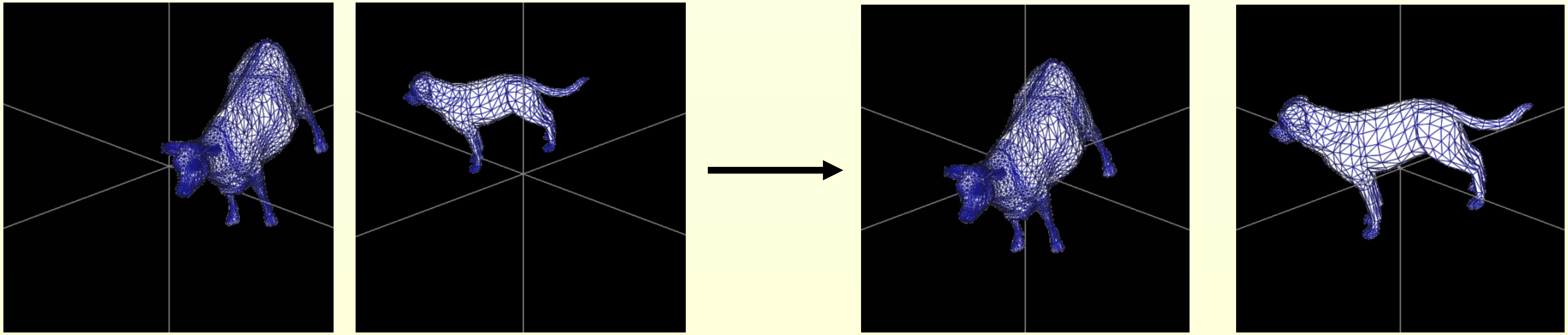


Shape Descriptors: Alignment

Normalization:

Put each model into a canonical frame

- Translation: Center of Mass
- Rotation:



Initial Models

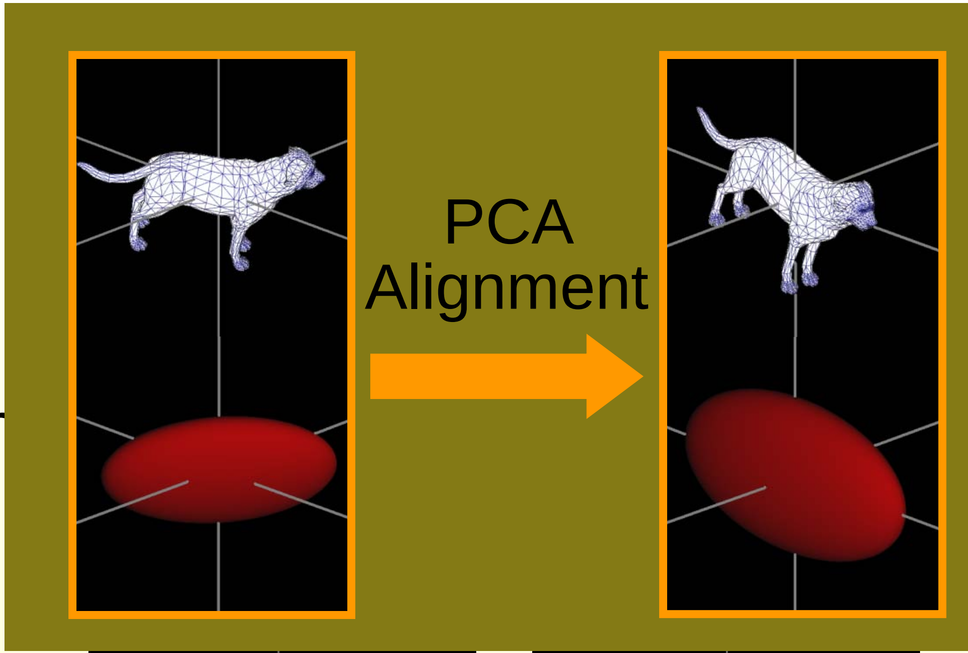
Translation-Aligned Models

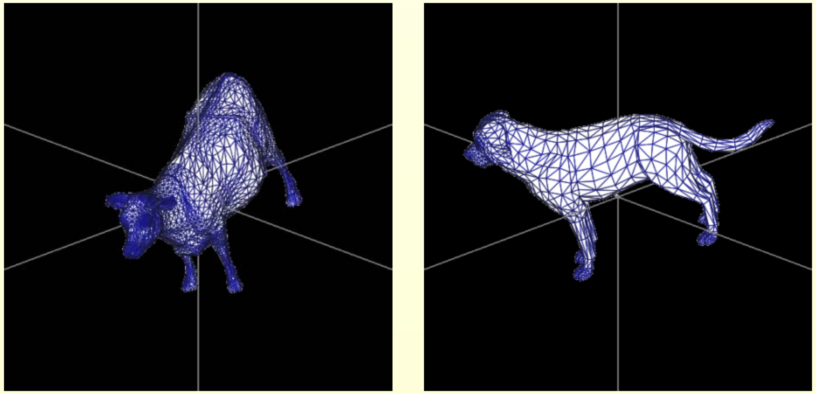
Shape Descriptors: Alignment

Normalization:

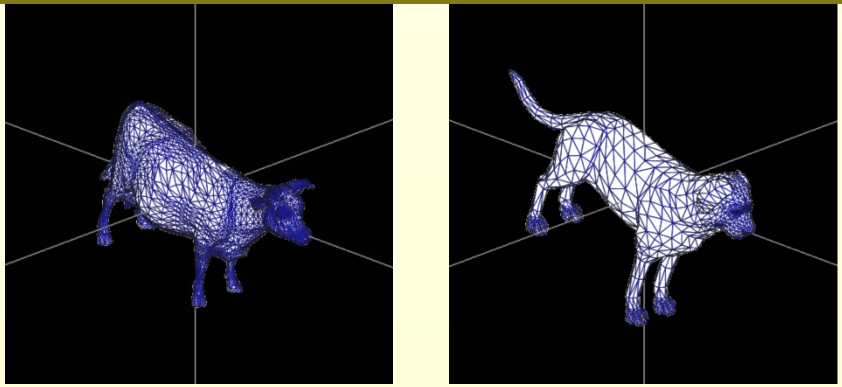
Put each model into

- Translation: Center
- Rotation: PCA Align





Translation-Aligned Models



Fully Aligned Models

Shape Descriptors: Alignment

<u>Normalization</u>:

Put each model into a canonical frame

- Translation: Center of Mass
- Rotation: PCA Alignment

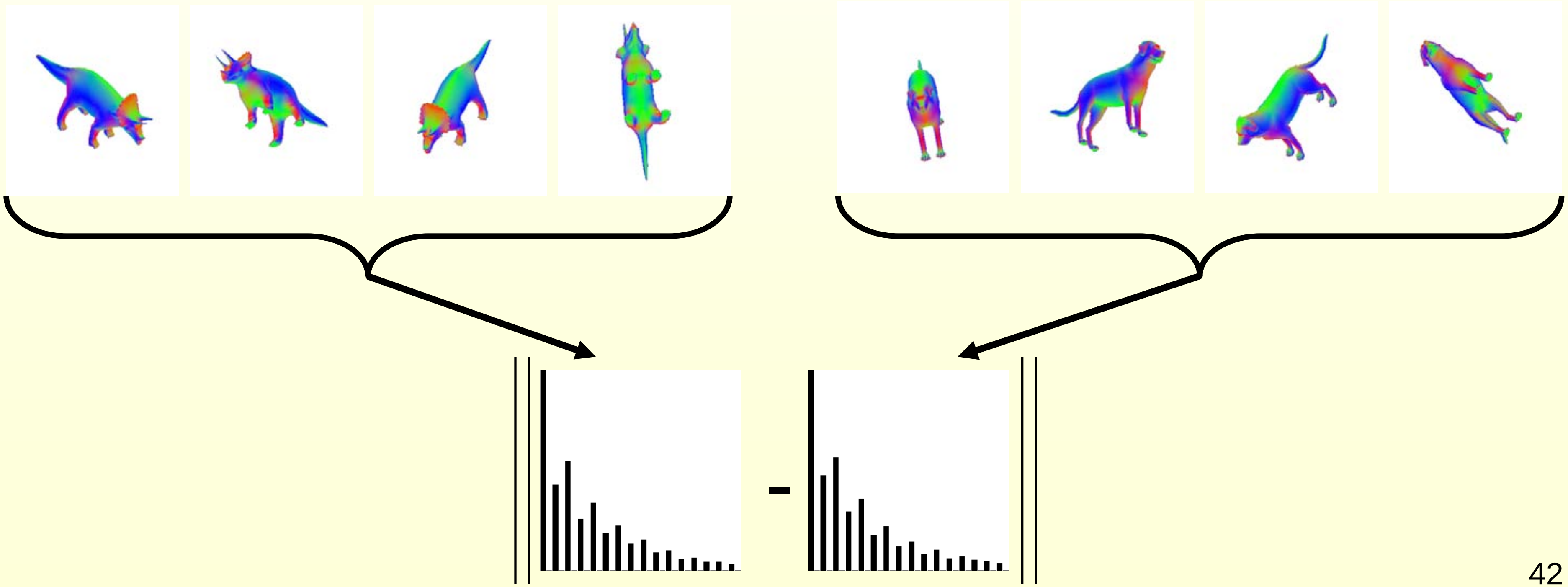
<u>Properties</u>:

- Efficient
- Not always robust – sensitive to missing data
- Cannot be used for feature matching

Shape Descriptors: Alignment

Invariance:

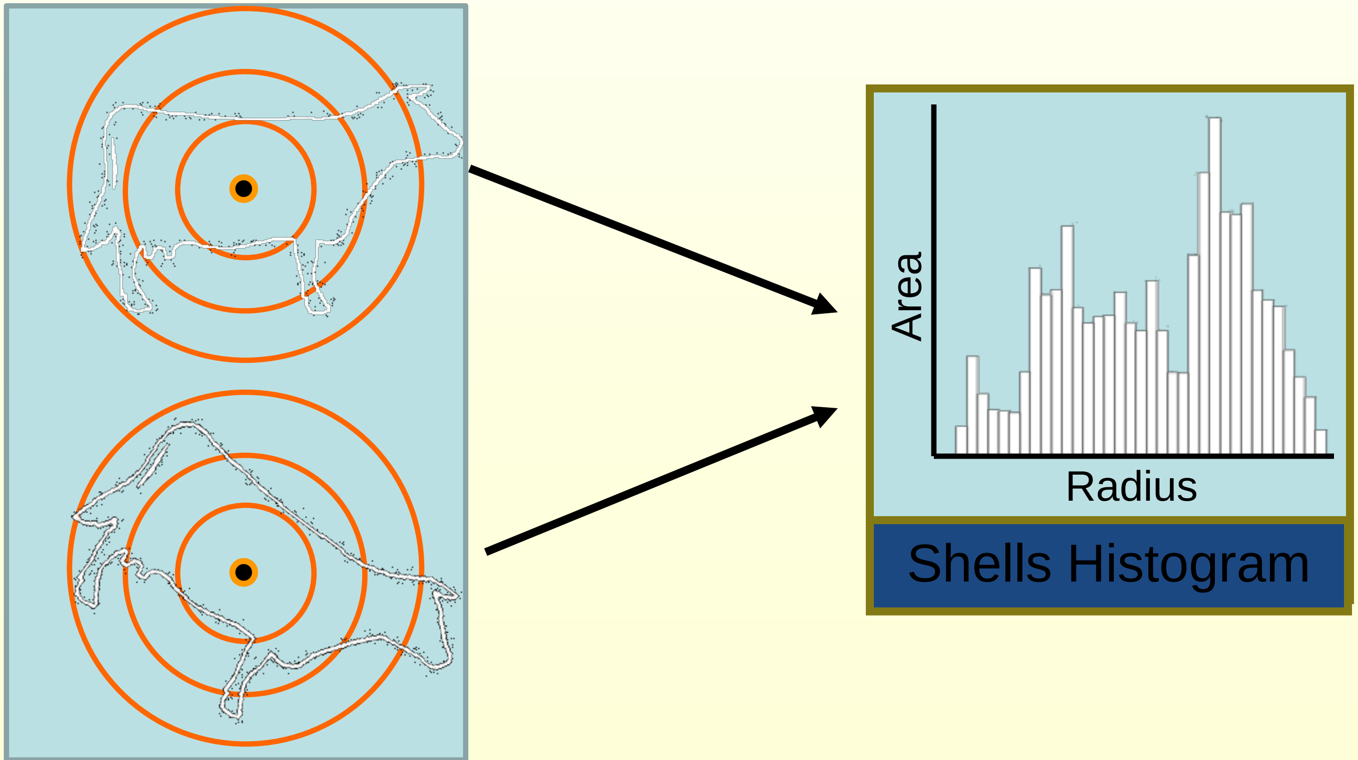
Represent a model by a shape descriptor that is independent of the pose.



Shape Descriptors: Alignment

Example: Ankerst's *Shells* [Ankerst *et al*. 1999]

A histogram of the radial distribution of surface area.

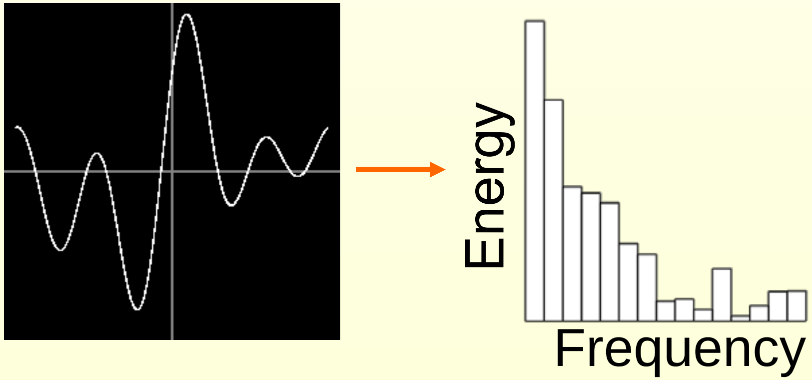


Shape Descriptors: Alignment

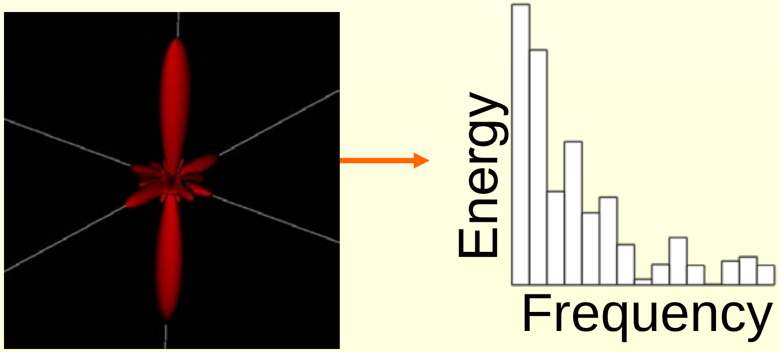
Invariance:

Power spectrum representation

- Fourier transform for translations
- Spherical harmonic transform for rotations



Circular Power Spectrum

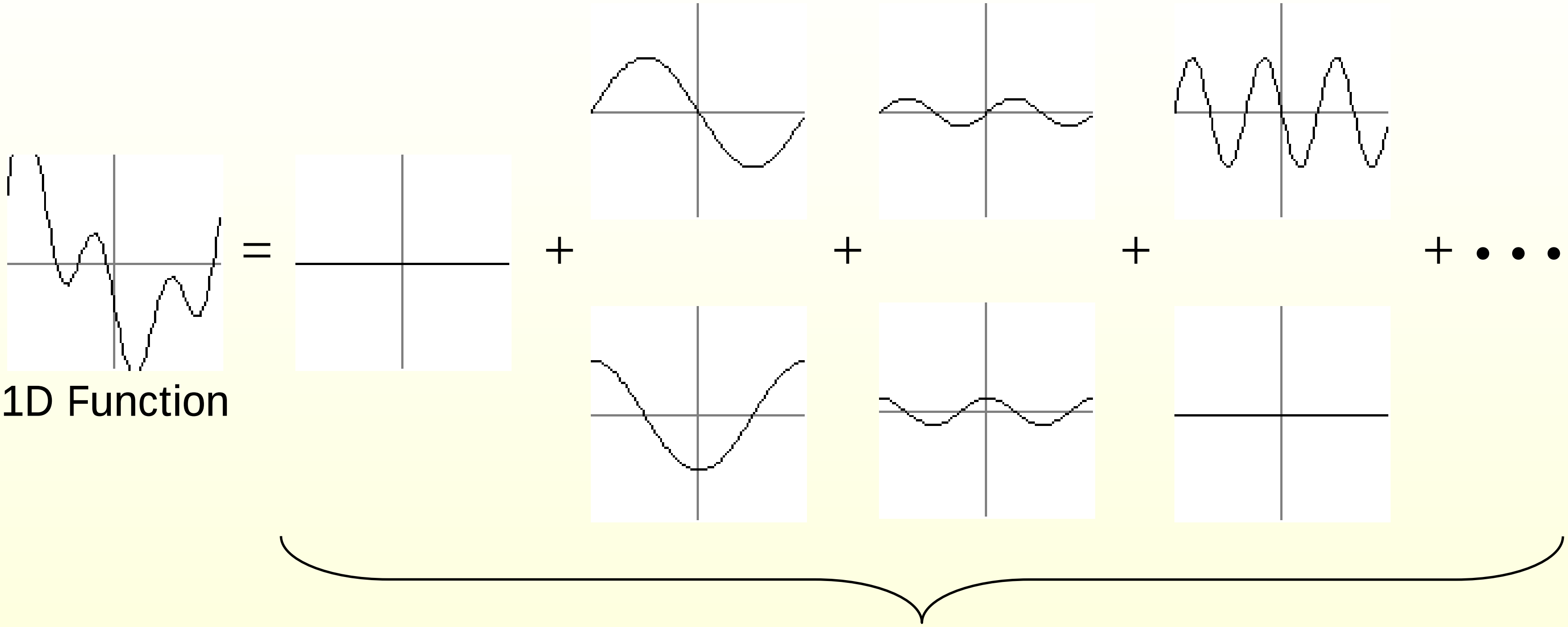


Spherical Power Spectrum

Translation Invariance

1D Function

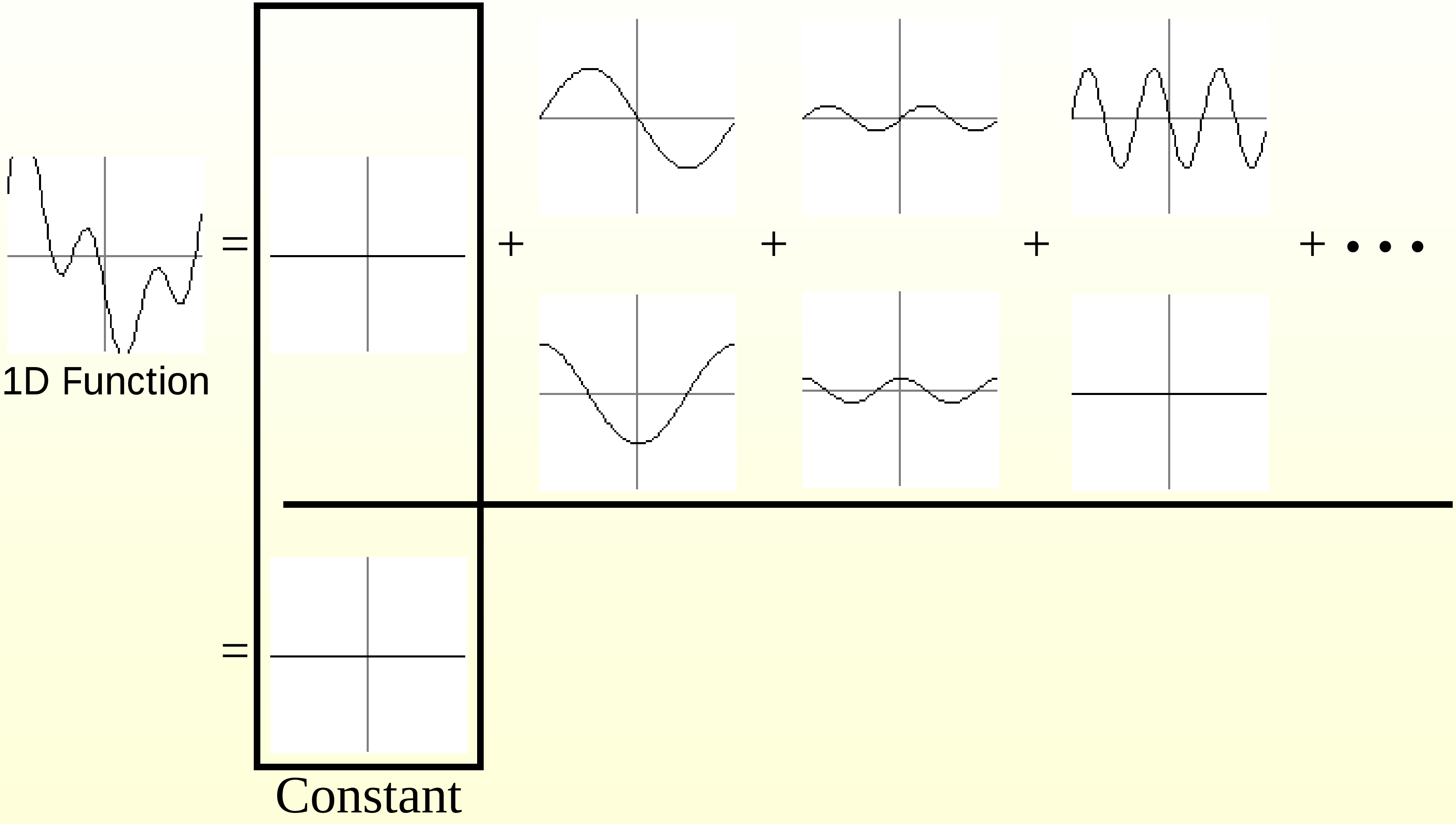
Translation Invariance



1D Function

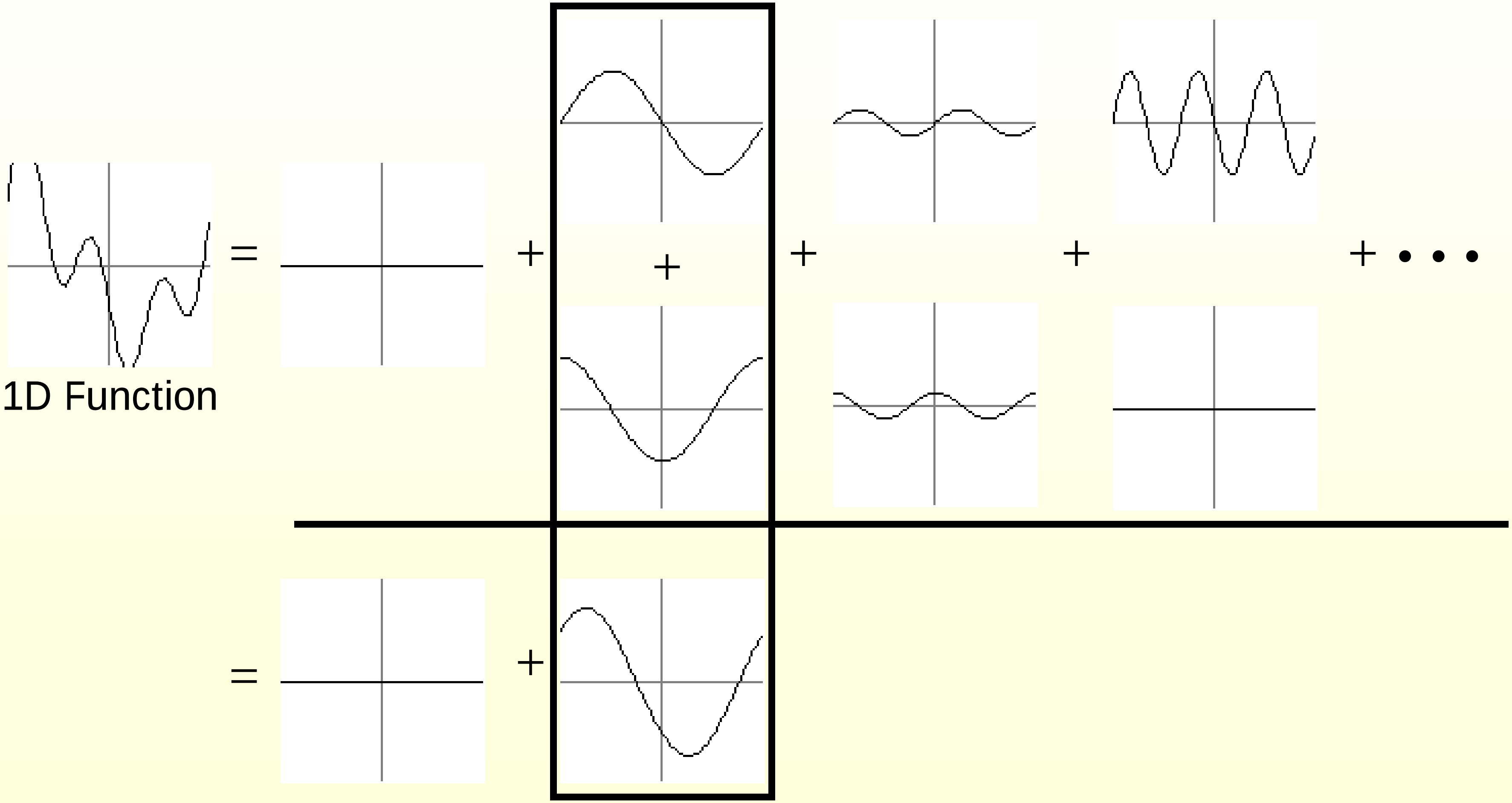
Cosine/Sine Decomposition

Translation Invariance



Frequency Decomposition

Translation Invariance



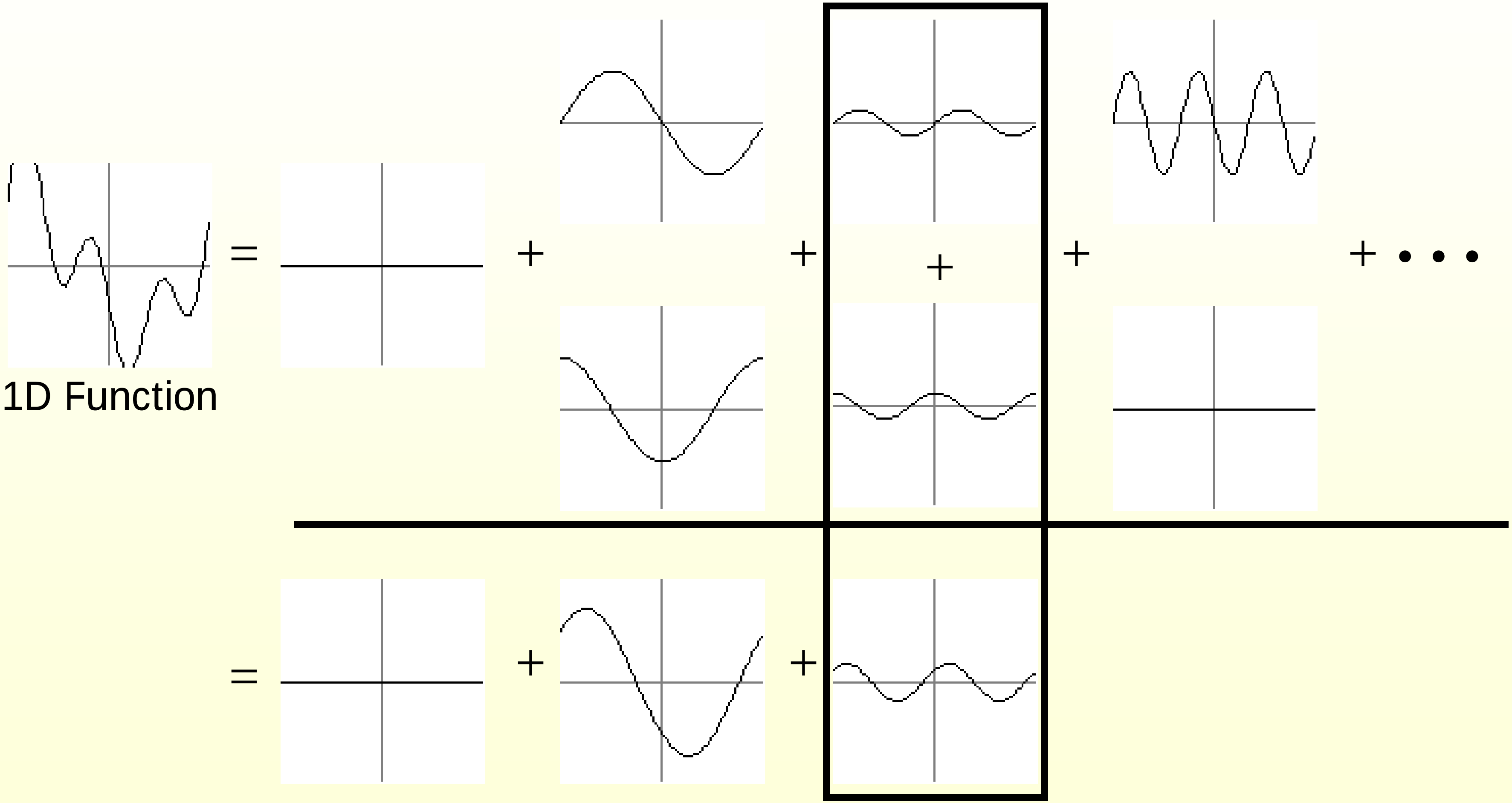
1D Function

Constant

1st Order

Frequency Decomposition

Translation Invariance



1D Function

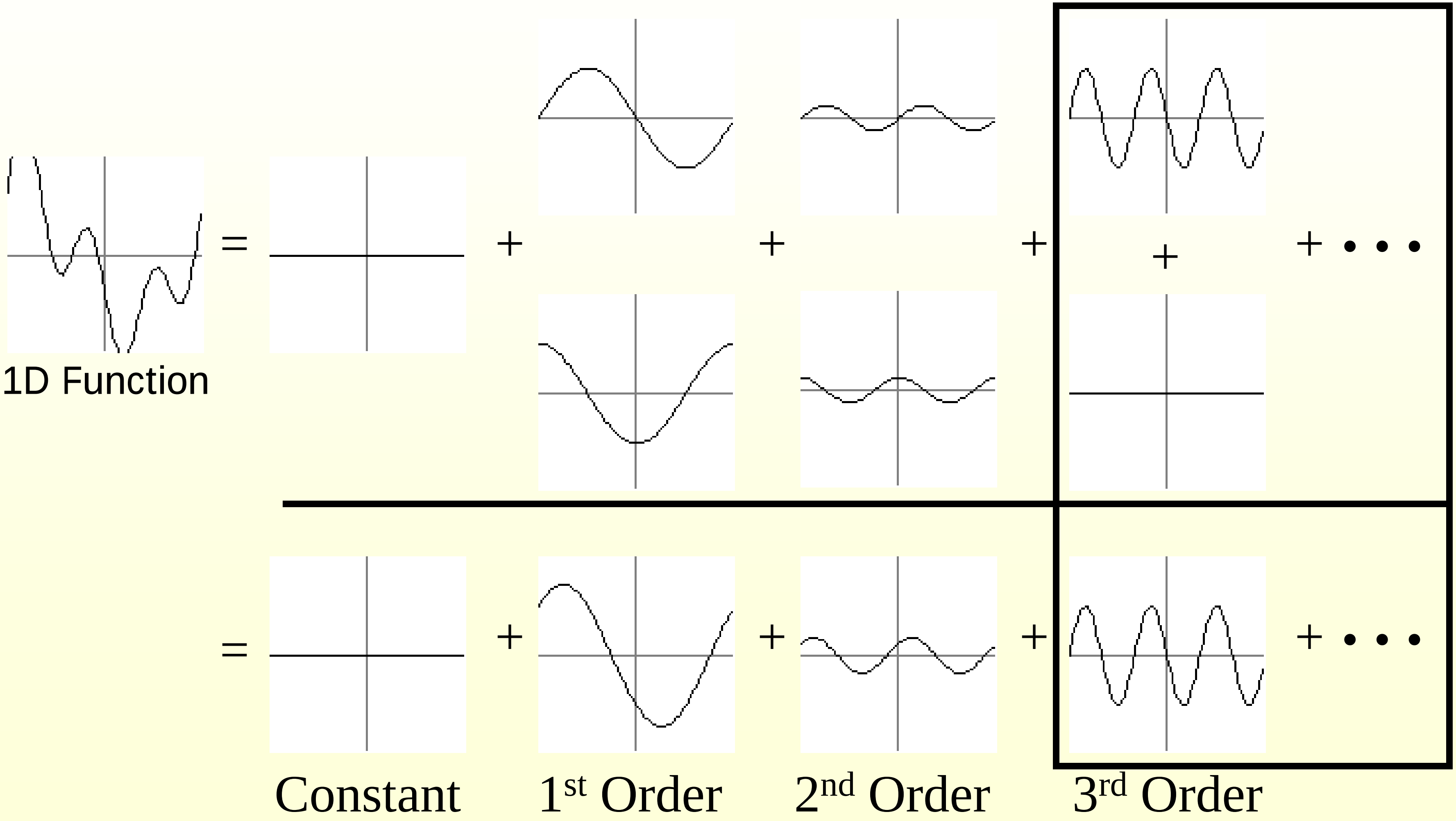
Constant

1st Order

2nd Order

Frequency Decomposition

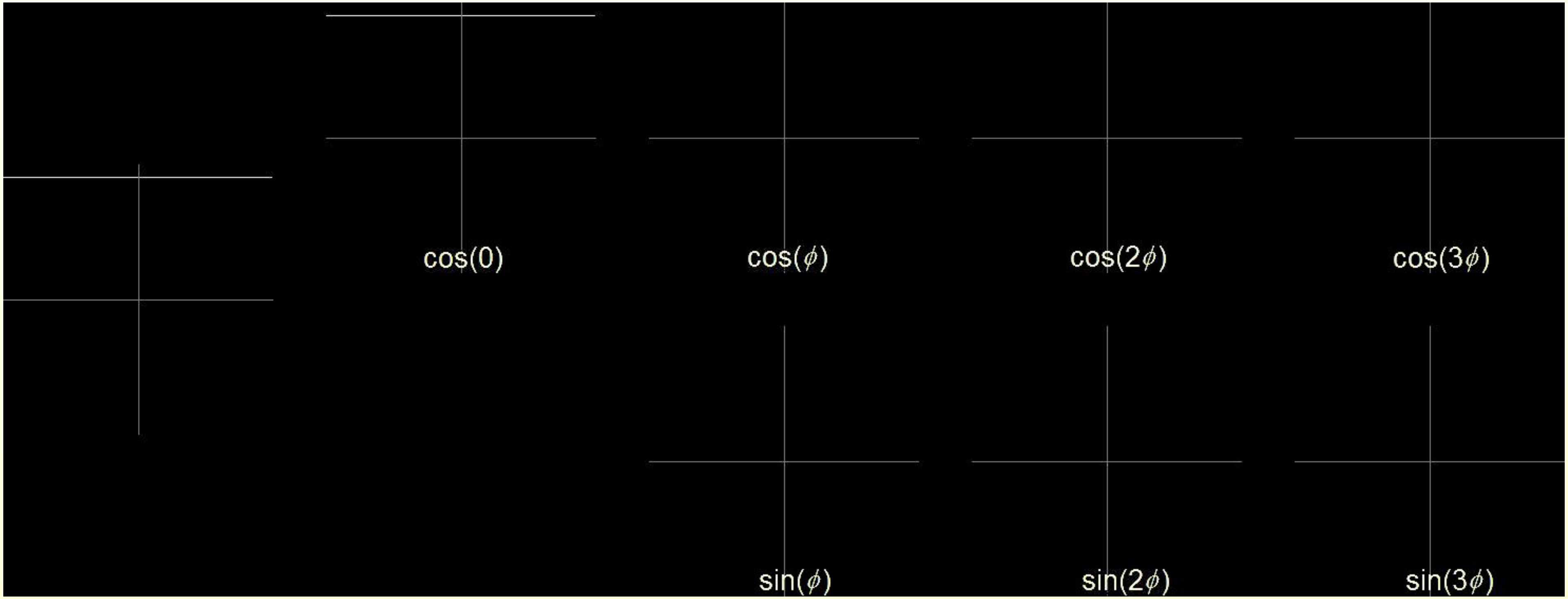
Translation Invariance



Frequency Decomposition

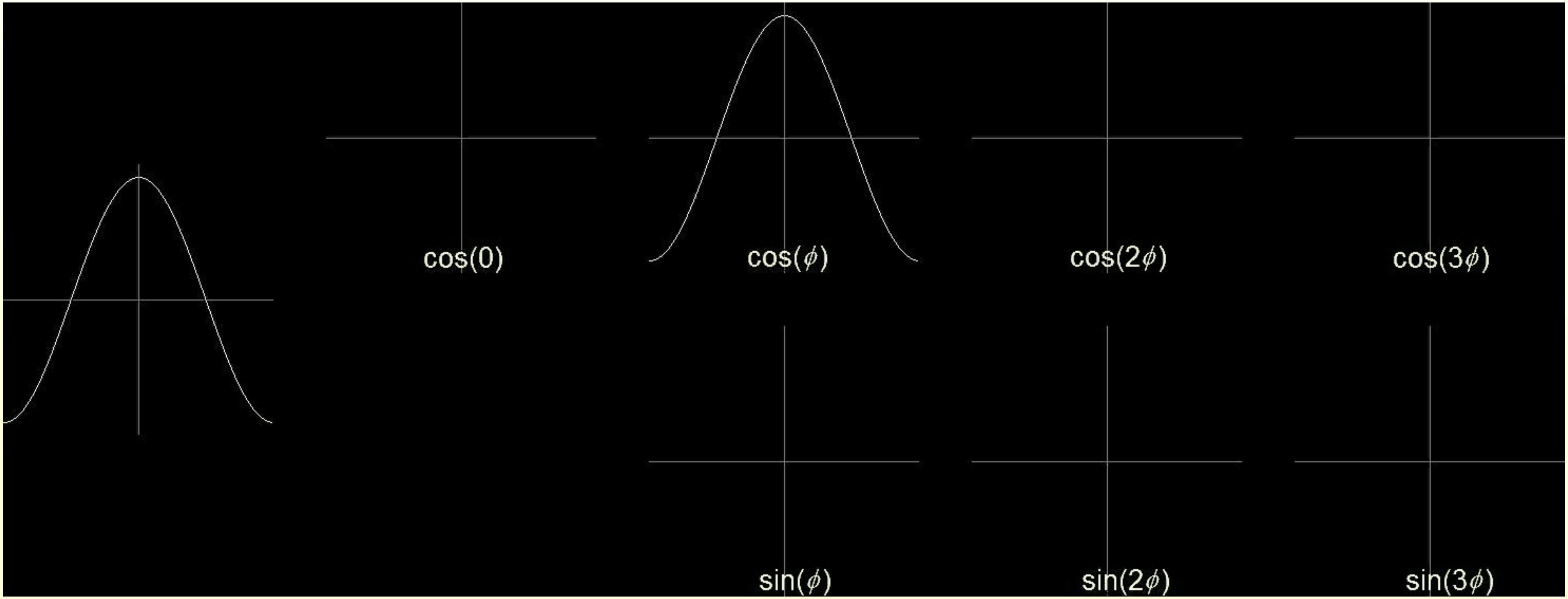
Translation Invariance

Frequency subspaces are fixed by translations:



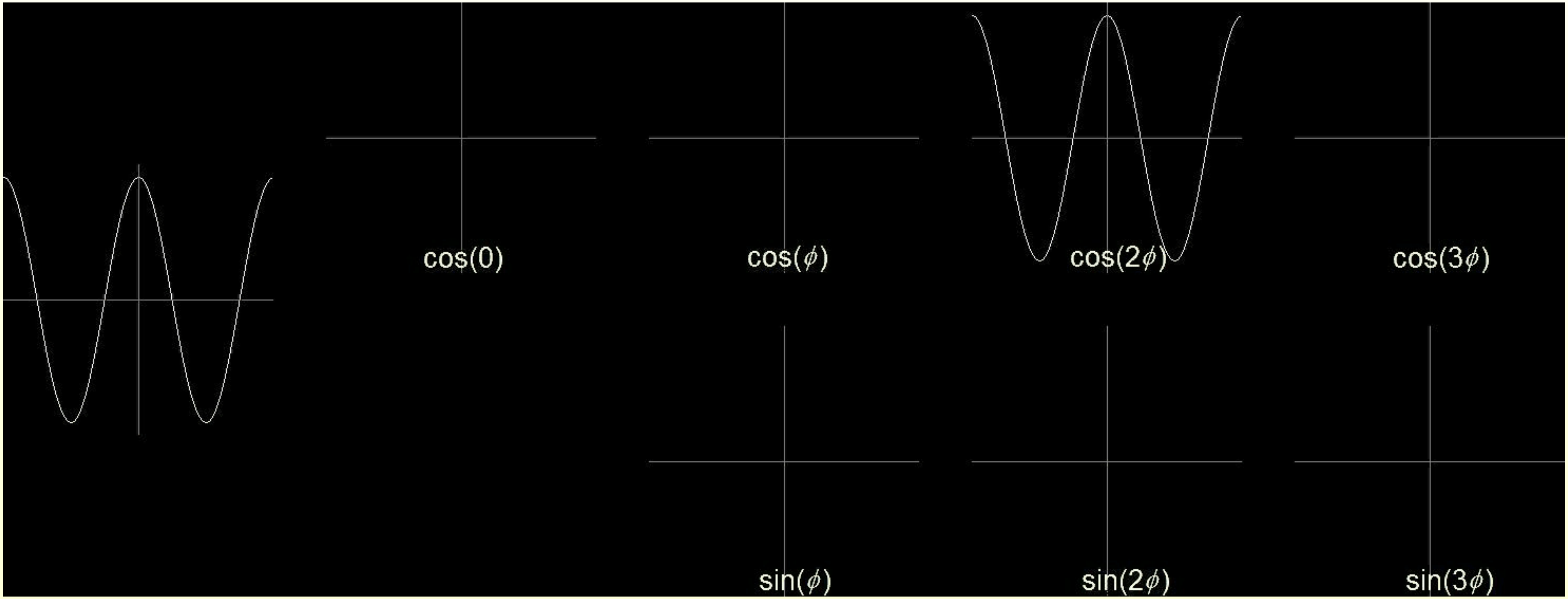
Translation Invariance

Frequency subspaces are fixed by translation:



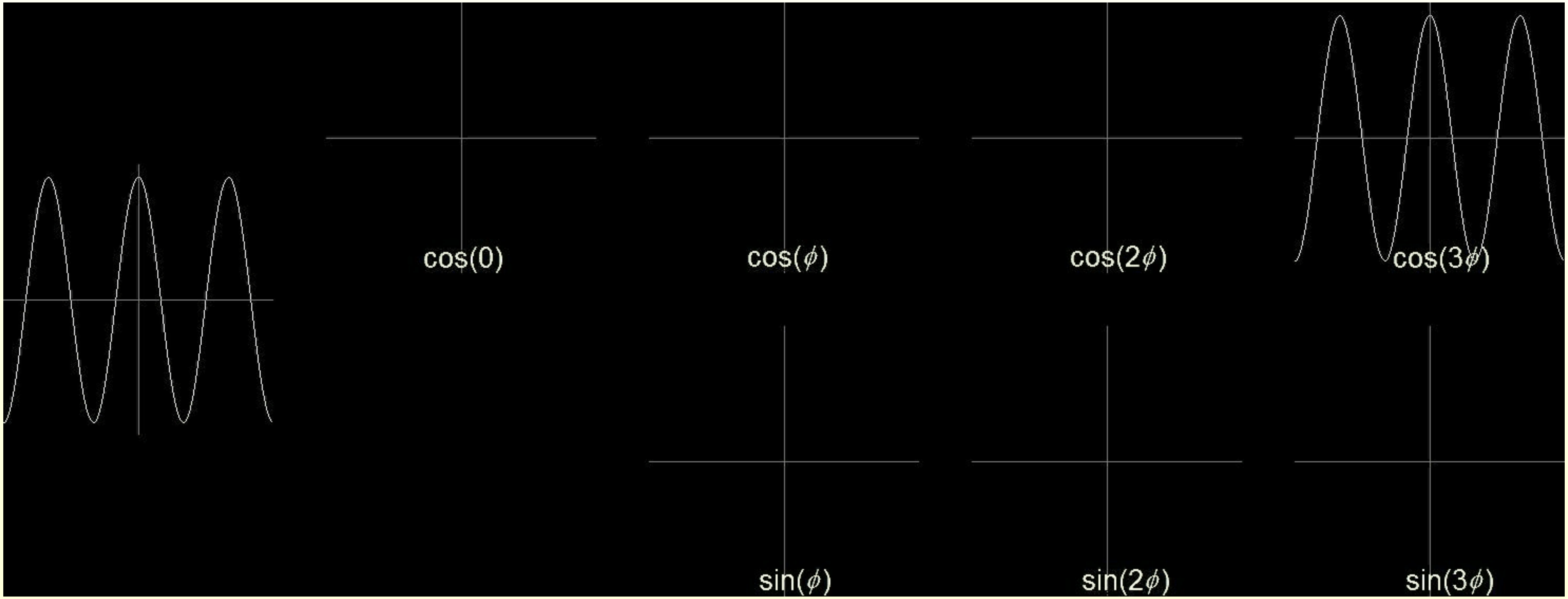
Translation Invariance

Frequency subspaces are fixed by translation:

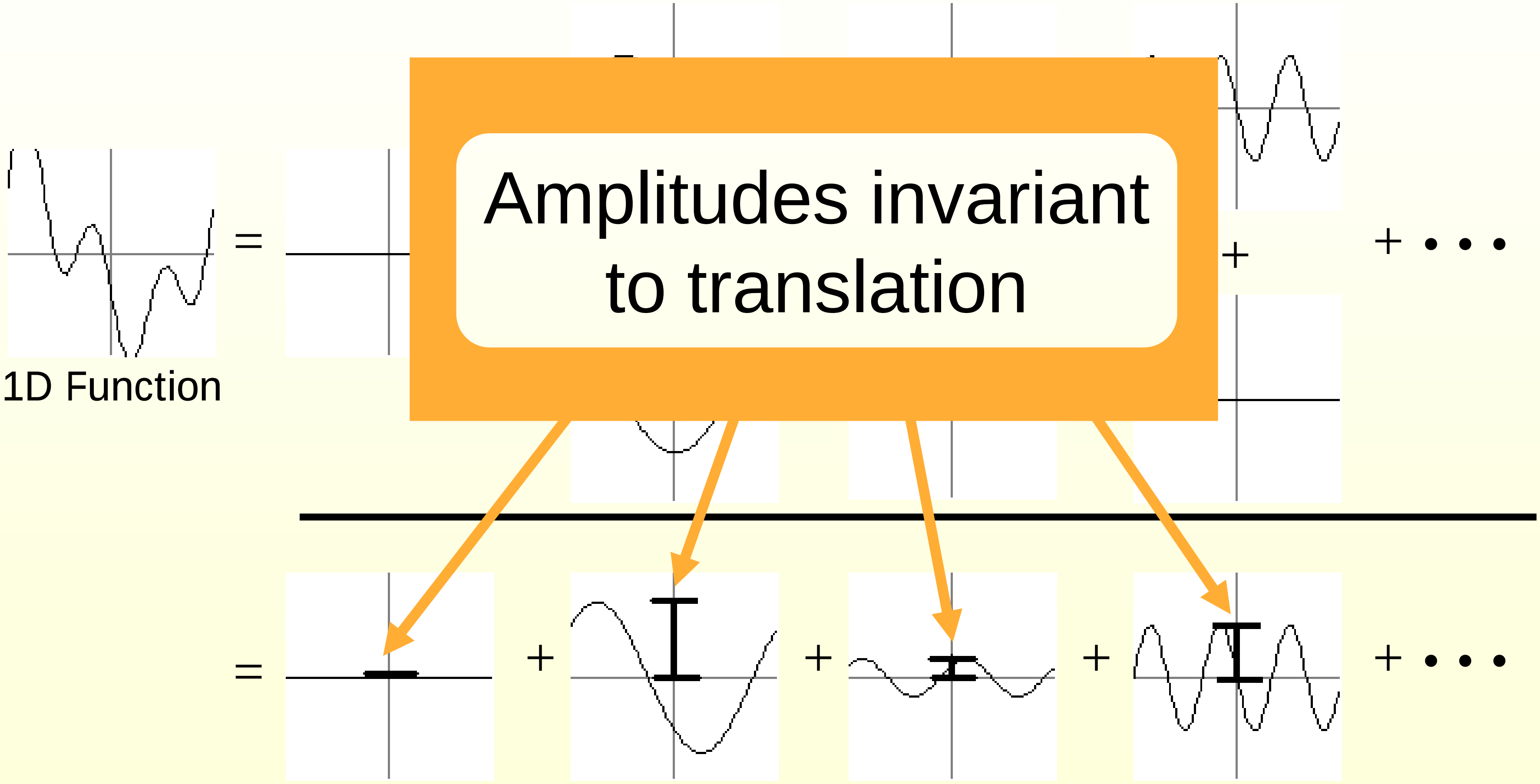


Translation Invariance

Frequency subspaces are fixed by translation:



Translation Invariance



Amplitudes invariant to translation

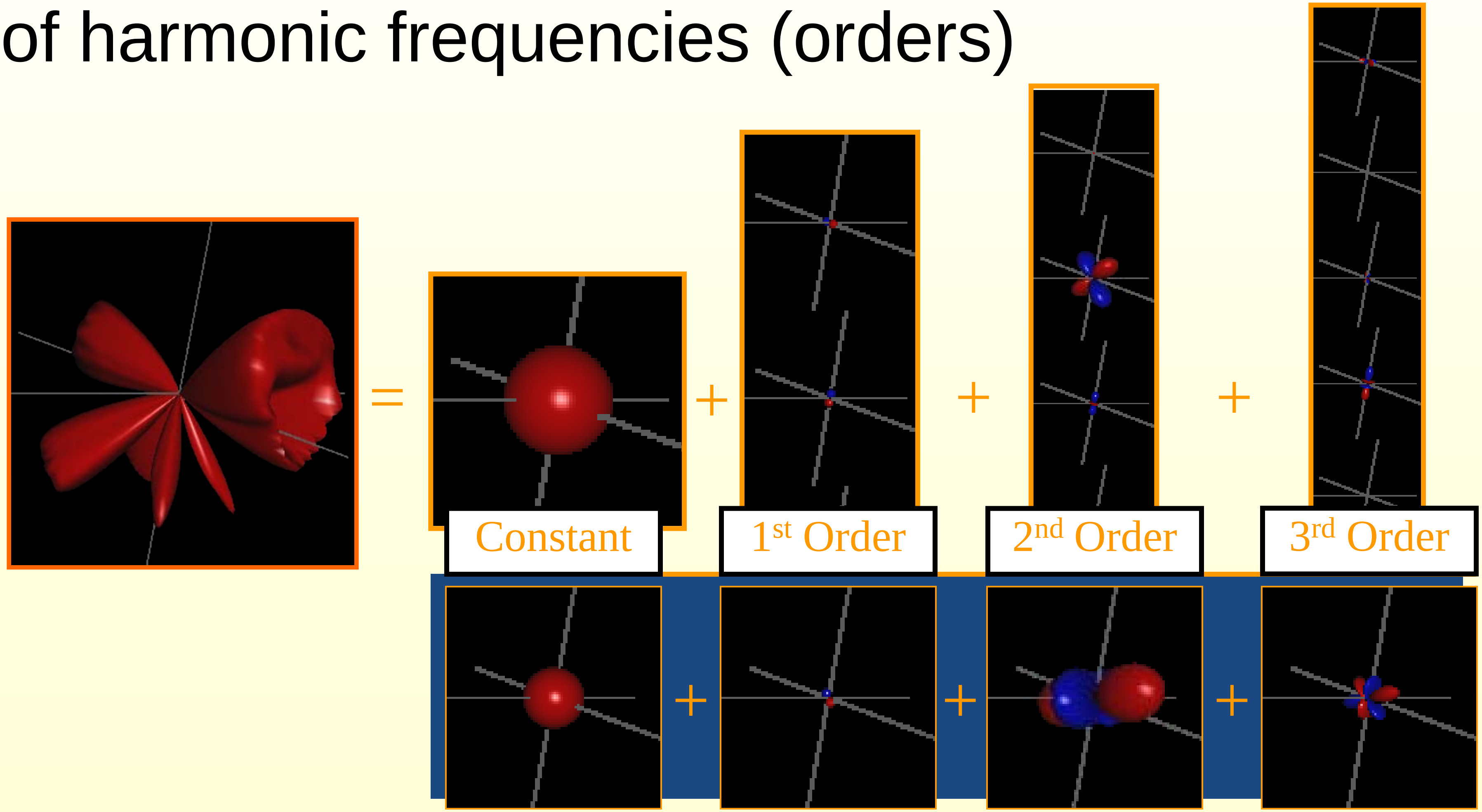
1D Function

Constant 1^{st} Order 2^{nd} Order 3^{rd} Order

Frequency Decomposition

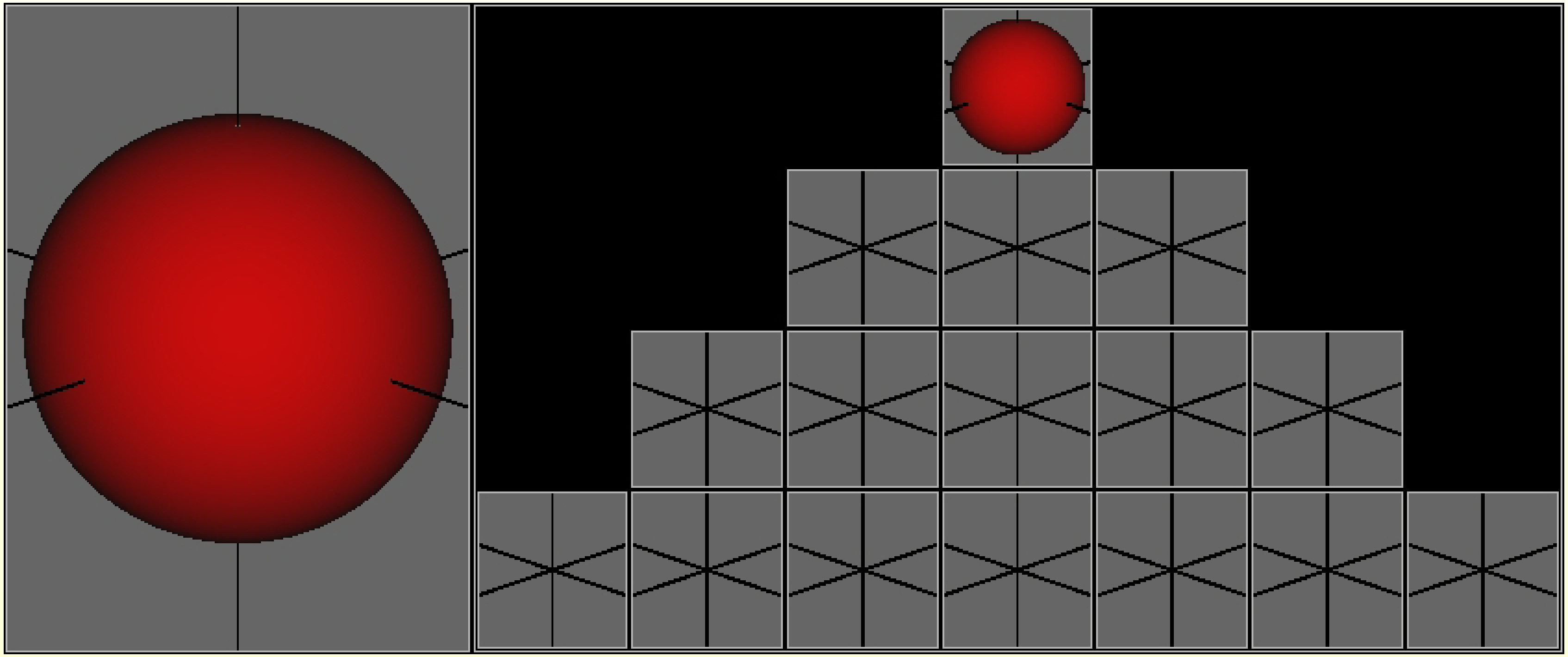
Rotation Invariance

Represent each spherical function as a sum of harmonic frequencies (orders)



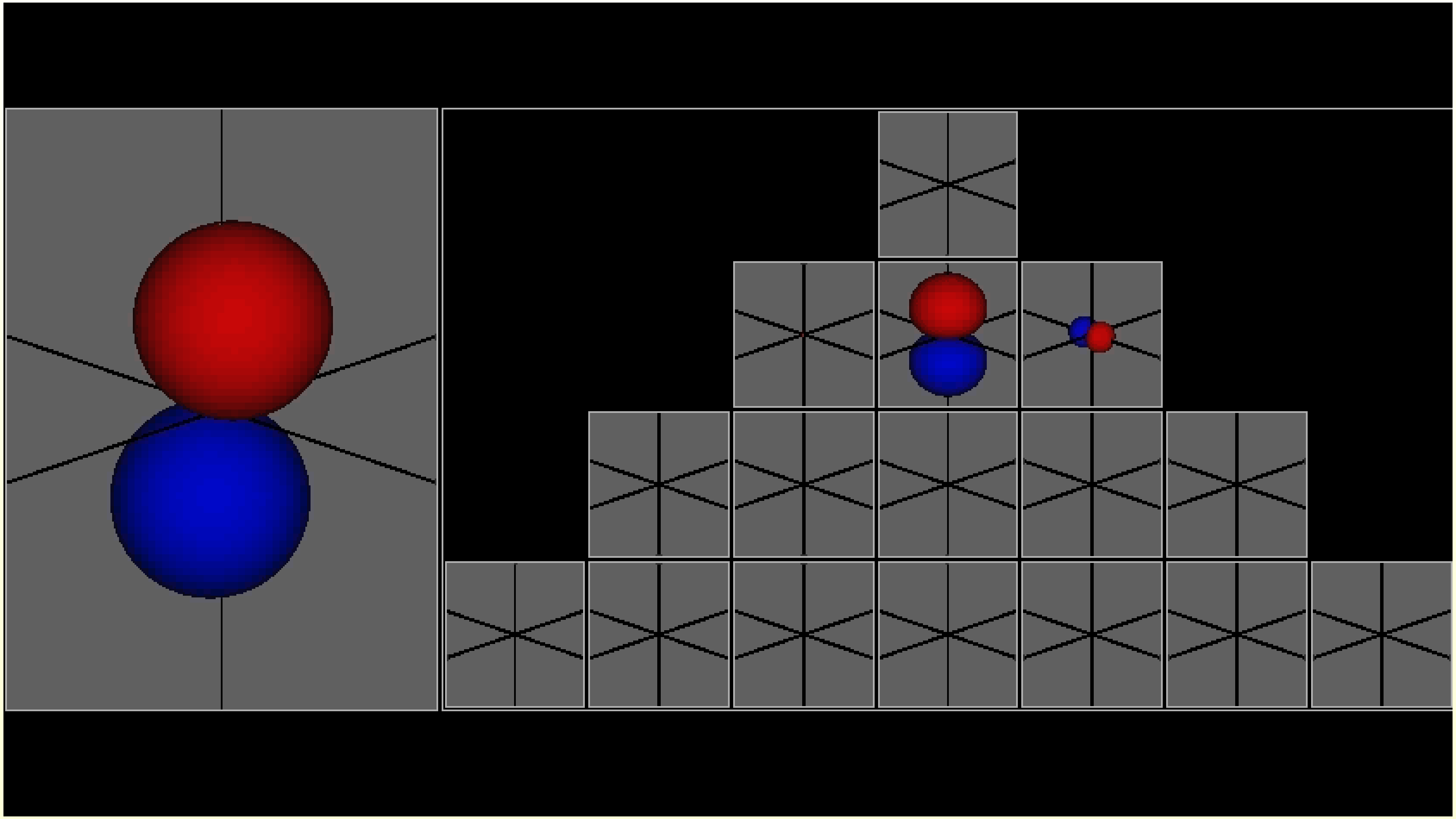
Rotation Invariance

Frequency subspaces are fixed by rotations:



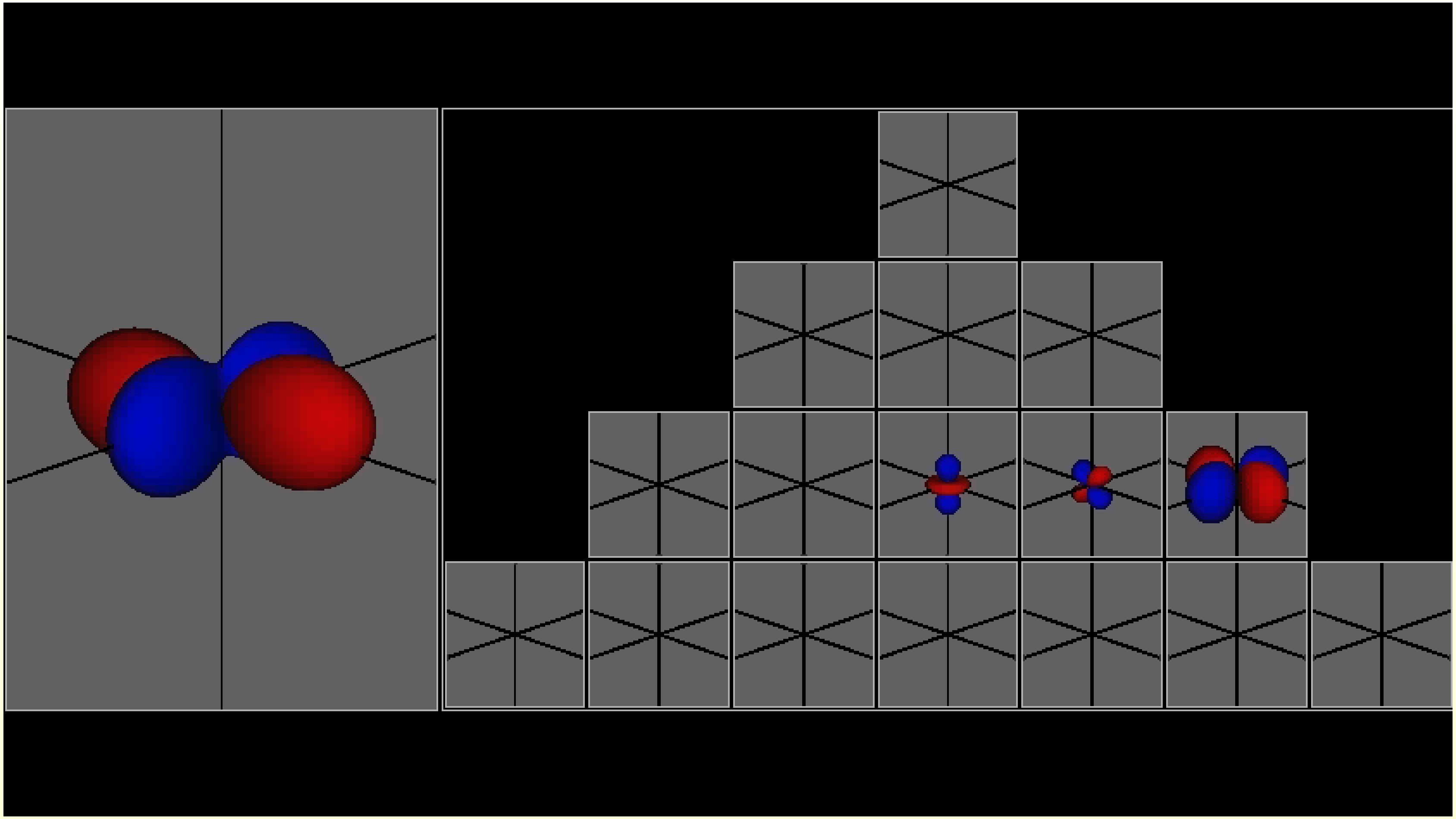
Rotation Invariance

Frequency subspaces are fixed by rotations:



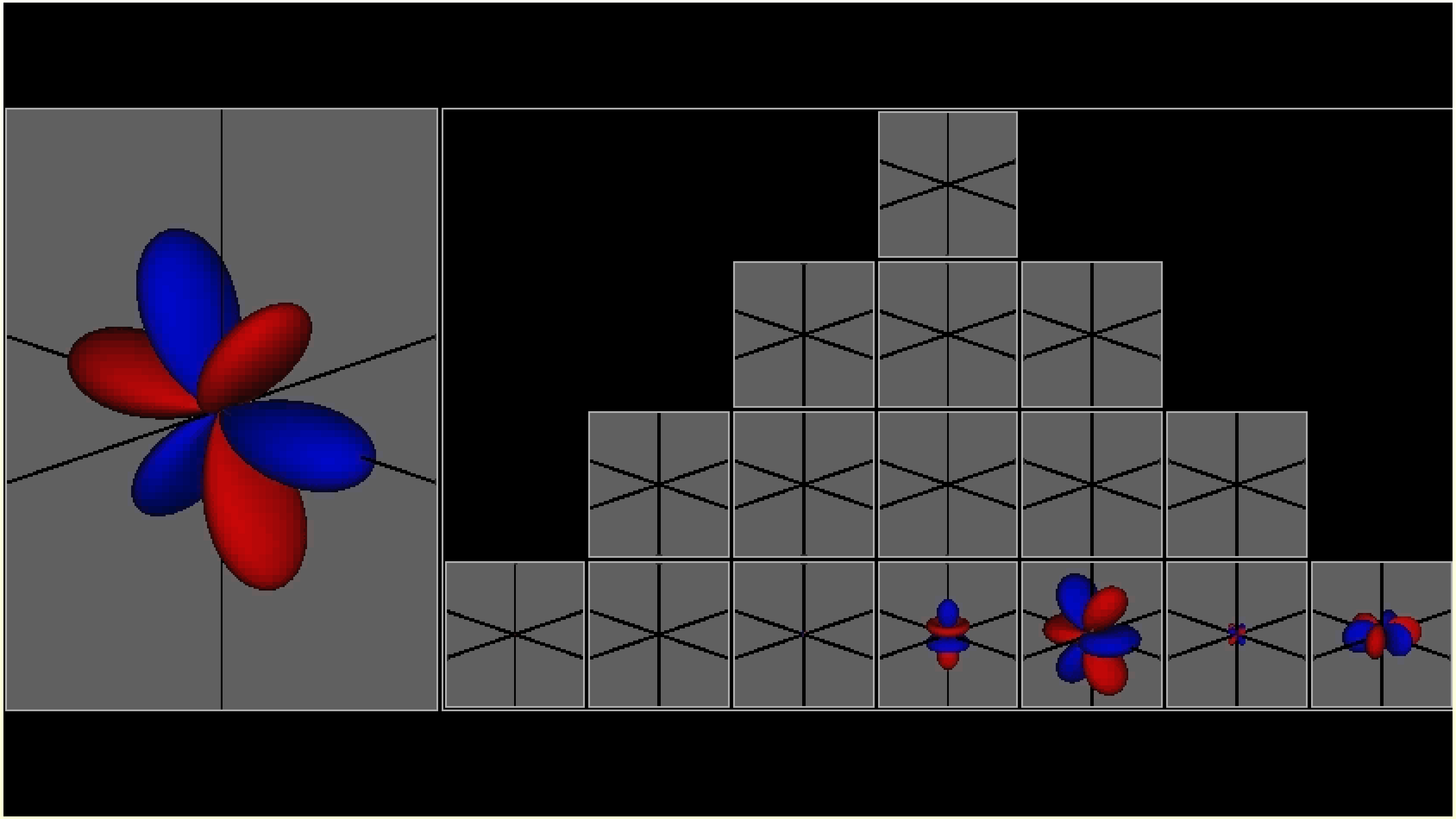
Rotation Invariance

Frequency subspaces are fixed by rotations:



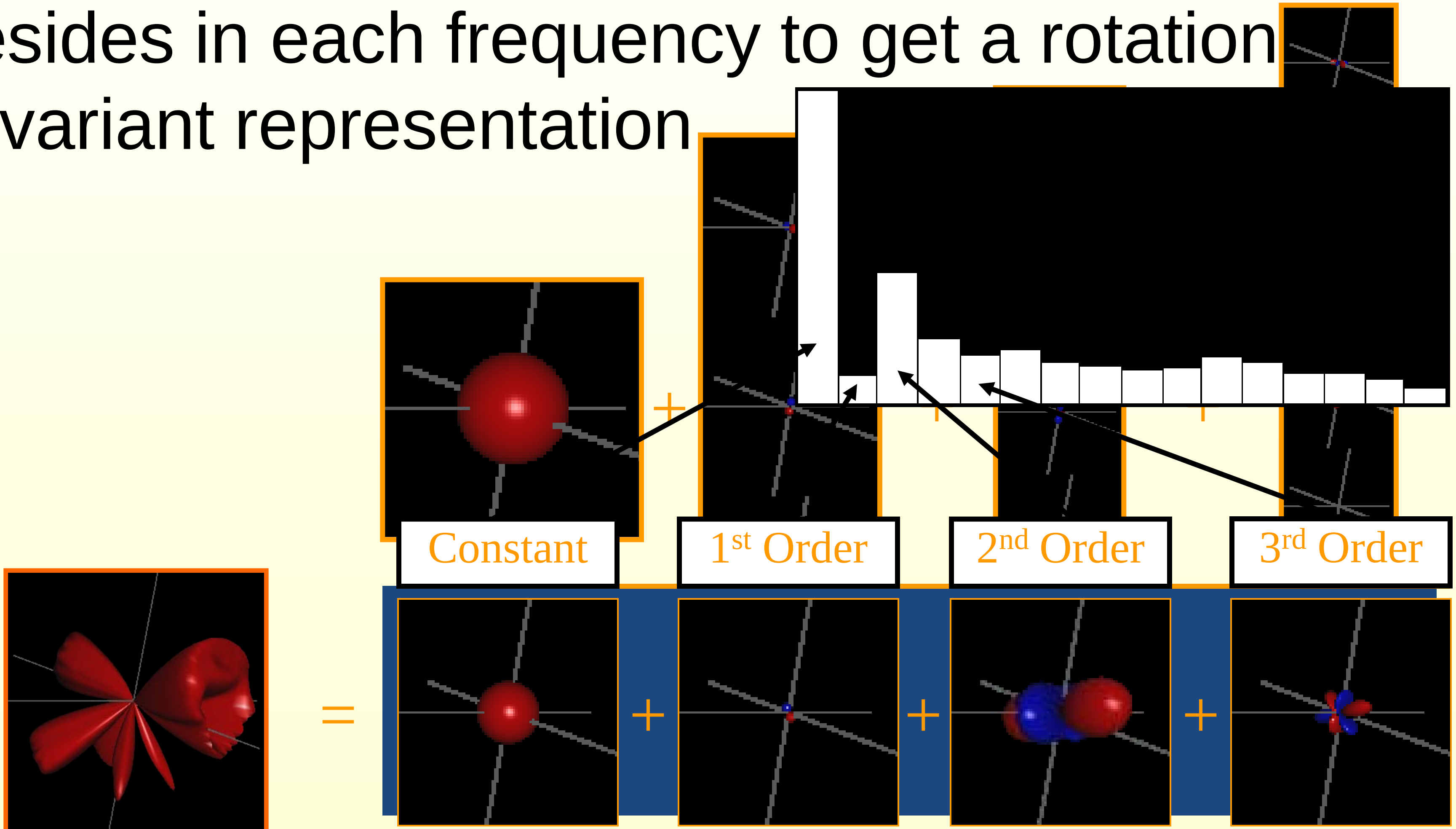
Rotation Invariance

Frequency subspaces are fixed by rotations:



Rotation Invariance

Store “how much” (L_2-norm) of the shape resides in each frequency to get a rotation invariant representation



Shape Descriptors: Alignment

Invariance:

Represent a model by a shape descriptor that is independent of the pose.

Properties:

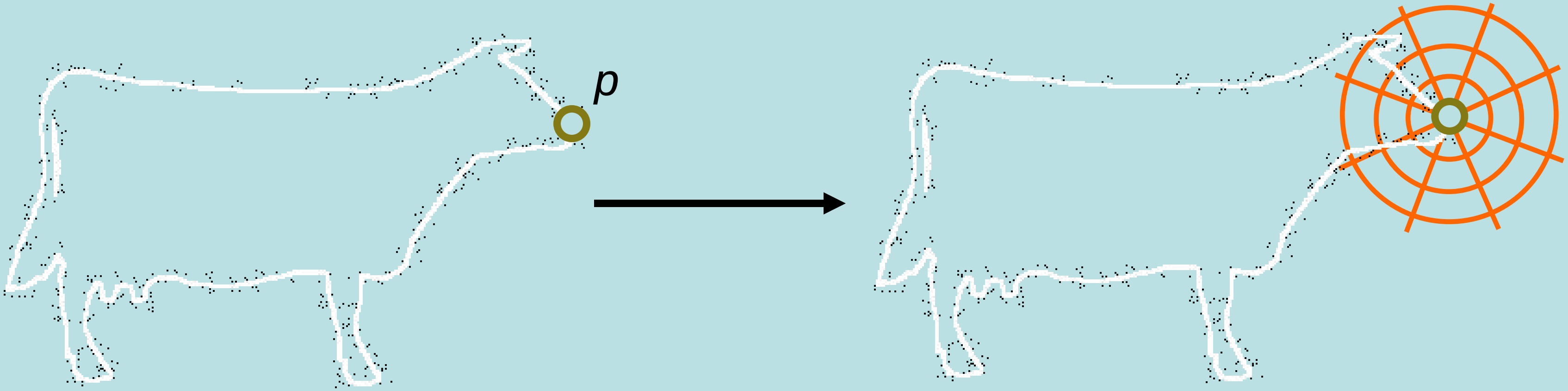
- Compact representation
- Not always discriminating

Local Shape Similarity

From Global to Local

To characterize the surface about a point *p*, take a global descriptor and:

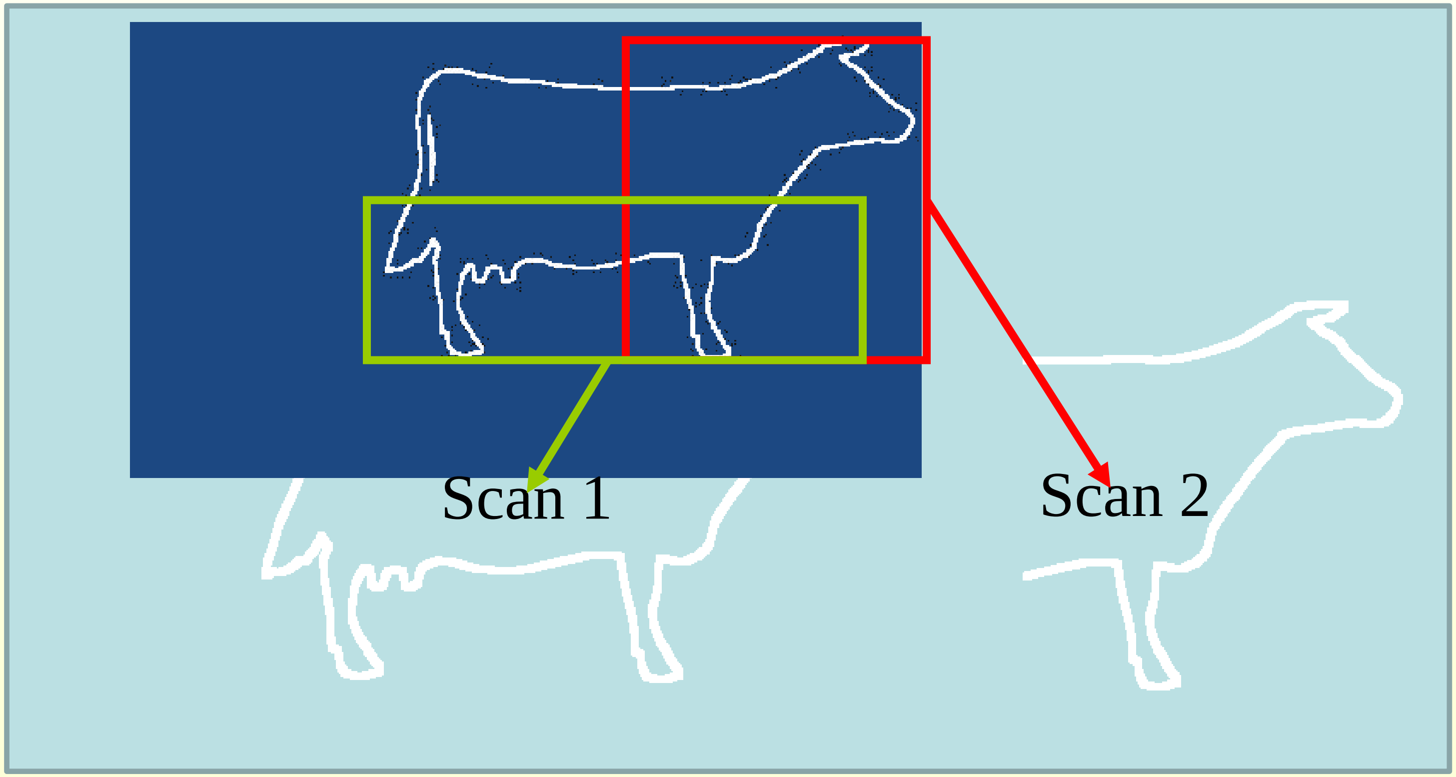
- center it about *p* (instead of the COM), and
- restrict the extent to a small region about *p*.



Shape histograms as local shape descriptors

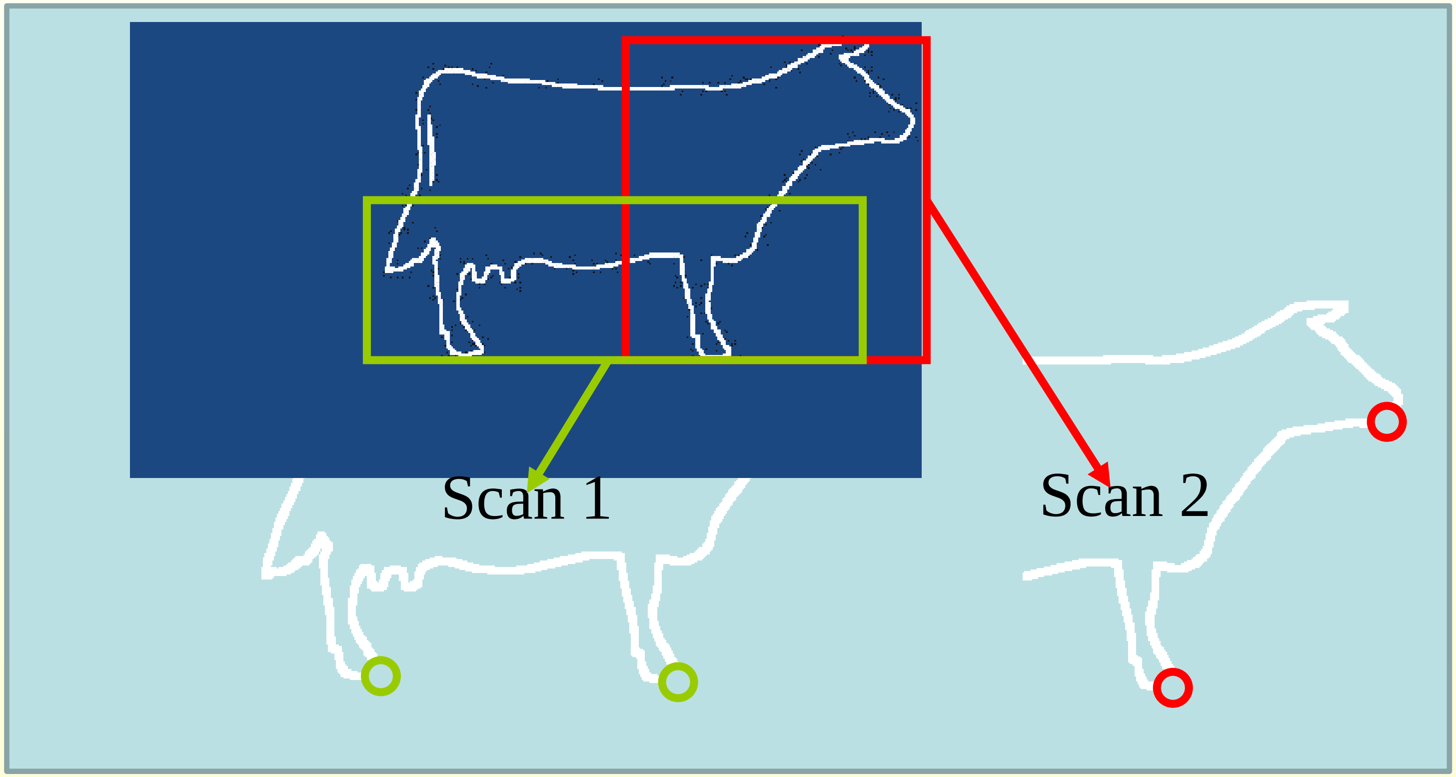
From Global to Local

Given scans of a model:



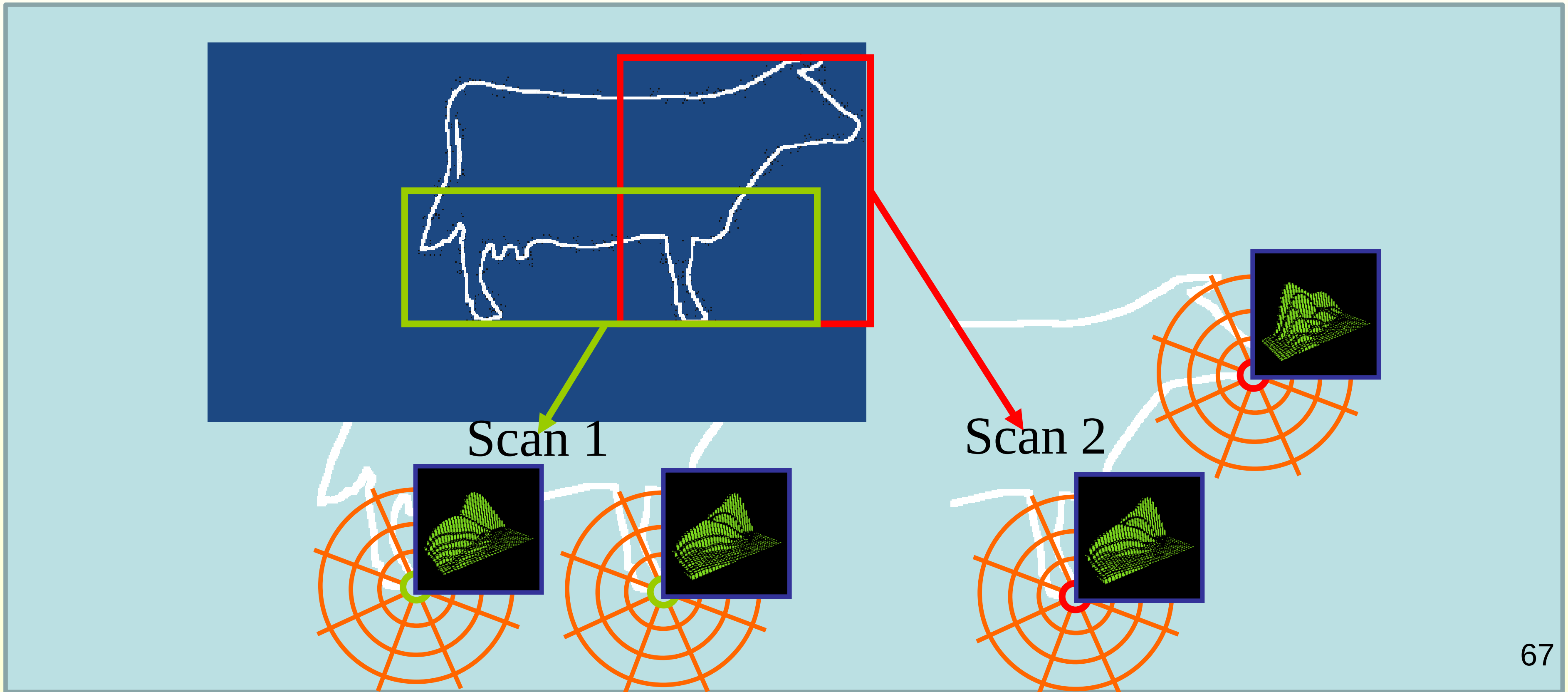
From Global to Local

- Identify the features



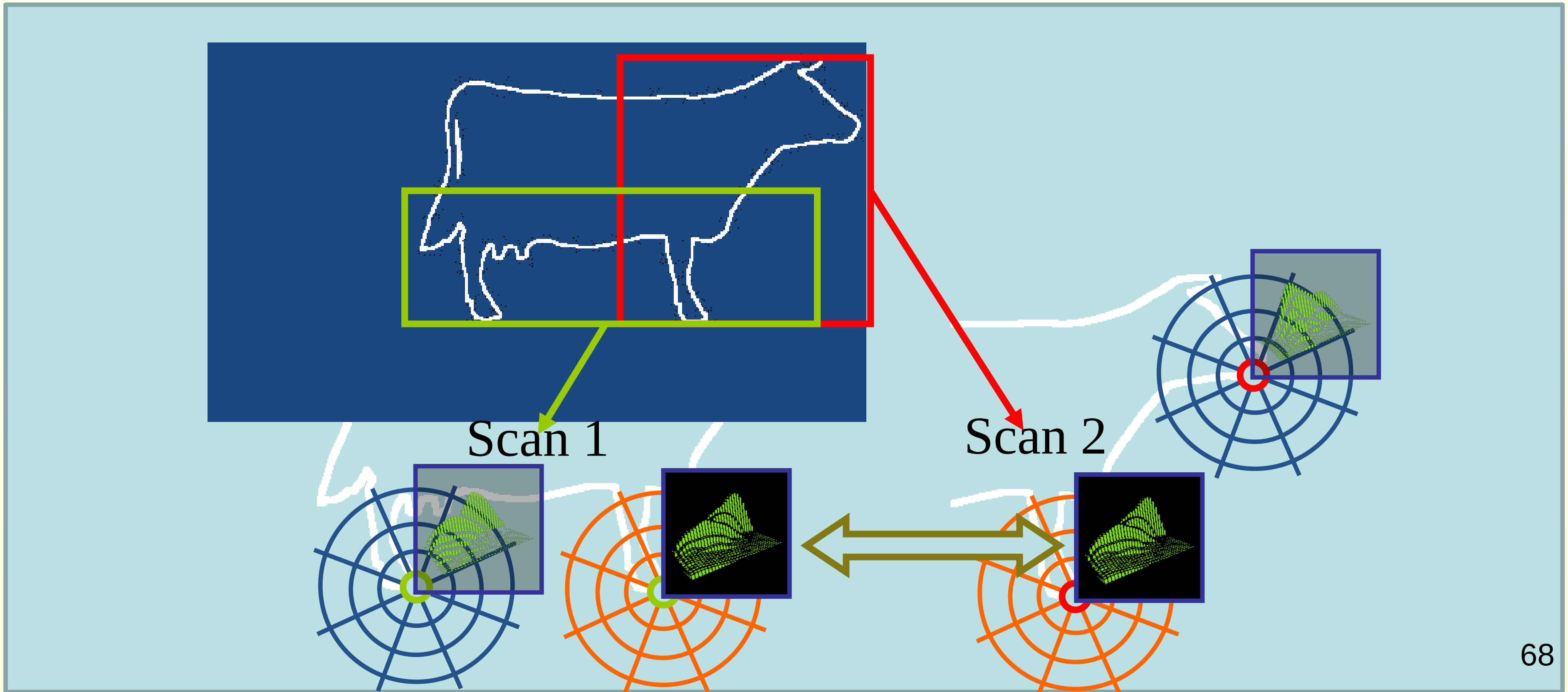
From Global to Local

- Identify the features
- Compute a local descriptor for each feature



From Global to Local

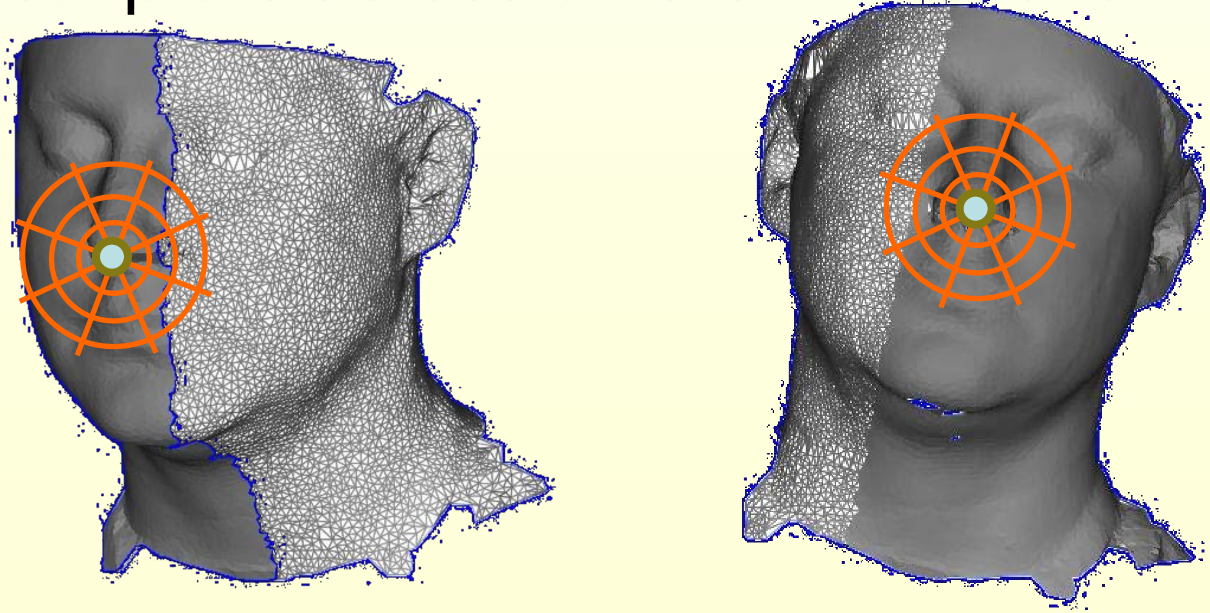
- Identify the features
- Compute a local descriptor for each feature
- Features correspond → descriptors are similar



Pose Normalization

From Global to Local

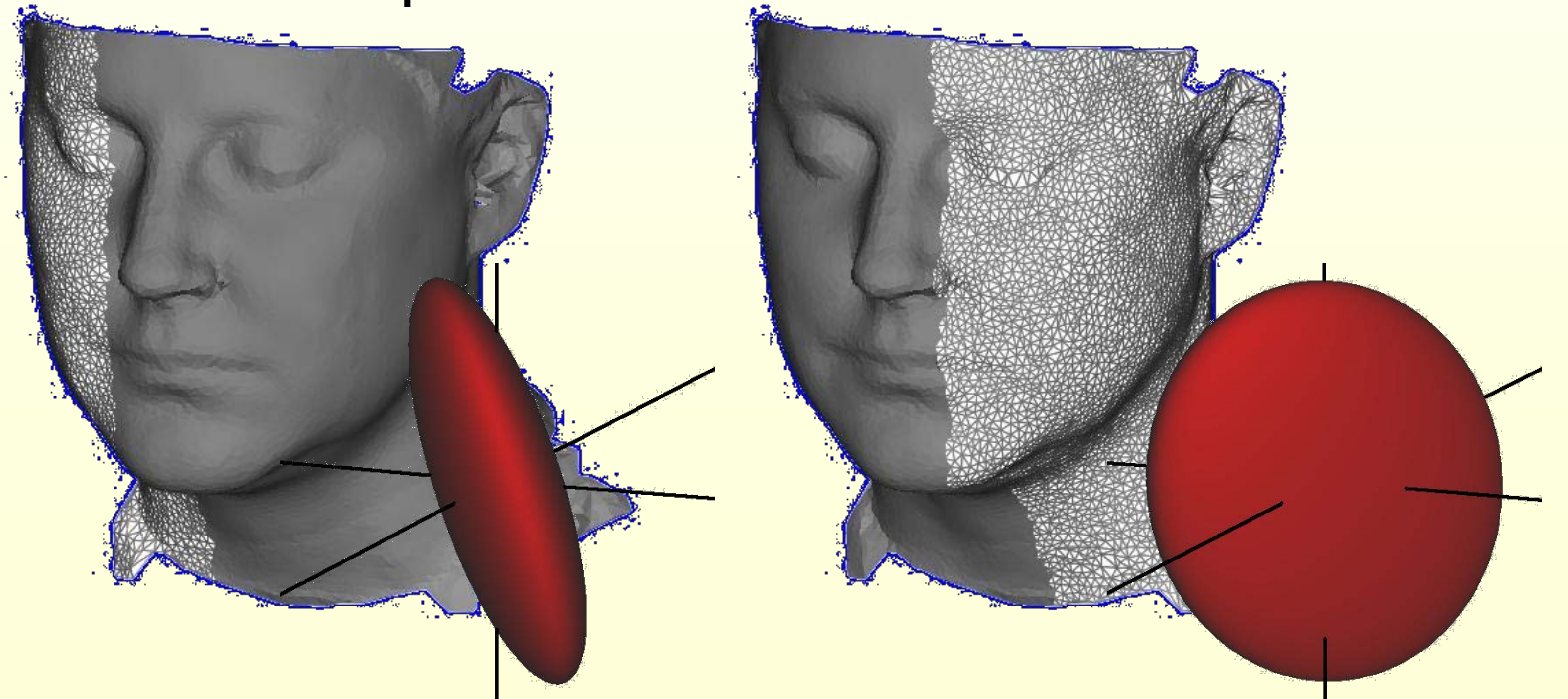
- ✓ Translation: Accounted for by centering the descriptor at the point of interest.
- ✗ Rotation: We still need to be able to match descriptors across different rotations.



Pose Normalization

Challenge

- Since only parts of the models are given, we cannot use global normalization to align the local descriptors



Pose Normalization

Challenge

- Since only parts of the models are given, we cannot use global normalization to align the local descriptors

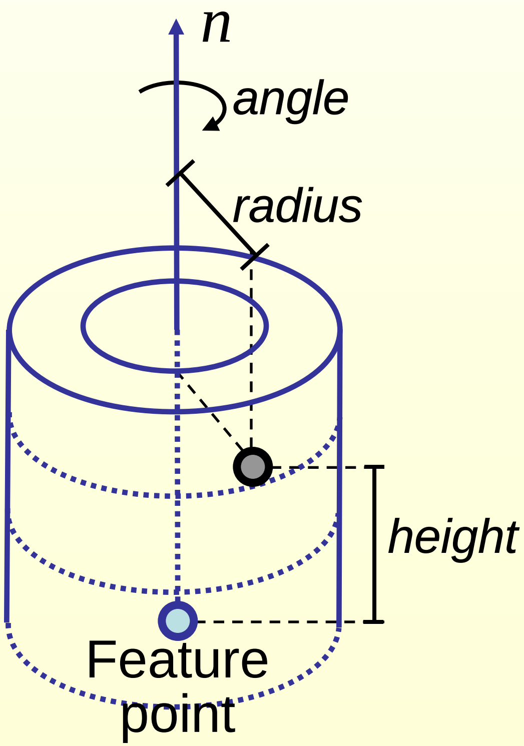
Solutions

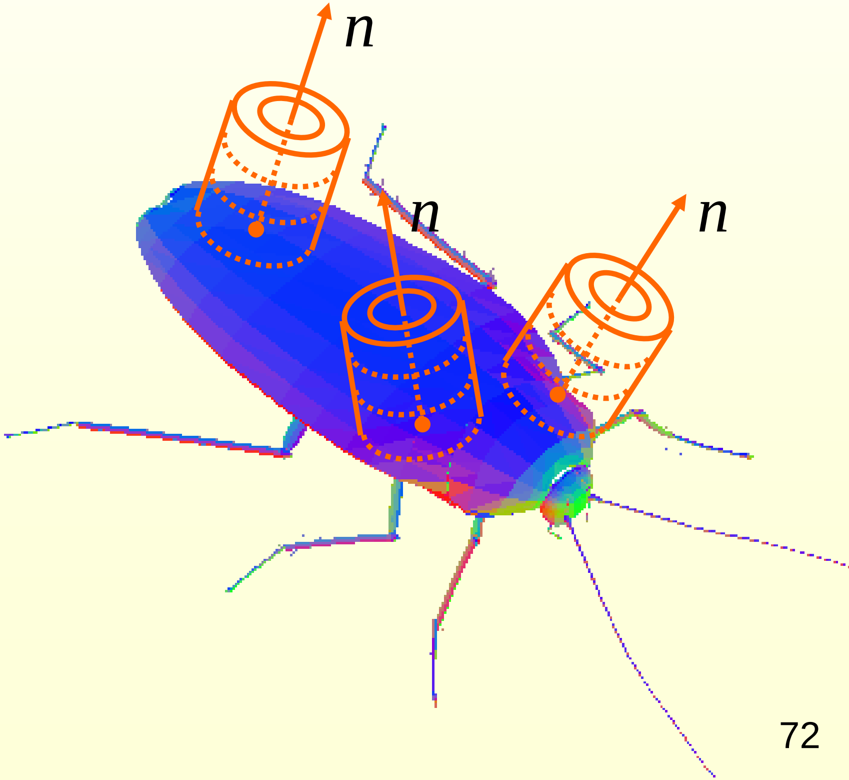
- Normalize using local information

Local Descriptors: Examples

Variations of Shape Histograms:

For each feature, represent its local geometry in cylindrical coordinates about the normal



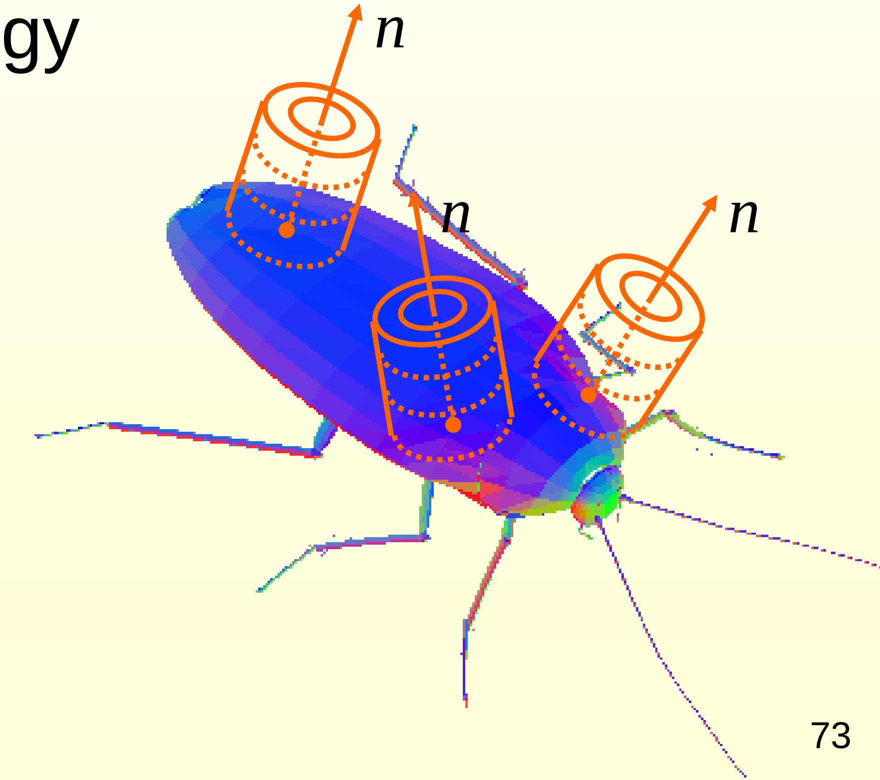


Local Descriptors: Examples

<u>Variations of Shape Histograms</u>:

For each feature, represent its local geometry in cylindrical coordinates about the normal

- ***Spin Images***: Store energy in each normal ring

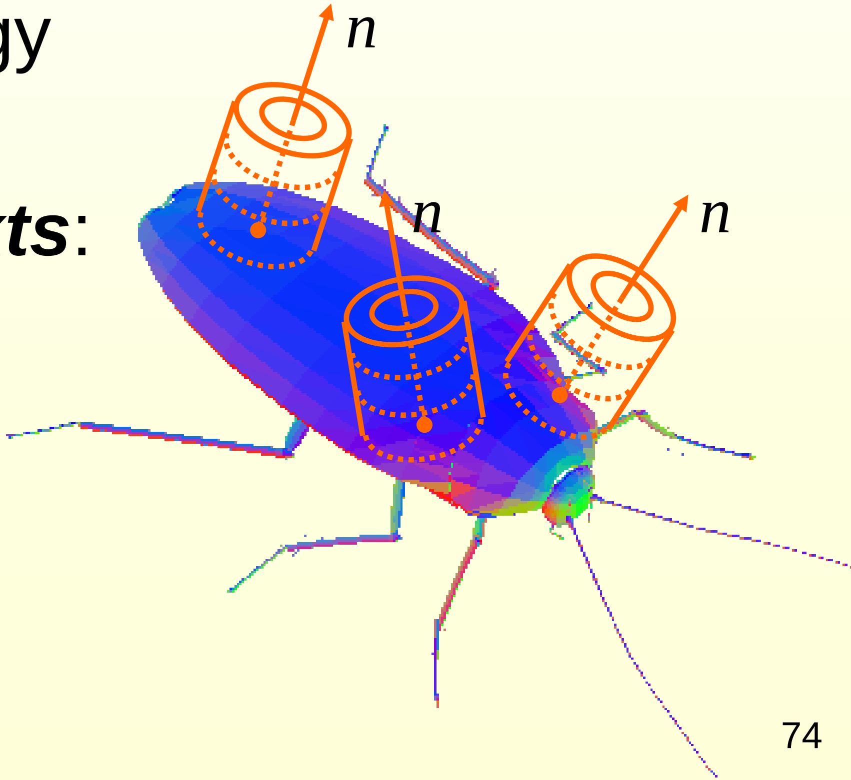


Local Descriptors: Examples

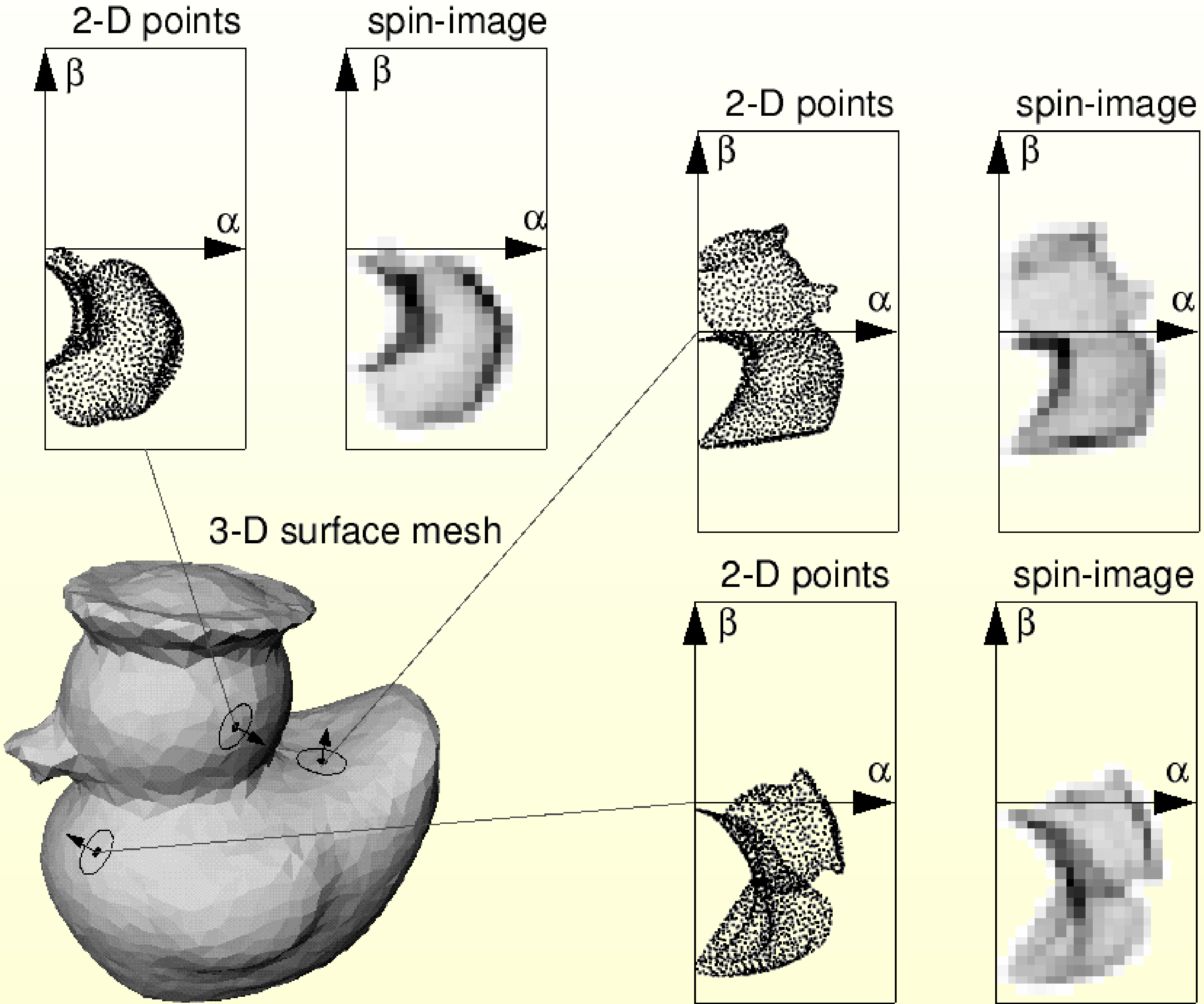
Variations of Shape Histograms:

For each feature, represent its local geometry in cylindrical coordinates about the normal

- ***Spin Images***: Store energy in each normal ring
- ***Harmonic Shape Contexts***: Store power spectrum of each normal ring



Spin Images

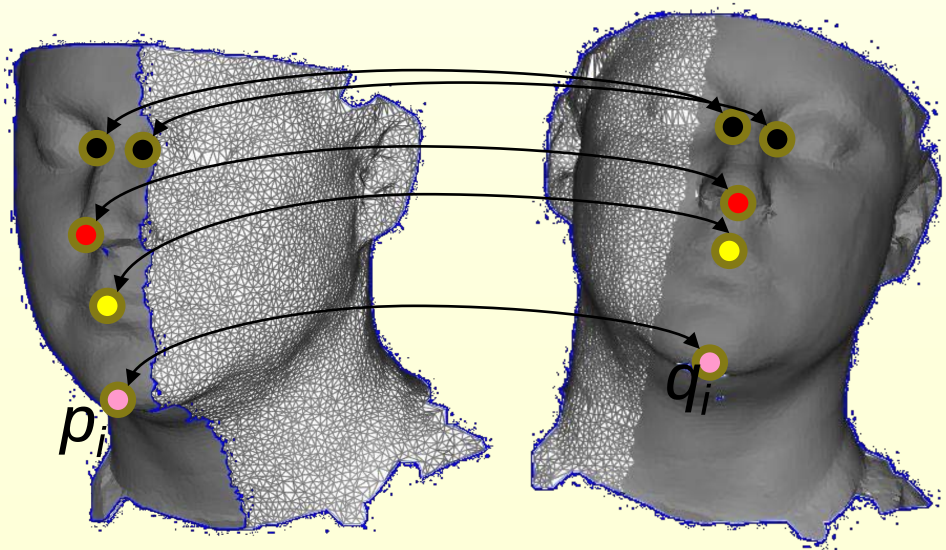

2-D points
spin-image
β
α
3-D surface mesh
2-D points
spin-image
2-D points
spin-image

Registration With and Without Correspondences

Registration

Ideal Case:

- Every feature point on one scan has a (single) corresponding feature on the other.

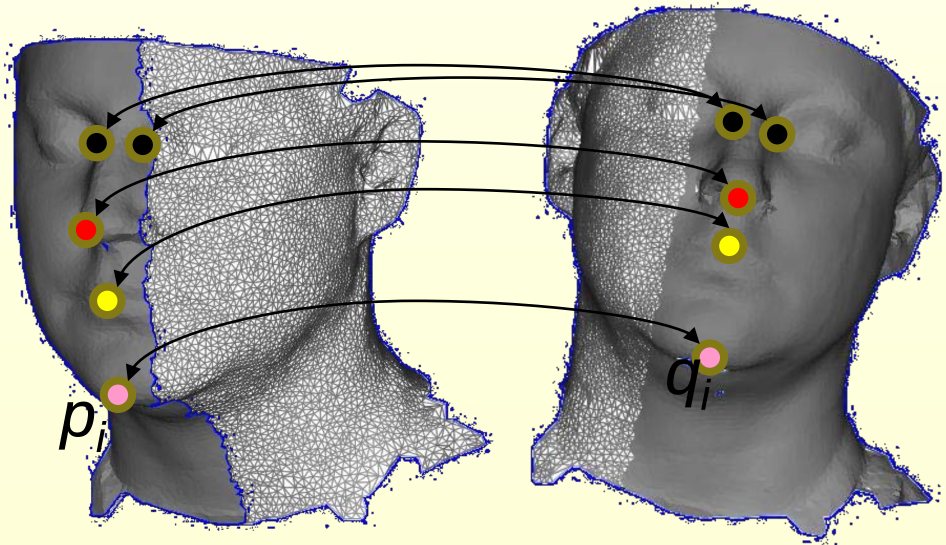


Registration

<u>Ideal Case</u>:

- Every feature point on one scan has a (single) corresponding feature on the other.
- Solve for the optimal transformation *T*:

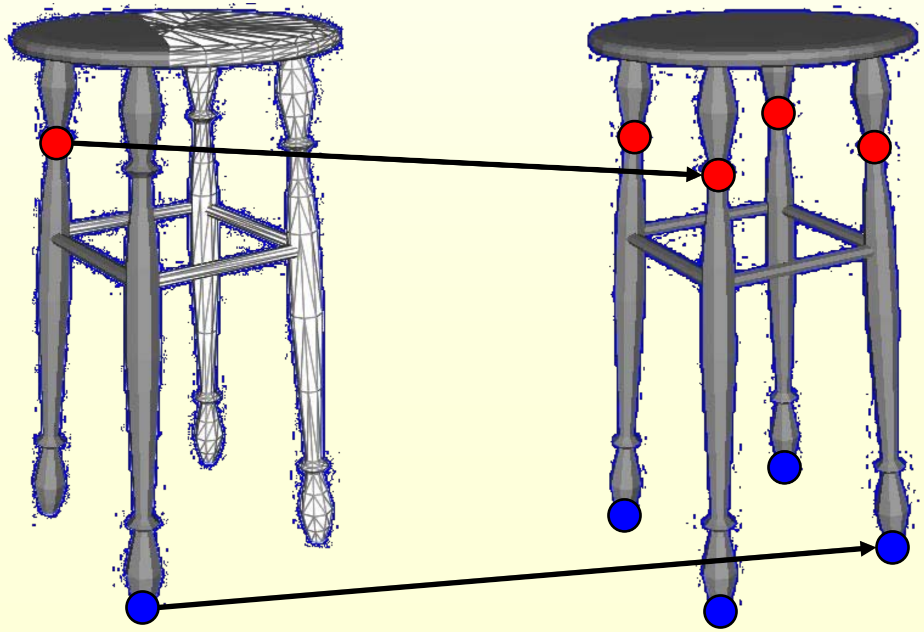
$$\sum_{i=1}^{n} \left\| p_i - T(q_i) \right\|^2$$



Registration

Challenge:

- Even with good descriptors, symmetries in the model and the locality of descriptors can result in multiple and incorrect correspondences



Registration

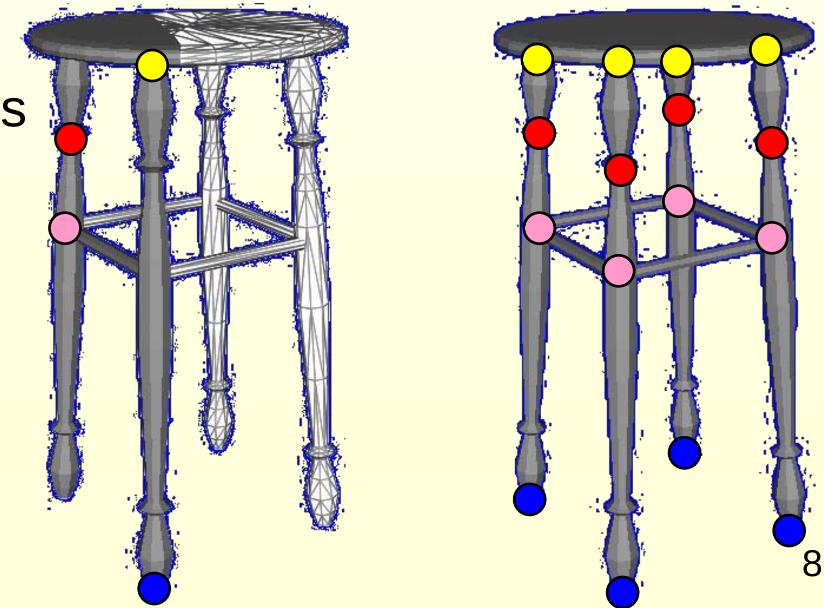
Exhaustive Search:

- Compute alignment error at each permutation of correspondences and use the optimal one.

$$\text{Error} = \arg\min_{\pi\in\Psi}\left(\arg\min_{T\in E^3}\sum_{i=1}^{n}\left\|p_i - T(\pi(p_i))\right\|^2\right)$$

Ψ = Set of possible correspondence

E^3 = Group of rigid body transformations



Registration

Exhaustive Search:

- Compute alignment error at each permutation of correspondences and use the optimal one.

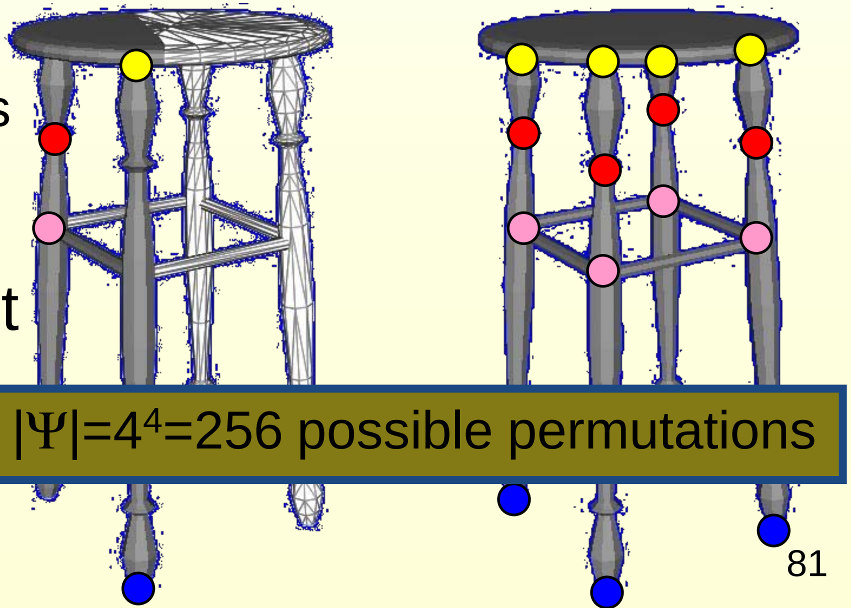
$$\text{Error} = \underset{\pi \in \Psi}{\arg\min} \left(\underset{T \in E^3}{\arg\min} \sum_{i=1}^{n} \left\| p_i - T(\pi(p_i)) \right\|^2 \right)$$

Ψ = Set of possible correspondence

E^3 = Group of rigid body transformations

Given points $\{p_1, \ldots, p_n\}$ on the query, if p_i matches m_i different target points:

$$|\Psi| = \prod_{i=1}^{n} m_i$$

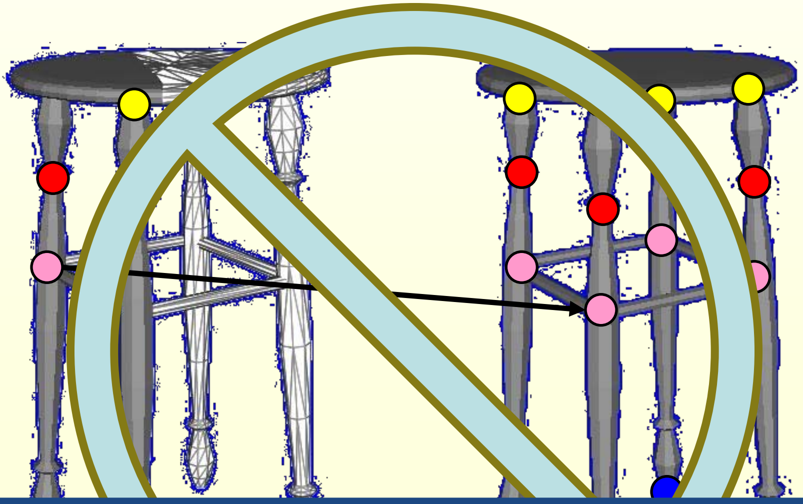


$|\Psi| = 4^4 = 256$ possible permutations

Branch and Bound

Key Idea:

- Try all permutations but terminate early if the alignment can be predicted to be bad.

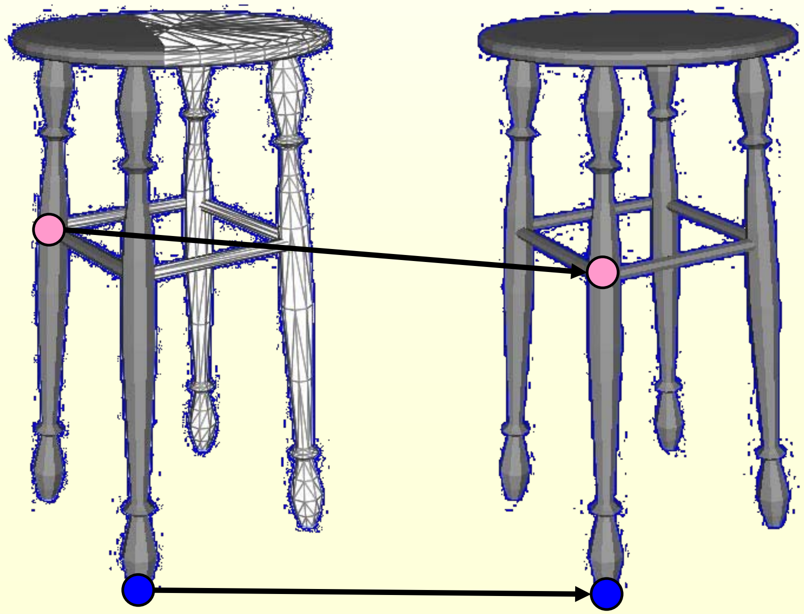


By performing two comparisons, it was possible to eliminate 16 different possibilities

Branch and Bound

Goal:

- Need to be able to determine if the alignment will be a good one without knowing all of the correspondences.



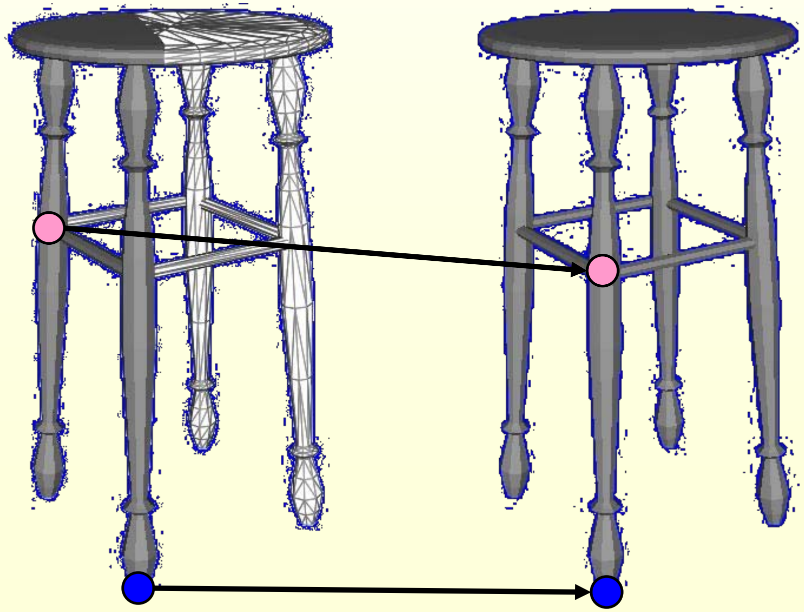
Branch and Bound

Goal:

- Need to be able to determine if the alignment will be a good one without knowing all of the correspondences

Observation:

- Alignment needs to preserve the lengths between points in a single scan



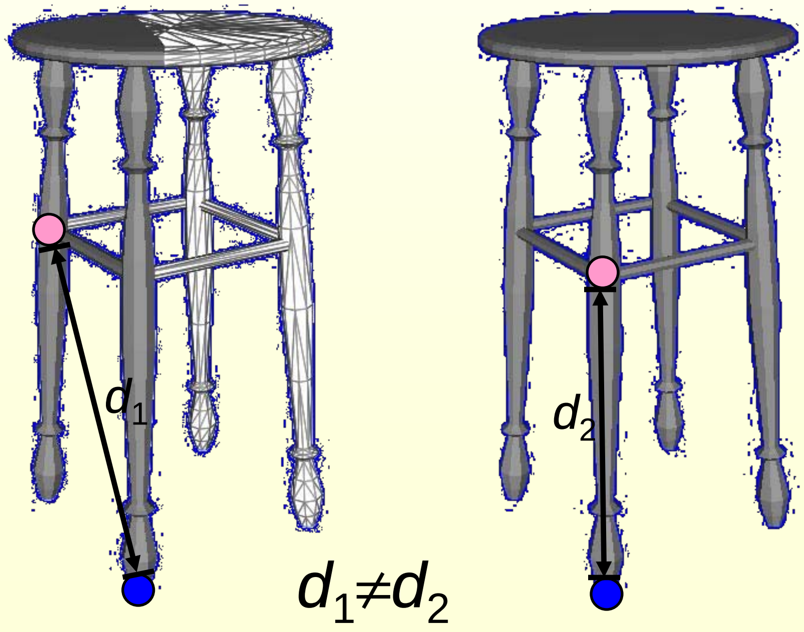
Branch and Bound

Goal:

- Need to be able to determine if the alignment will be a good one without knowing all of the correspondences

Observation:

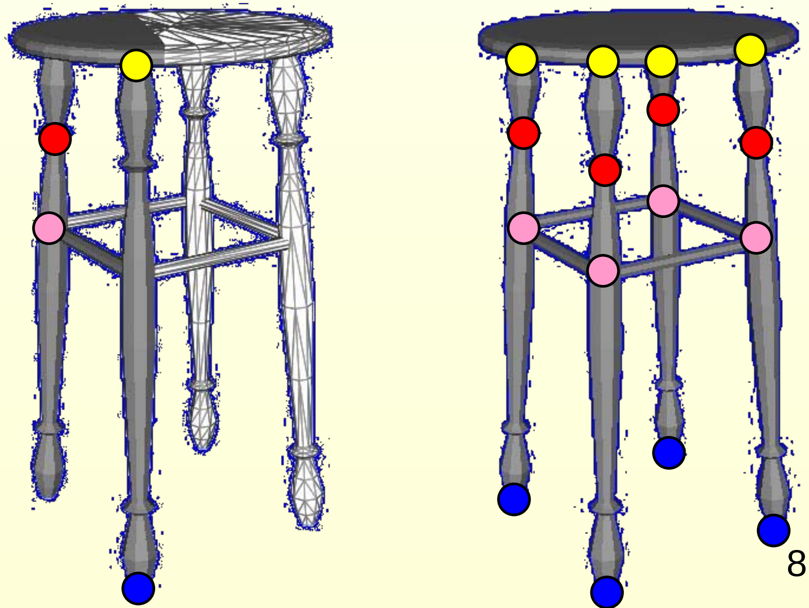
- Alignment needs to preserve the lengths between points in a single scan



Random Sample Consensus (RANSAC)

Observation:

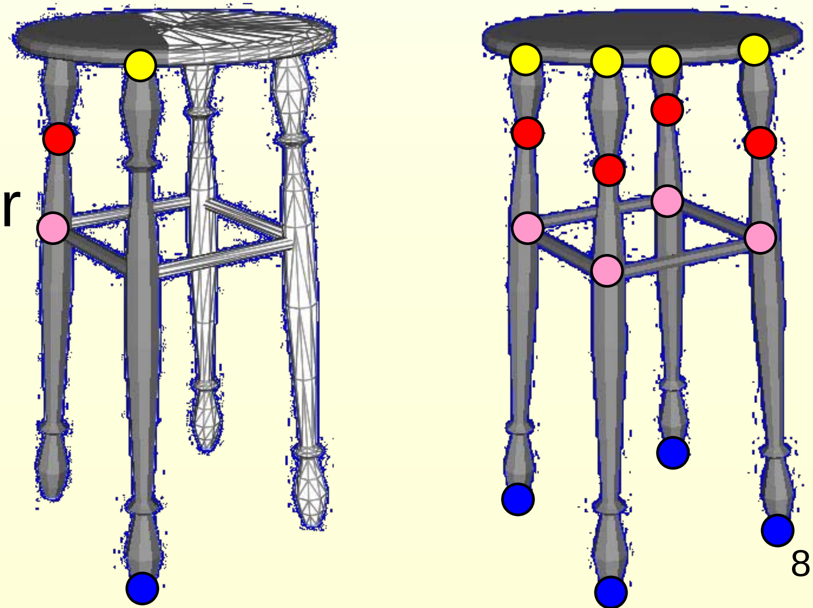
- In 3D only three pairs of corresponding points are needed to define a rigid motion



Random Sample Consensus

Algorithm:

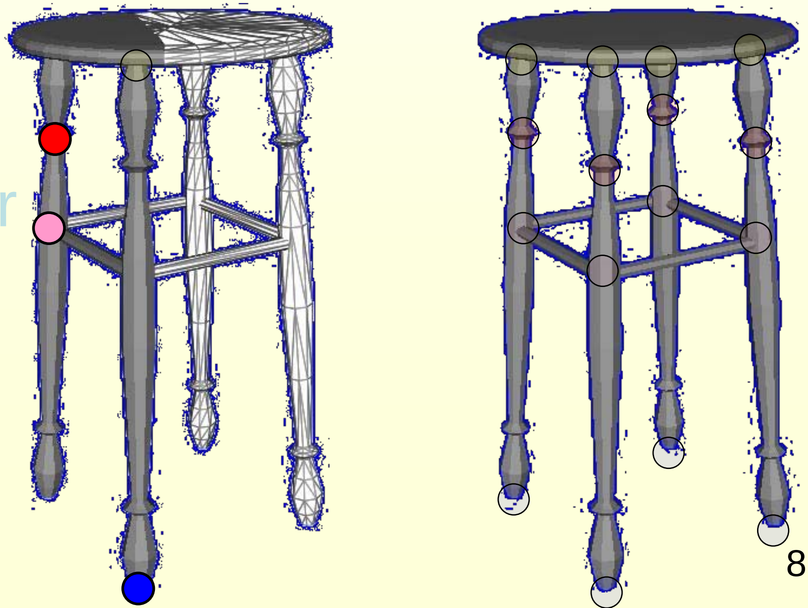
- Randomly choose three points on source
- For all possible correspondences on target:
  - Compute the aligning transformation *T*
  - For every other source point *p* :
    - Find corresponding point closest to *T*(*p*)
  - Compute alignment error



Random Sample Consensus

Algorithm:

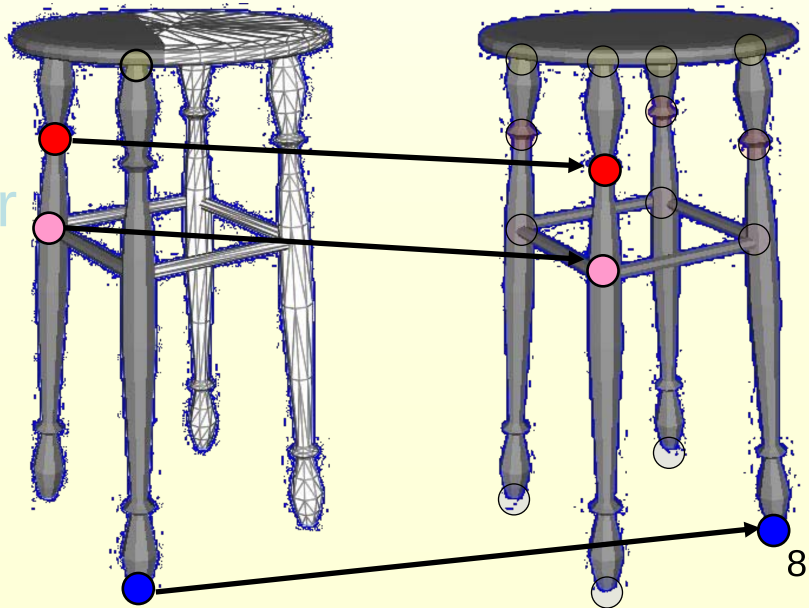
- Randomly choose three points on source
- For all possible correspondences on target:
  - Compute the aligning transformation *T*
  - For every other source point *p* :
    - Find corresponding point closest to *T*(*p*)
  - Compute alignment error



Random Sample Consensus

Algorithm:

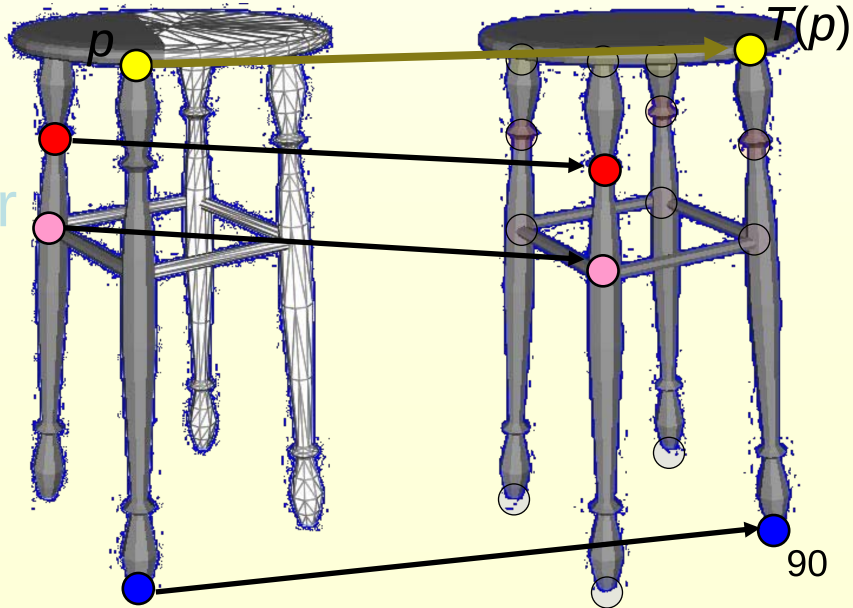
- Randomly choose three points on source
- For all possible correspondences on target:
  - Compute the aligning transformation *T*
  - For every other source point *p* :
    - Find corresponding point closest to *T*(*p*)
  - Compute alignment error



Random Sample Consensus

Algorithm:

- Randomly choose three points on source
- For all possible correspondences on target:
  - Compute the aligning transformation *T*
  - For every other source point *p* :
    - Find corresponding point closest to *T*(*p*)
  - Compute alignment error

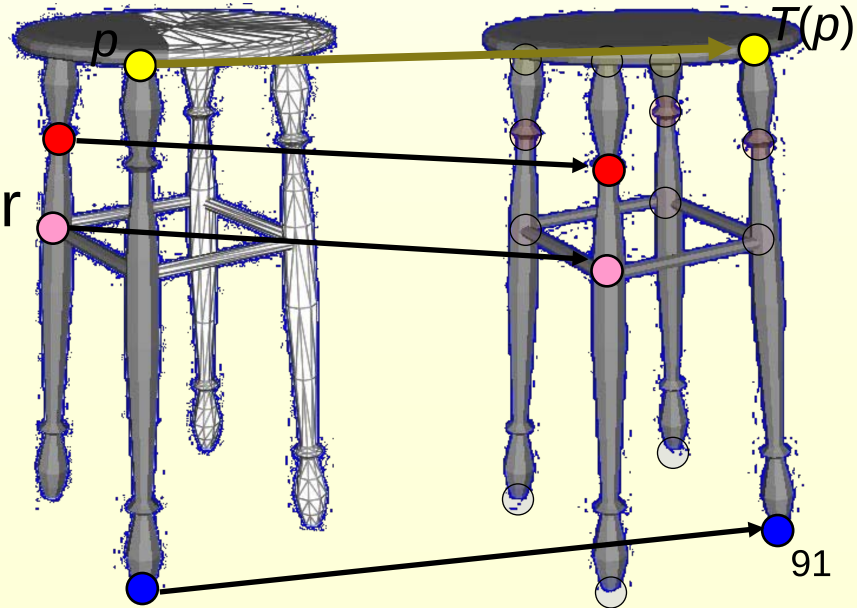


Random Sample Consensus

Algorithm:

- Randomly choose three points on source
- For all possible correspondences on target:
  - Compute the aligning transformation *T*
  - For every other source point *p* :
    - Find corresponding point closest to *T*(*p*)
  - Compute alignment error

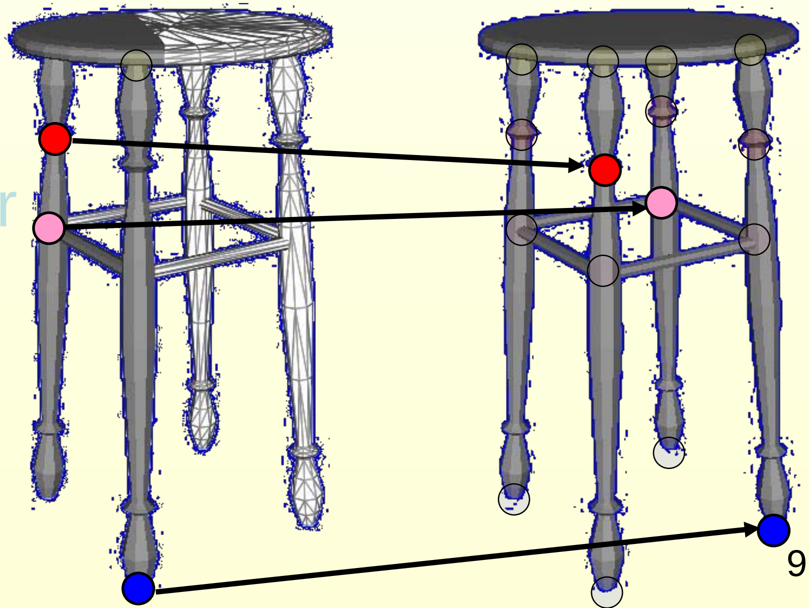
Error = 0



Random Sample Consensus

Algorithm:

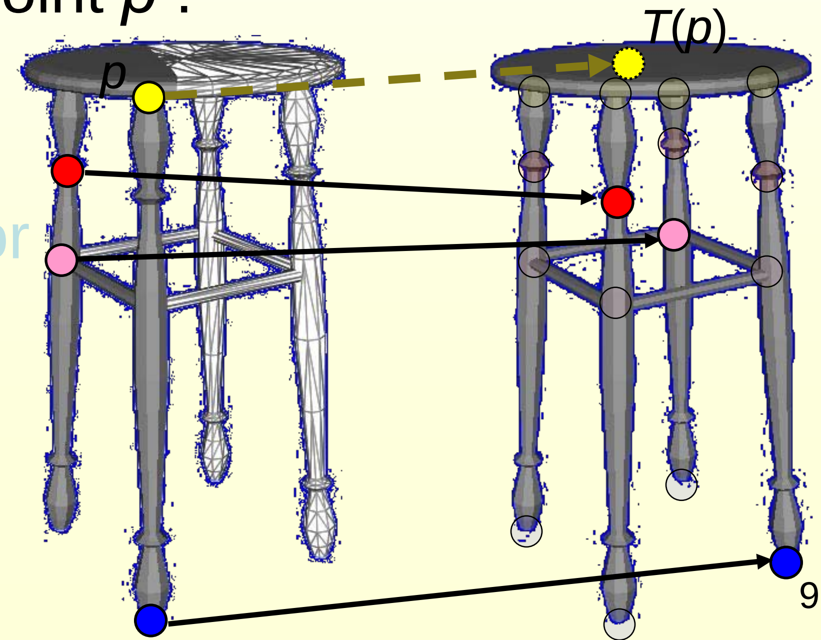
- Randomly choose three points on source
- For all possible correspondences on target:
  - Compute the aligning transformation *T*
  - For every other source point *p* :
    - Find corresponding point closest to *T*(*p*)
  - Compute alignment error



Random Sample Consensus

Algorithm:

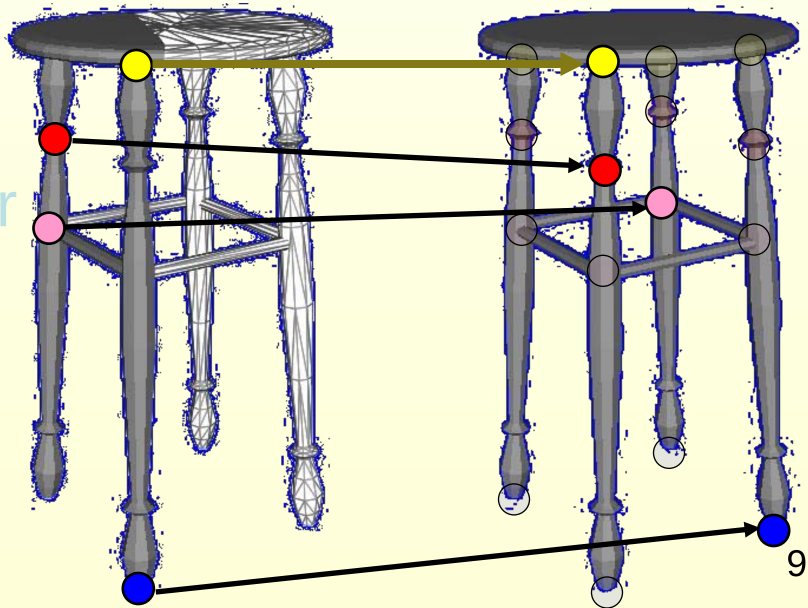
- Randomly choose three points on source
- For all possible correspondences on target:
  - Compute the aligning transformation *T*
  - For every other source point *p* :
    - Find corresponding point closest to *T*(*p*)
  - Compute alignment error



Random Sample Consensus

Algorithm:

- Randomly choose three points on source
- For all possible correspondences on target:
  - Compute the aligning transformation *T*
  - For every other source point *p* :
    - Find corresponding point closest to *T*(*p*)
  - Compute alignment error

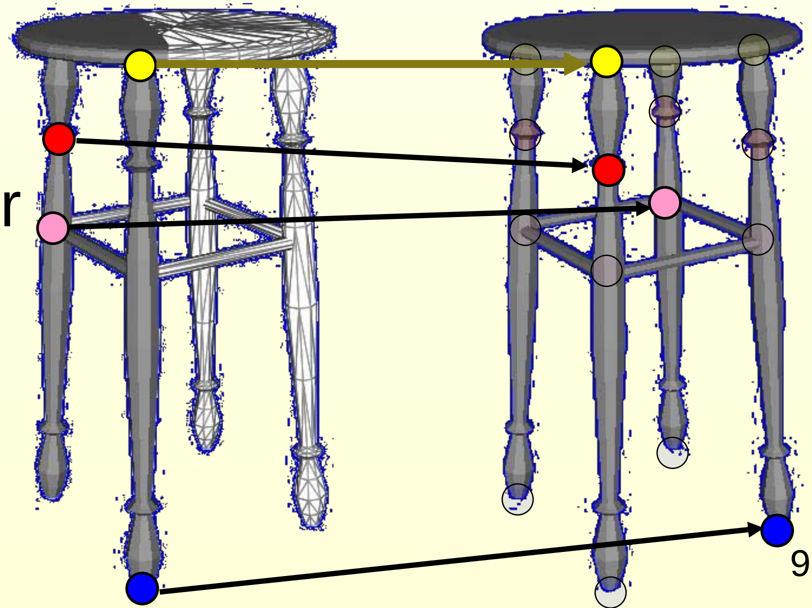


Random Sample Consensus

Algorithm:

- Randomly choose three points on source
- For all possible correspondences on target:
  - Compute the aligning transformation *T*
  - For every other source point *p* :
    - Find corresponding point closest to *T*(*p*)
  - Compute alignment error

Error > 0

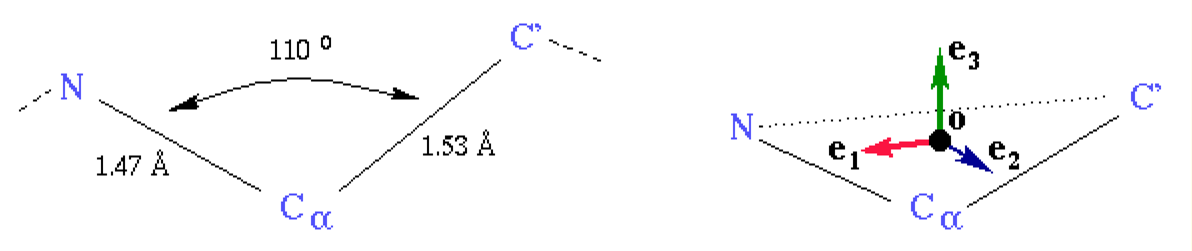


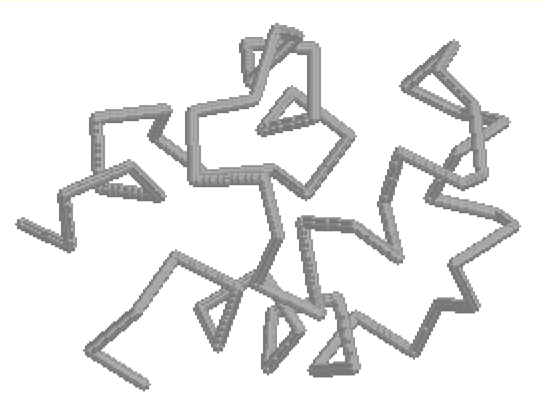
Geometric Hashing: Protein Alignment

Key idea: A rigid motion is defined by few (six) parameters or degrees of freedom. A good aligning motion will bring many atoms of *A* close to corresponding atoms of *B*. If we can guess a small number of these correspondences, that is sufficient to approximately determine the appropriate transform.

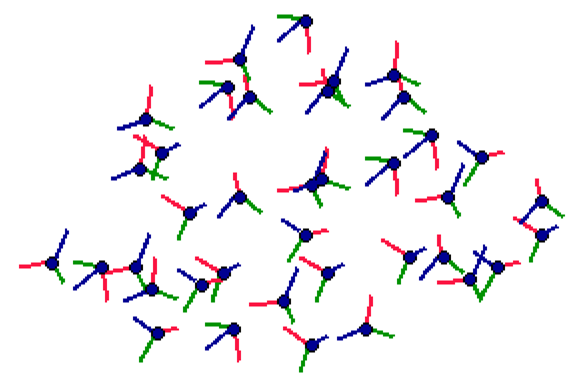
[Wolfson *et. al*., 88]
[Pennec, '98]

One approach:





Each C_α atom becomes a reference frame



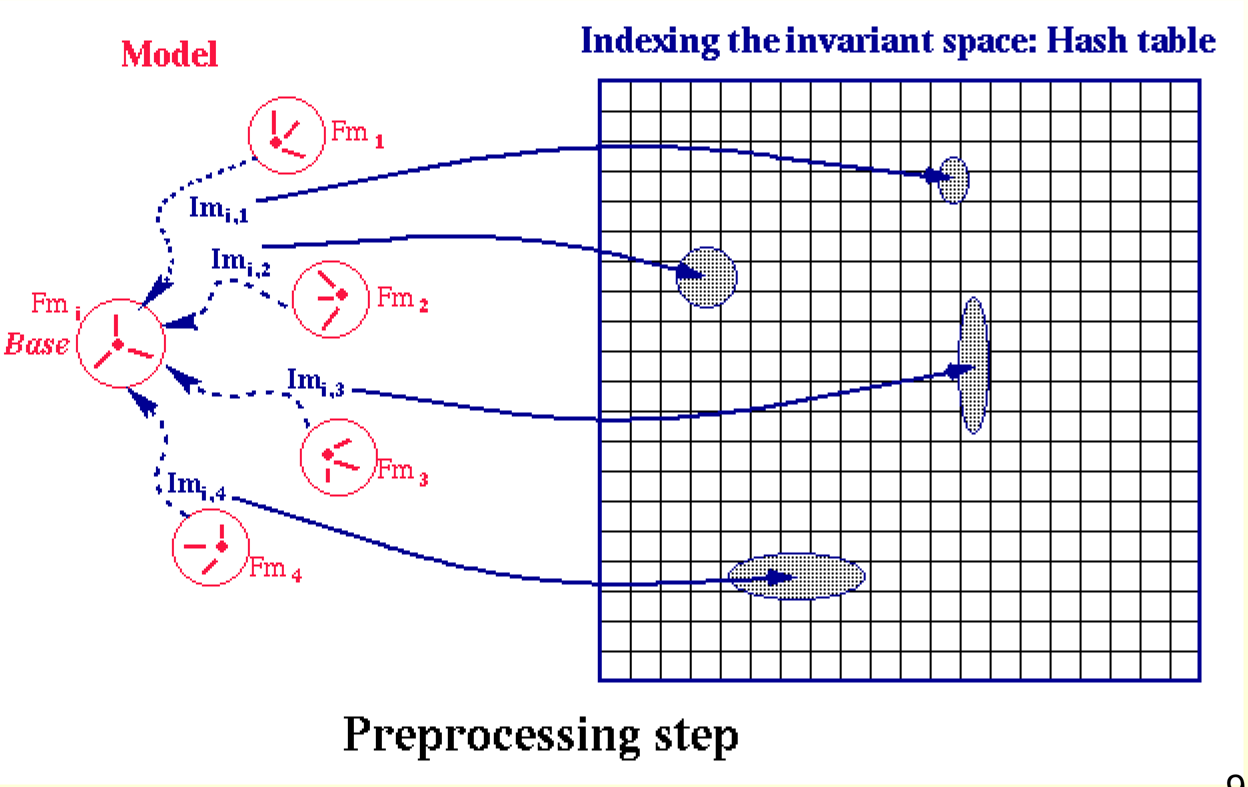
Matching a reference frame of *A* to one of *B* fully defines a rigid motion

Invariant Representation

Preprocessing: For each *base frame*, we store the transforms between the base frame and all other frames of the model *A*. We discretize the 6-d transform space into a grid. Each grid cell holds the pairs (base frame, individual frame) that fall in it.

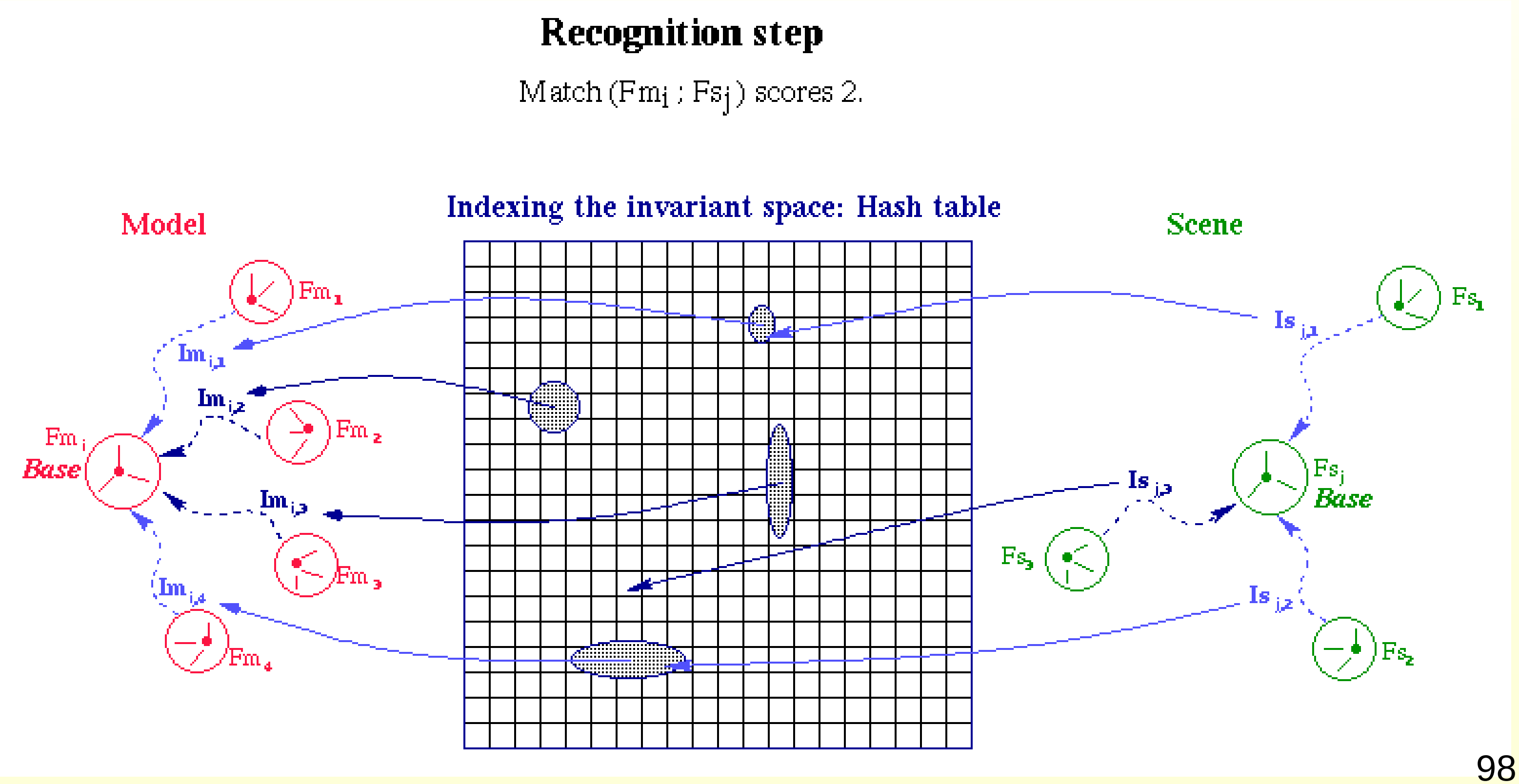
The bin size for the grid has to be chosen carefully.

Since the grid is typically very large and sparcely occupied, a hash table is used.



Matching by Voting

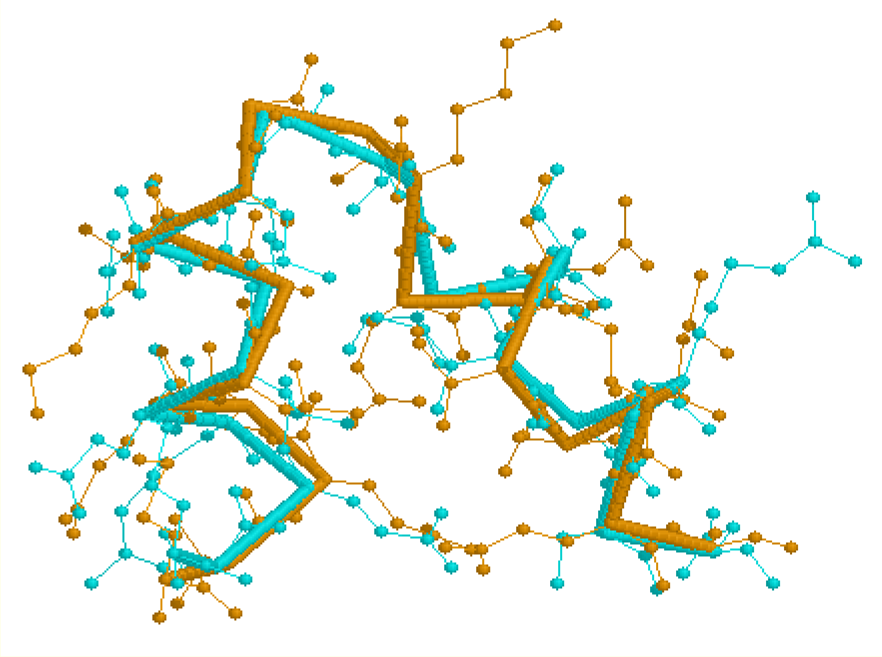
Query: Next, in succession, we choose each frame of *B* as a base frame. We hash the rigid motion between all other frames of *B and the base frame* of *B*. Each time we land in a cell obtaining some frames of *A*, *we cast a vote in favor of the corresponding A base frame.*



Picking the Winners

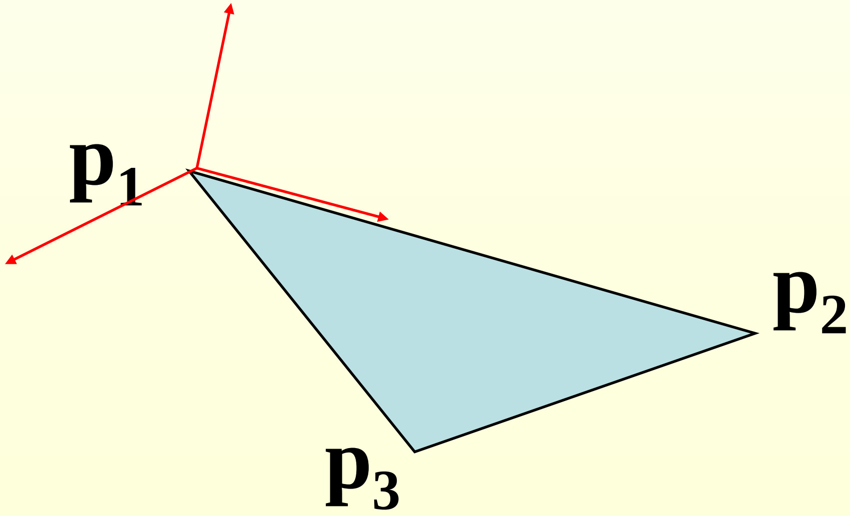
- Each pair (ref. frame of *A*, ref. frame of *B*) that gets many of votes becomes a candidate for an approximation to the optimum alignment
- Each candidate alignment is further improved by ICP and then scored

Alignment of the HTH motifs of the E. Coli TRP repressor and the Phage 434 Cro protein



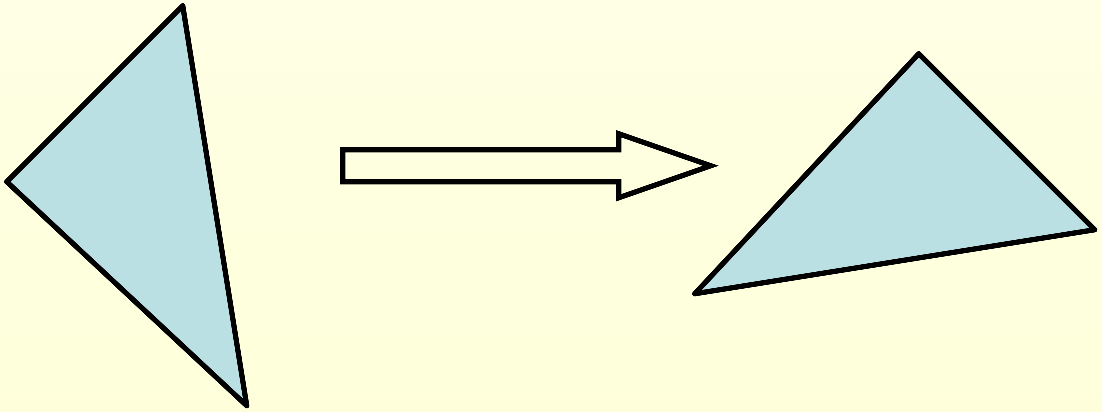
Another Variation on GH

A 3-D reference frame can be uniquely defined by the ordered vertices of a non-degenerate triangle of any three vertices or atoms p_1, p_2, and p_3 (these points need not be neighbors).



Aligning Triangles

- For each pair of triplets, one from each shape, which define 'almost' congruent non-degenerate triangles, we can compute a rigid motion that superimposes them.
- We can now perform geometric hashing using these more general bases for the transformations.



Geometric Hashing - Preprocessing

- Pick a reference frame satisfying pre-specified constraints
- Compute the coordinates of all the other points (in a pre-specified neighborhood) in this reference frame
- Use each coordinate as an address to the hash (look-up) table and record in that entry the (ref. frame, shape signature, point).
- Repeat above steps for each reference frame.

Geometric Hashing - Recognition 1

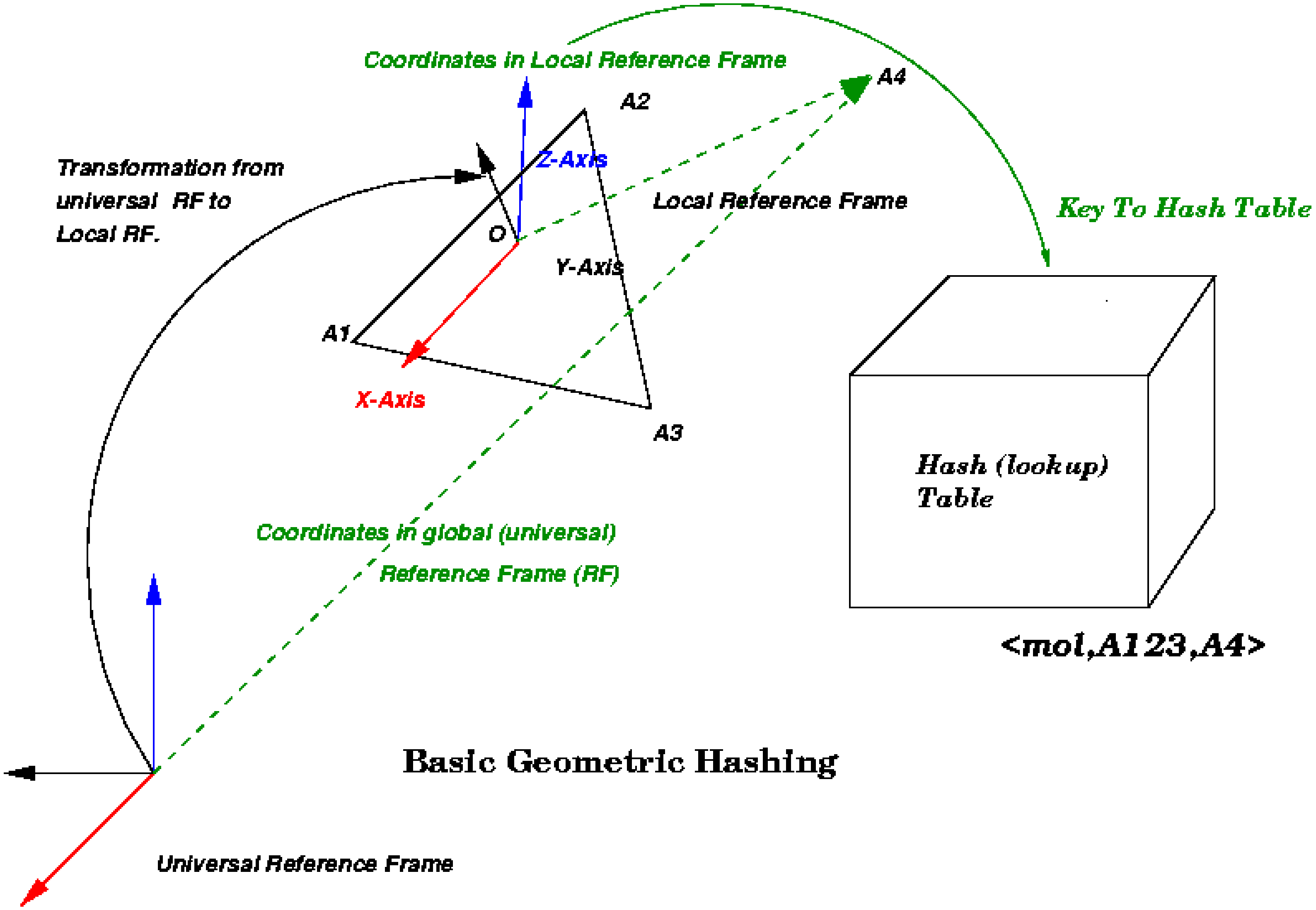
For the target protein do :

- Pick a *reference frame* satisfying pre-specified constraints
- Compute the coordinates of all other points in the current reference frame
- Use each coordinate to access the hash-table to retrieve all the records (reference frame, shape signature, point).

Geometric Hashing – Recognition 2

- For records with matching shape signatures, vote for the (reference frame)
- Compute the transformations of the “high scoring” hypotheses
- Repeat the above steps for each reference frame

- Cluster similar transformations
- Extend best matches



Basic Geometric Hashing

GH Analysis

- Variation 1 (*n* is number of atoms, *b* the bin size):

$$O(n^2) + O(bn^2) \approx O(n^2) \quad \text{vs. naive } O(n^3)$$

- Variation 2:

$$O(n^4) + O(bn^4) \approx O(n^4) \quad \text{vs. naive } O(n^7)$$

Can assume *b* = *O*(1); large bins do not discriminate, so they can be ignored

naive = “pose clustering”

GH Advantages

- Sequence order independent
- Can match partial, disconnected substructures
- Highly efficient
- Can be applied to protein-protein interfaces, motif detection, docking
- Can match against multiple targets – a trivial extension to the method
- Parallel implementation is straightforward

Locating Shape Features

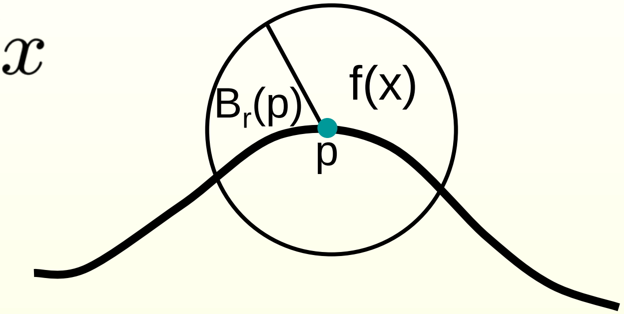
Feature Discovery

1. Use descriptors to identify features
    - Differential vs. integral descriptors
2. Build correspondence search space
    - Few correspondences for each feature
3. Efficiently explore search space
    - Distance error metric
    - Pruning algorithms

Integral Descriptors

[Manay 04]

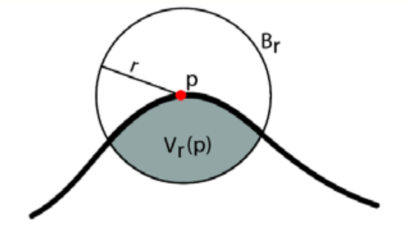
- $$F_r(p) = \int_{B_r(p) \cap S} f(x)\,dx$$

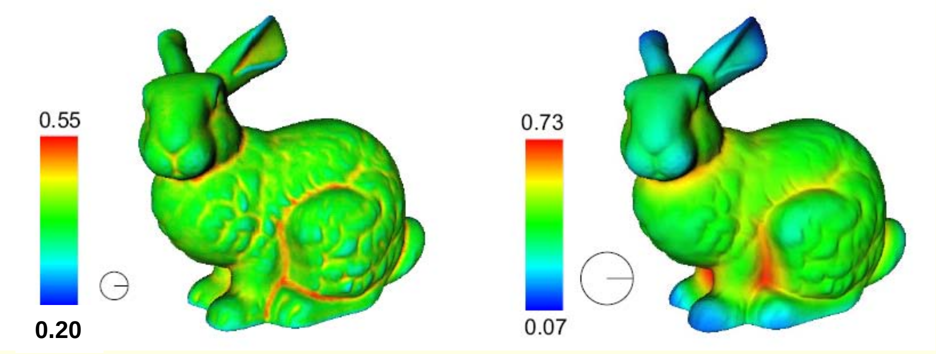


- Multiscale
- Inherent smoothing

Integral Volume Descriptor

- $$V_r(p) = \int_{B_r(p)\cap S} dx$$



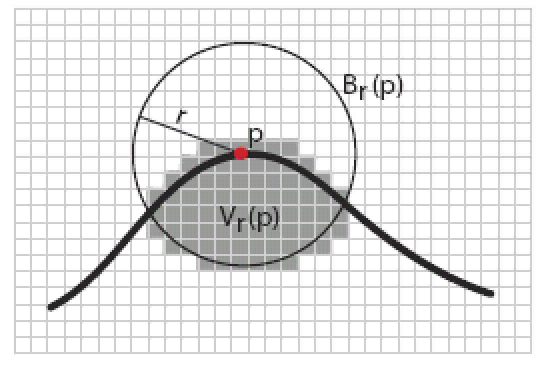


- Relation to mean curvature *H*

$$V_r(\mathbf{p}) = \frac{2\pi}{3} r^3 - \frac{\pi H}{4} r^4 + O(r^5)$$

Descriptor Computation

- Approximate using a voxel grid

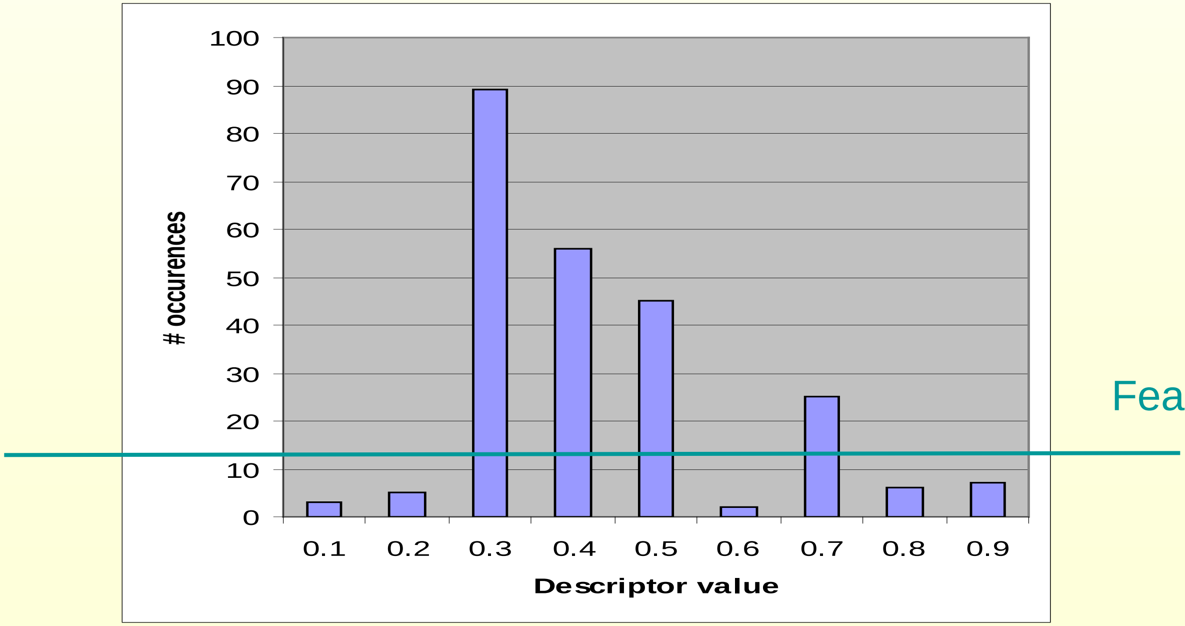


$$V(c) = (G_B * G_O)(c)$$

  - Convolution of occupancy grid with ball

Feature Identification

- Pick as features points with rare descriptor values
  - Rare in the data ⇨ rare in the model ⇨ few correspondences
- Works for any descriptor

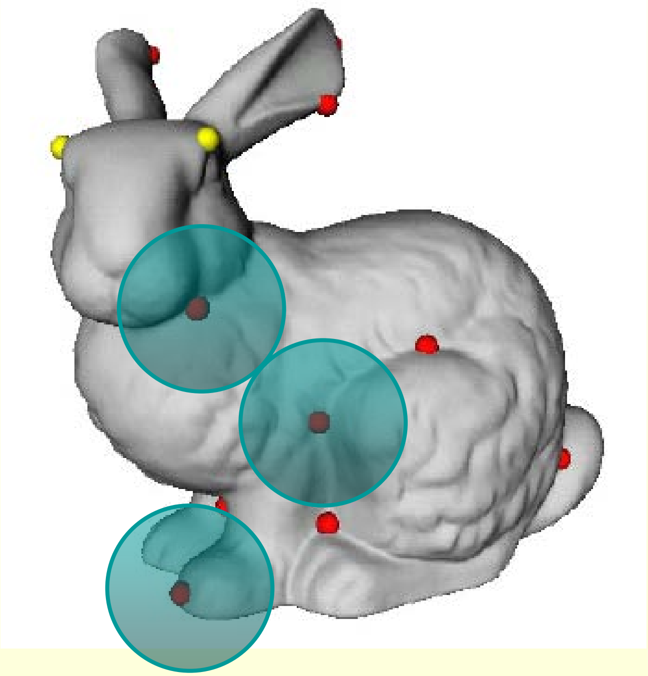


Features

Multiscale Algorithm

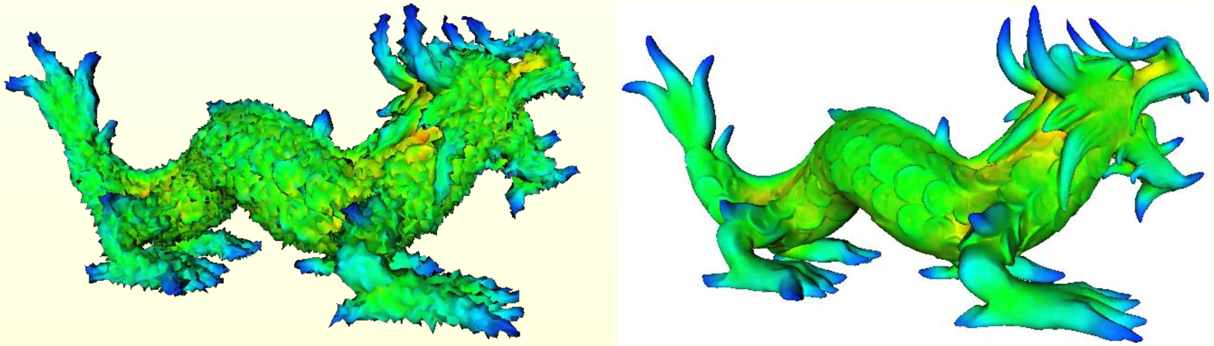
- Features should be persistent over scale change

r=0.5	r=1	r=2	r=4	r=8
Y	Y	Y	N	N
N	N	Y	Y	Y
N	N	Y	N	N



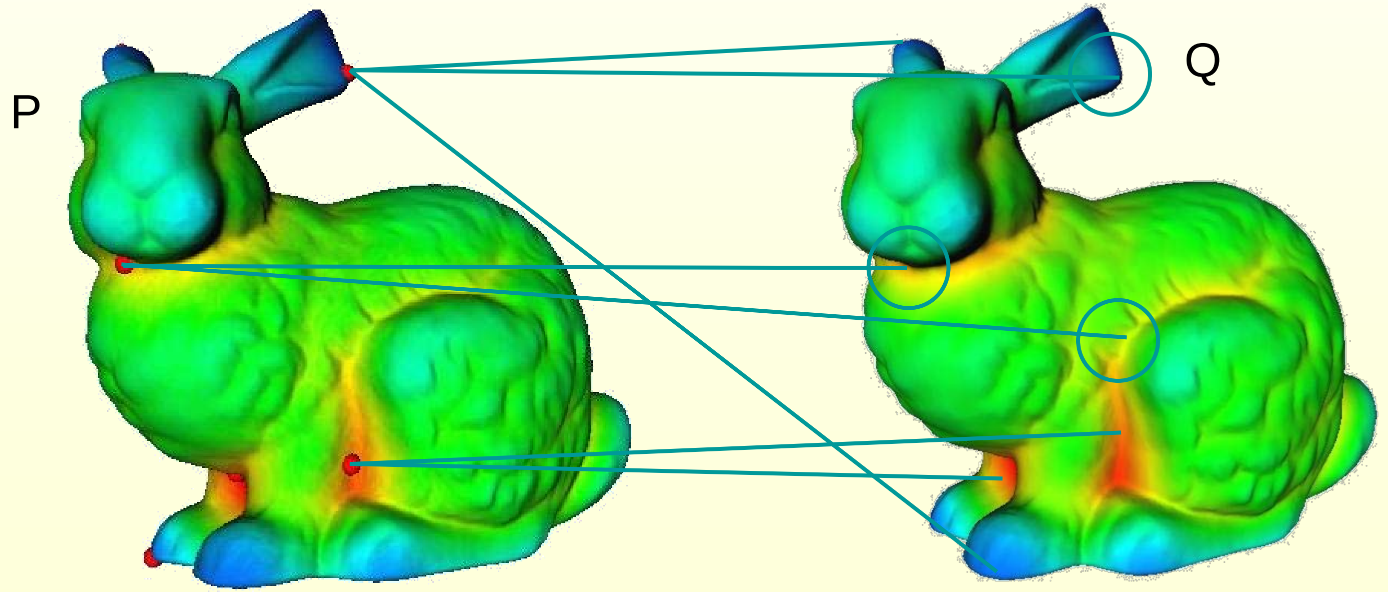
Feature Properties

- Sparse
- Robust to noise



Correspondence Space

- Search the whole model for correspondences
  - Range query for descriptor values
  - Cluster and pick representatives



Evaluating Correspondences

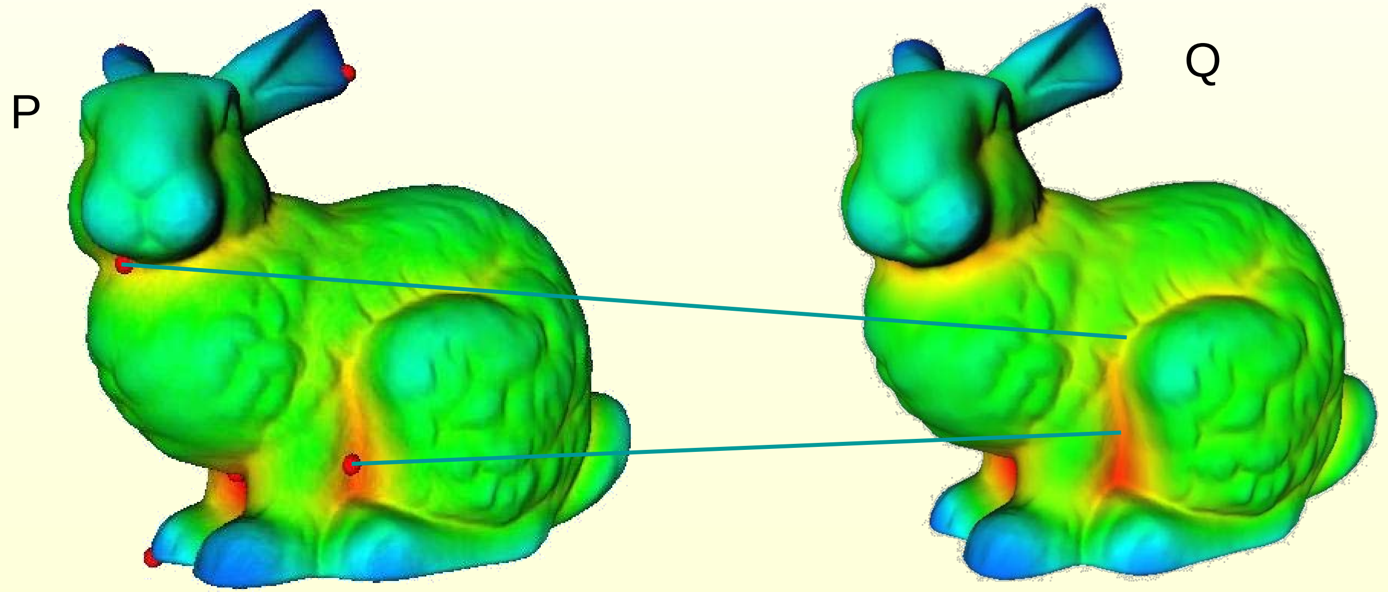
- *Coordinate* root mean squared distance

$$\mathrm{cRMS}^2(\mathbf{P},\mathbf{Q}) = \min_{\mathbf{R},\mathbf{t}} \frac{1}{n} \sum_{i=1}^{n} ||\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i||^2$$

  - Requires best aligning transform
  - Looks at correspondences individually

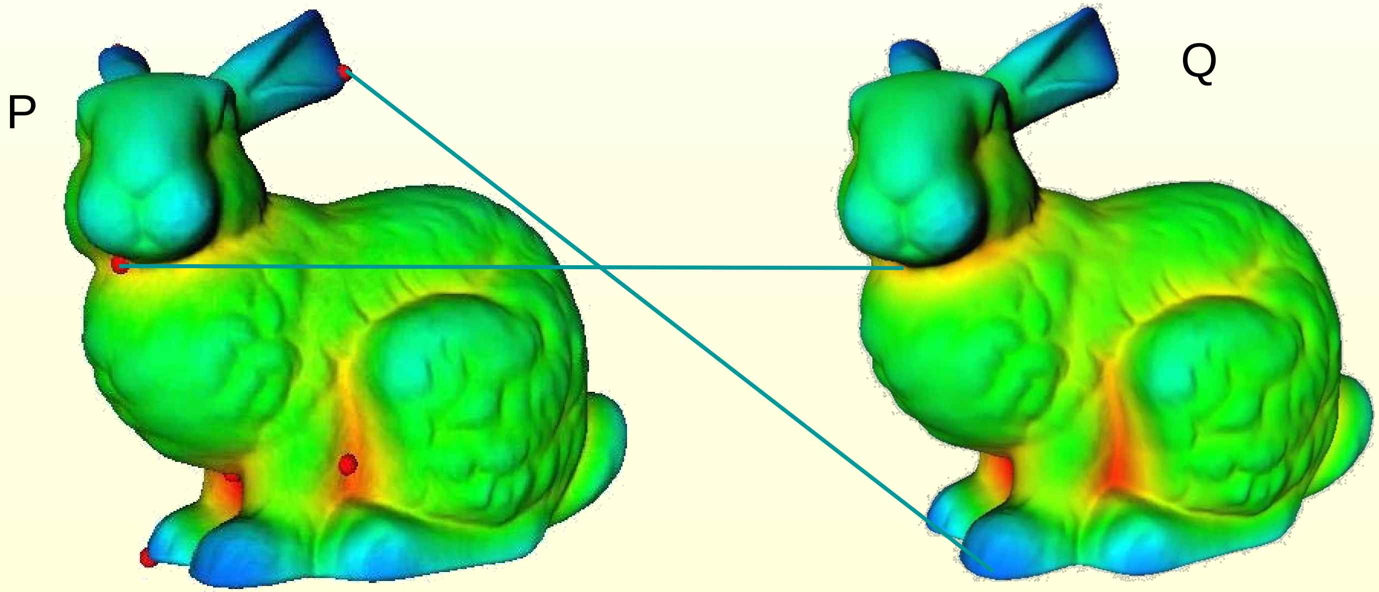
Rigidity Constraint

- Pair-wise distances between features and correspondences should be the same



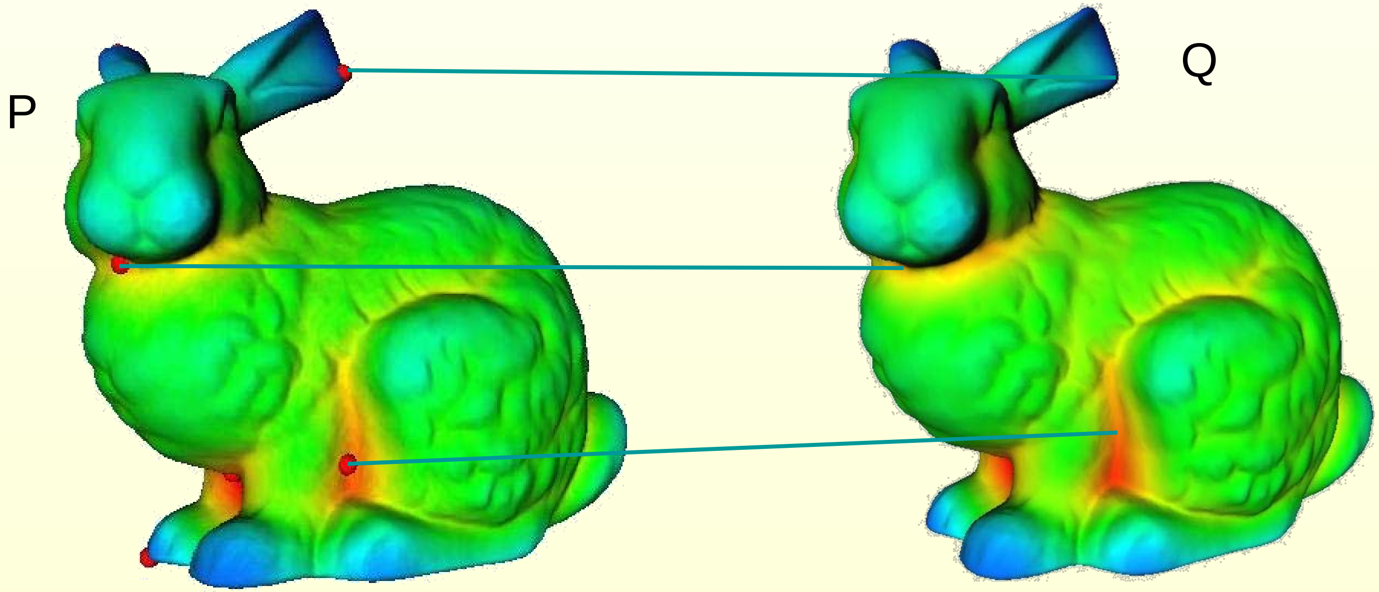
Rigidity Constraint

- Pair-wise distances between features and correspondences should be the same



Rigidity Constraint

- Pair-wise distances between features and correspondences should be the same



Evaluating Correspondences

- *Distance* root mean squared distance

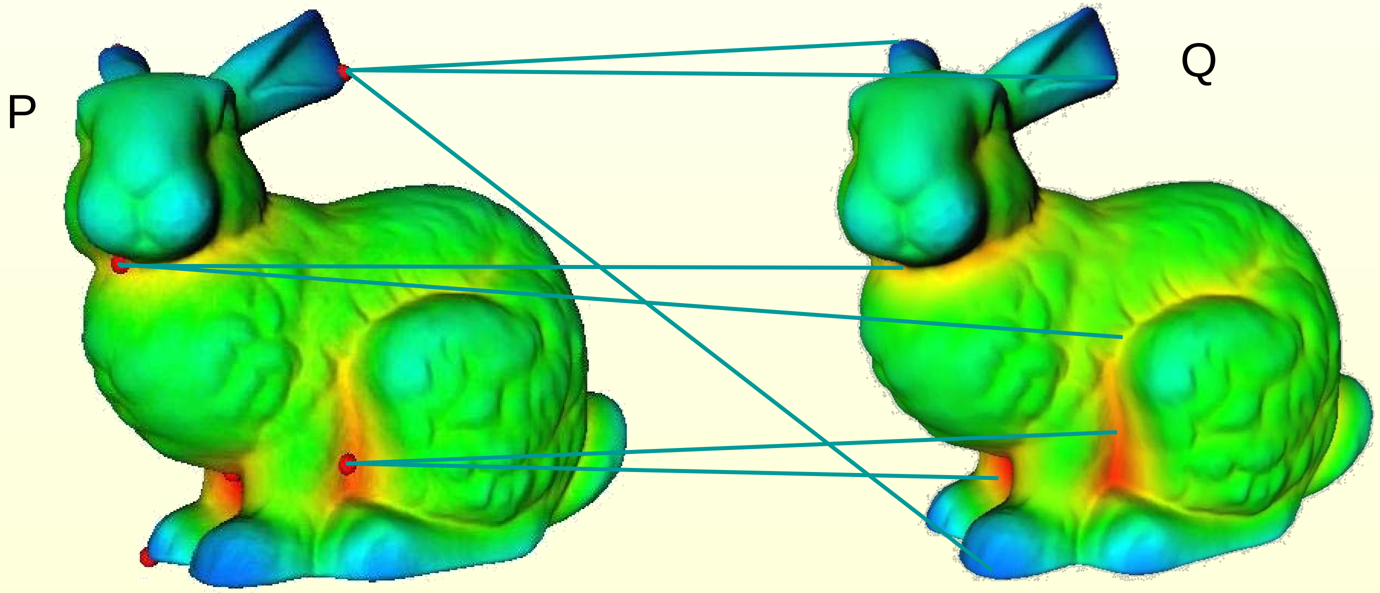
$$\mathrm{dRMS}^2(\mathbf{P},\mathbf{Q}) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(||\mathbf{p}_i - \mathbf{p}_j|| - ||\mathbf{q}_i - \mathbf{q}_j||)^2$$

- Depends only on internal distance matrix

$$\frac{1}{k\sqrt{n}}\mathrm{cRMS}(\mathbf{P},\mathbf{Q}) \leq \mathrm{dRMS}(\mathbf{P},\mathbf{Q}) \leq \sqrt{2}\ \mathrm{cRMS}(\mathbf{P},\mathbf{Q})$$

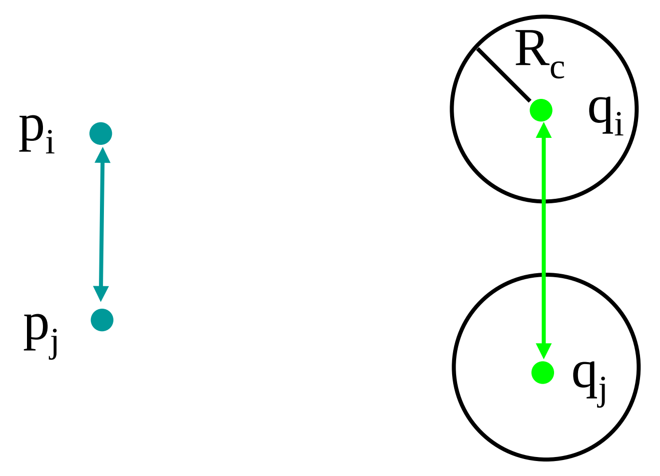
Search Algorithm

- Few features, each with few potential correspondences
  - Minimize dRMS
  - Exhaustive search still too expensive



Search Algorithm

- Branch and bound
  - Initial bound using greedy assignment
- Discard partial correspondences that fail thresholding test
  - $$\Big| ||\mathbf{p}_i - \mathbf{p}_j|| - ||\mathbf{q}_i - \mathbf{q}_j|| \Big| < 2R_c$$

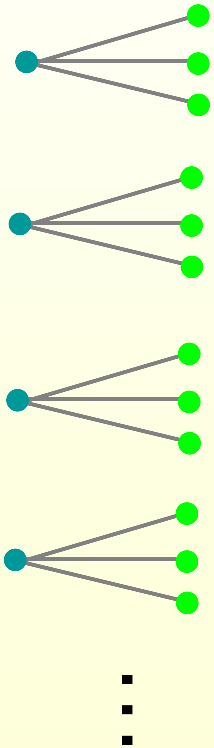


Search Algorithm

- Branch and bound
  - Initial bound using greedy assignment
- Discard partial correspondences that fail thresholding test
  - $\Big| ||\mathbf{p}_i - \mathbf{p}_j|| - ||\mathbf{q}_i - \mathbf{q}_j|| \Big| < 2R_c$
- Prune if partial correspondence exceeds bound
  - Spaced out features make incorrect correspondences fail quickly
- Since we explore the entire search space, we are guaranteed to find optimal alignment
  - Up to cluster size

Greedy Initialization

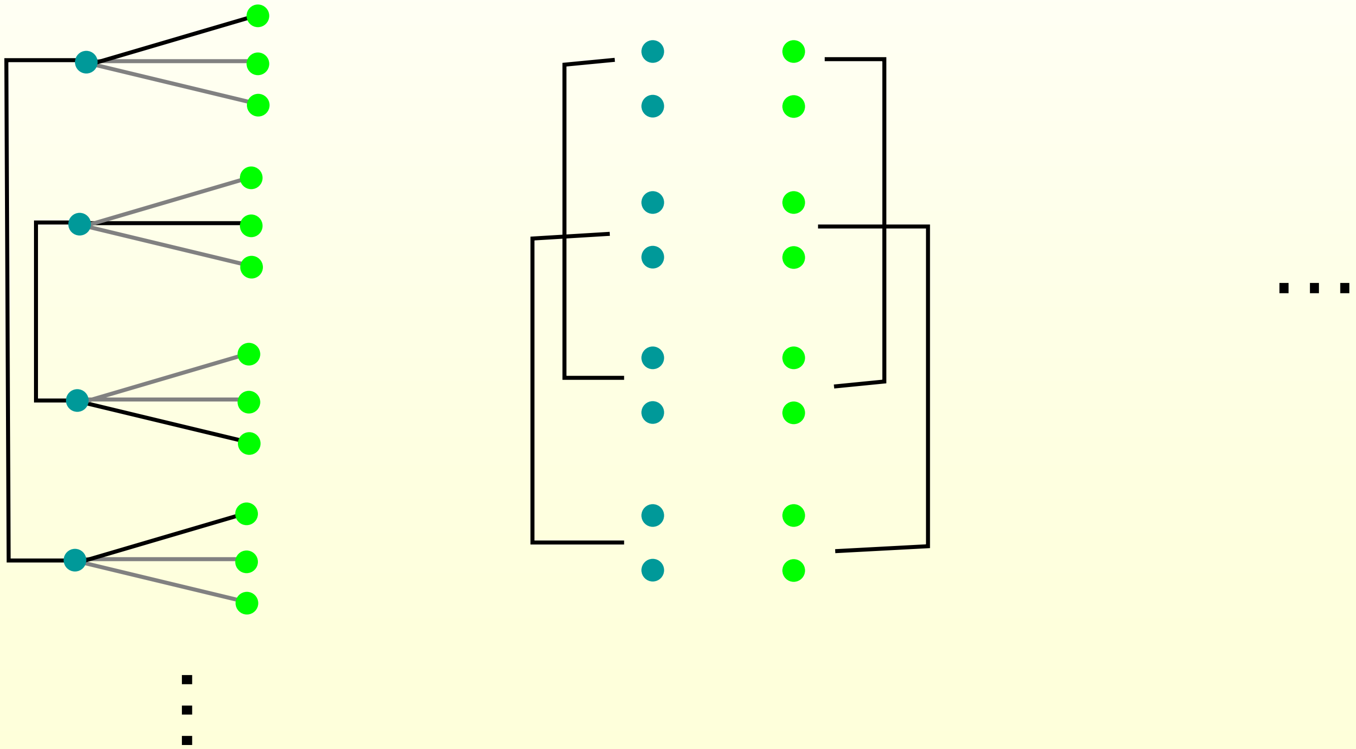
- Good initial bound is essential
- Build up correspondence set hierarchically



$$\min \left| ||\mathbf{p}_i - \mathbf{p}_j|| - ||\mathbf{q}_i - \mathbf{q}_j|| \right|$$

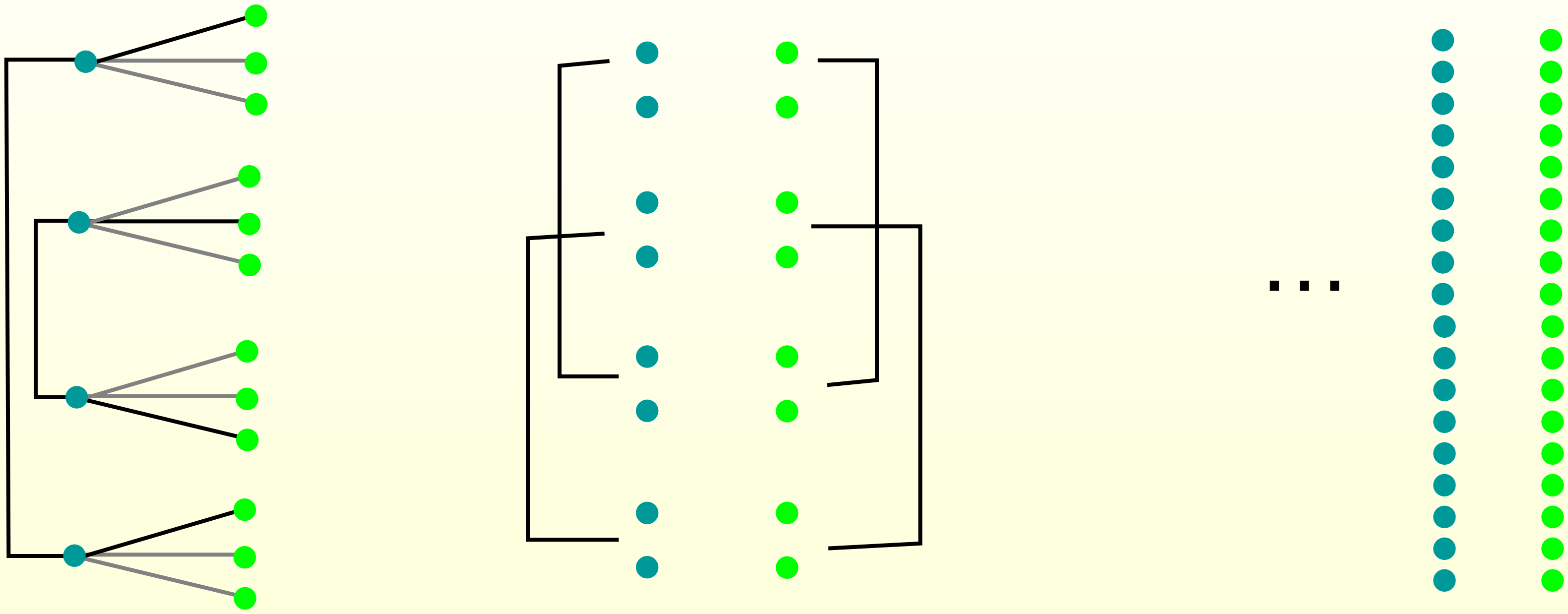
Greedy Initialization

- Good initial bound is essential
- Build up correspondence set hierarchically



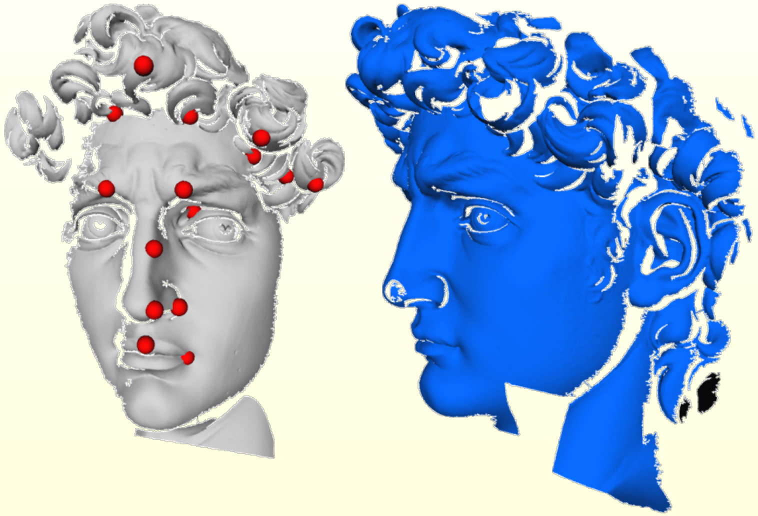
Greedy Initialization

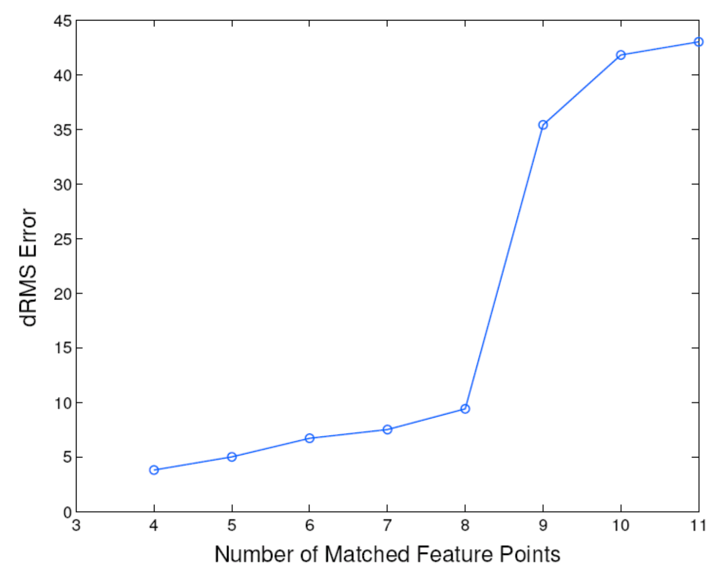
- Good initial bound is essential
- Build up correspondence set hierarchically



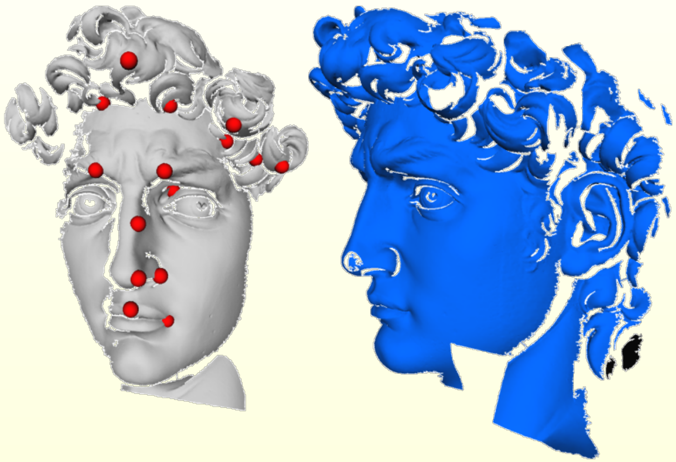
Partial Alignment

- Allow null correspondences, while maximizing the number of matched points





Alignment Results

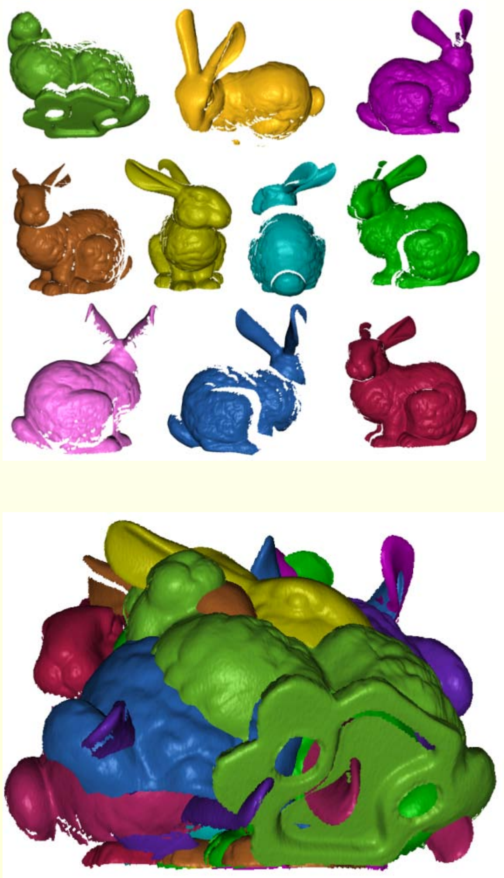


Input: 2 scans

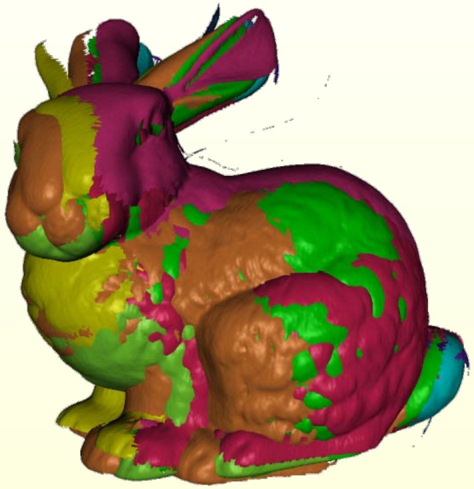
Feature-based alignment

Refined by ICP

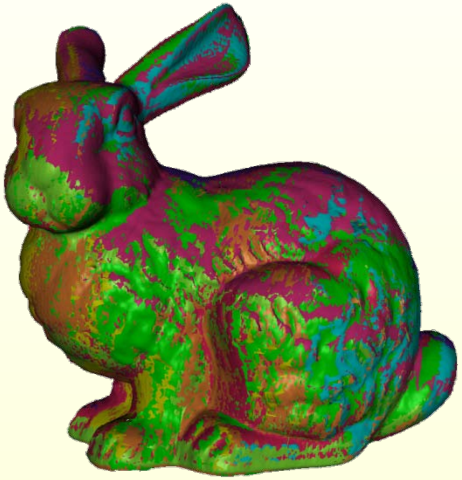
Alignment Results



Input: 10 scans



Alignment



Refined by ICP

The End