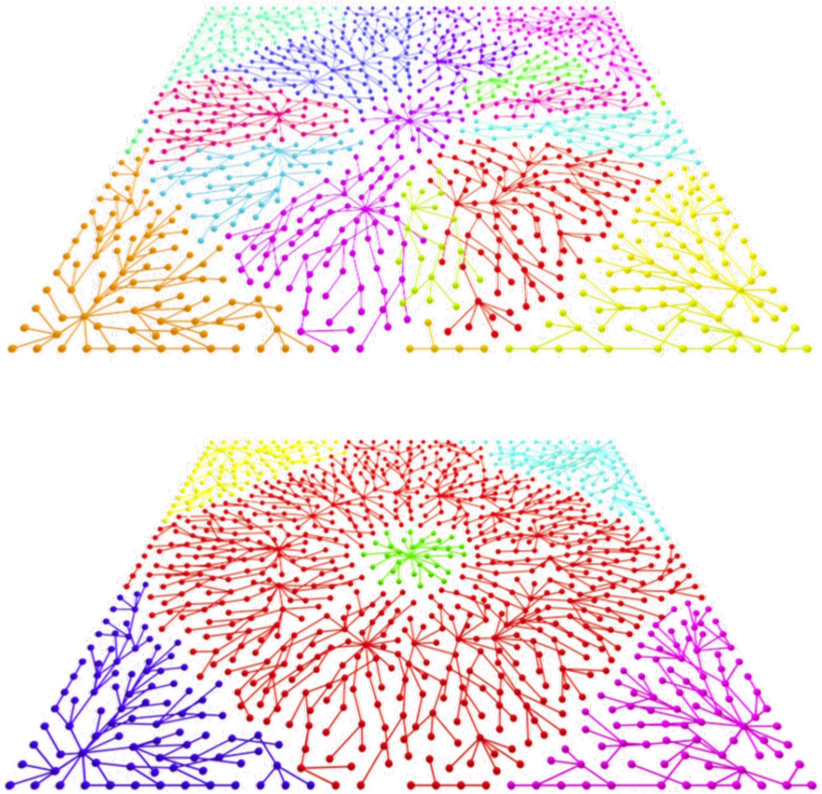
CS233, CME251: Geometric and Topological Data Analysis

Leonidas Guibas
Computer Science Department
Stanford University



Lecture 12
13 May 2020

Last Time: Differential Geometry of Surfaces and the Laplace-Beltrami Operator

Surface Curvatures

- **Principal curvatures**
  - Minimal curvature
  - Maximal curvature

$$\kappa_1 = \kappa_{\min} = \min_{\varphi} \kappa_n(\varphi)$$

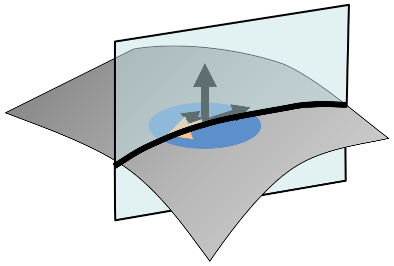
$$\kappa_2 = \kappa_{\max} = \max_{\varphi} \kappa_n(\varphi)$$

- **Mean curvature**

$$H = \frac{\kappa_1 + \kappa_2}{2} = \frac{1}{2\pi} \int_0^{2\pi} \kappa_n(\varphi) d\varphi$$

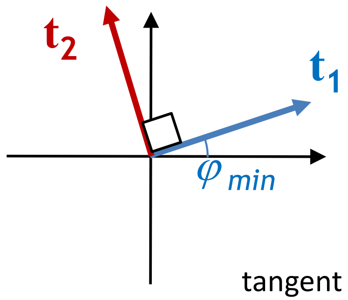
- **Gaussian curvature**

$$K = \kappa_1 \cdot \kappa_2$$

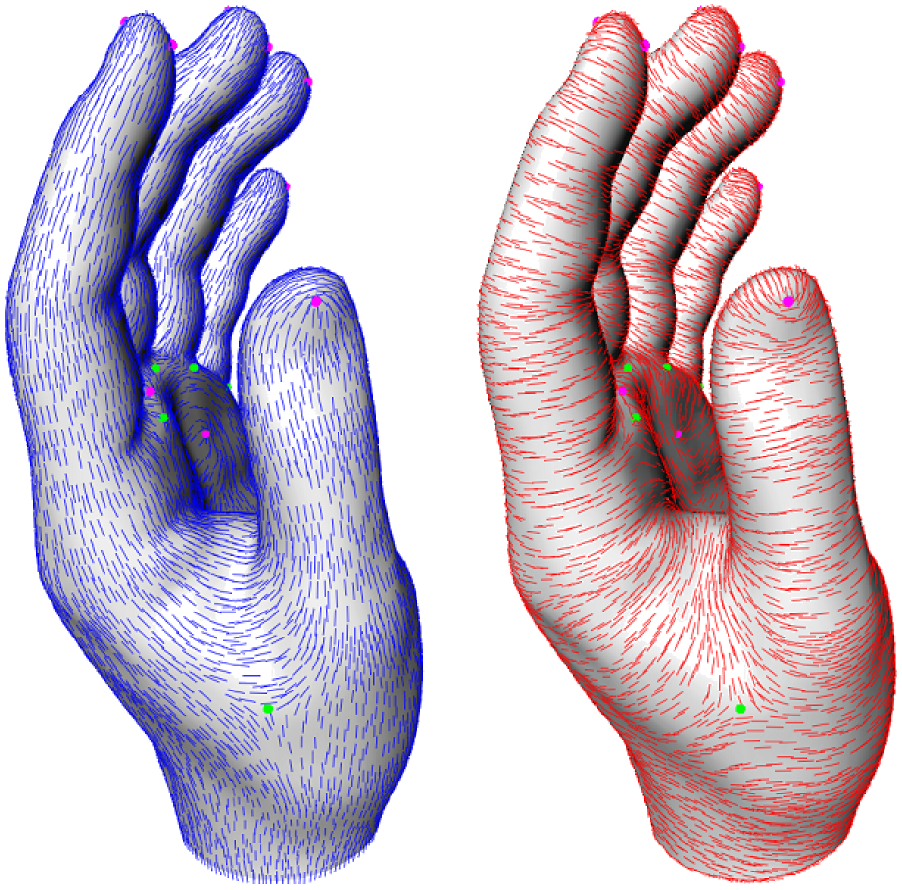


Principal Directions

- Principal directions: tangent vectors corresponding to φ_{max} and φ_{min}



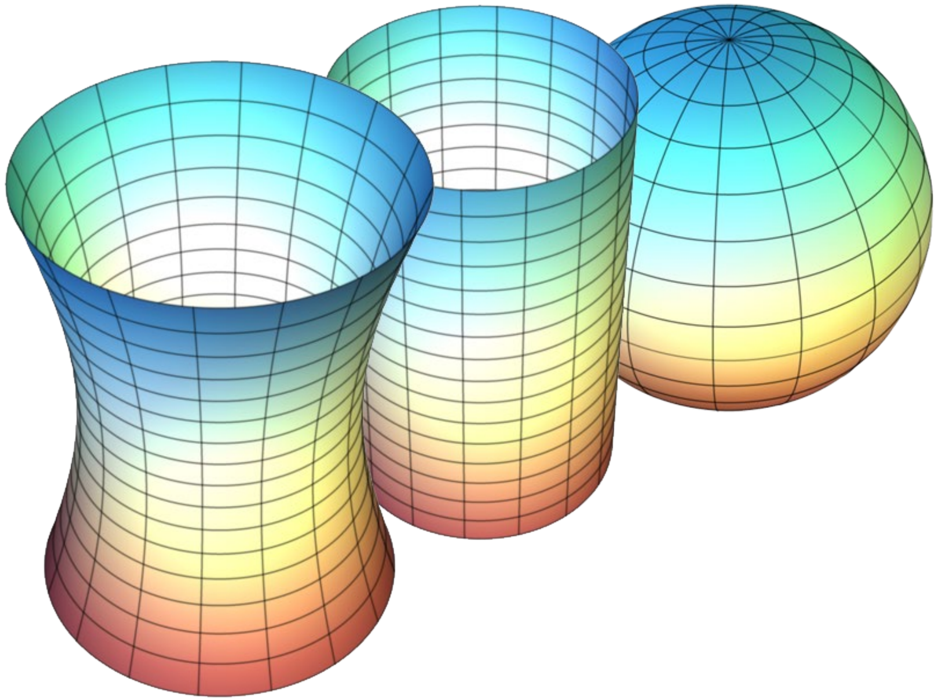
tangent plane



min curvature   max curvature

Classification

- A point $\mathbf{p}$ on the surface is called
  - Elliptic, if $K > 0$
  - Parabolic, if $K = 0$
  - Hyperbolic, if $K < 0$



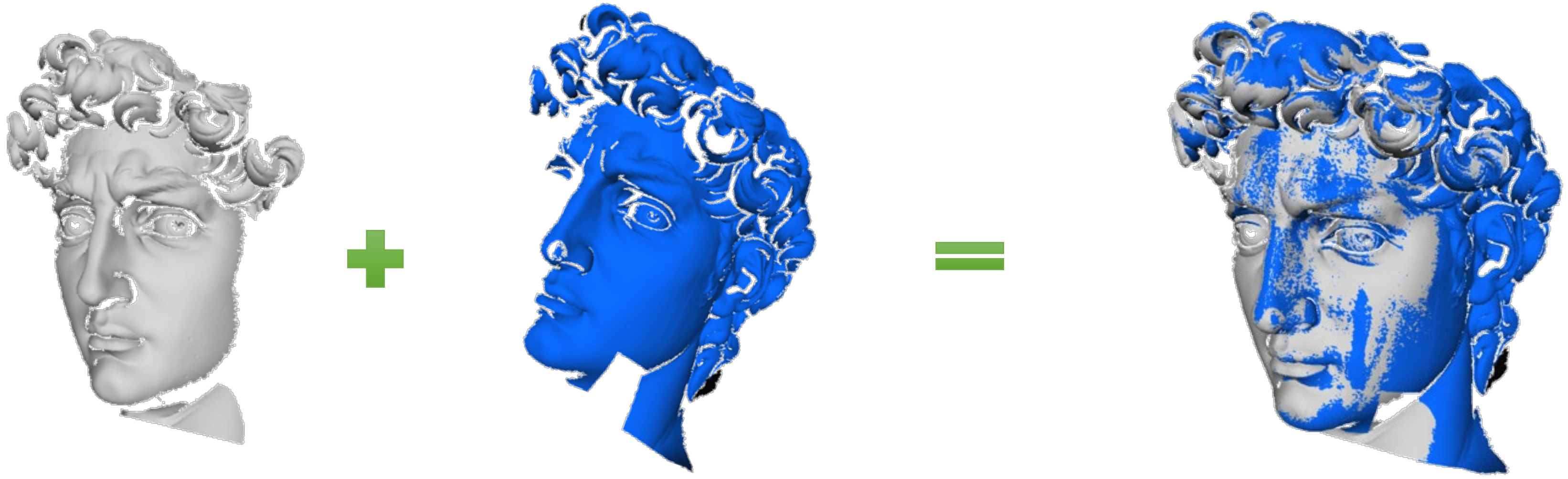
- Developable surface iff $K = 0$
  - can be mapped to the plane without distortion

Shape Features, Alignments, and Correspondences

The "wisdom of the collection" – joint data analysis

Rigid Alignment

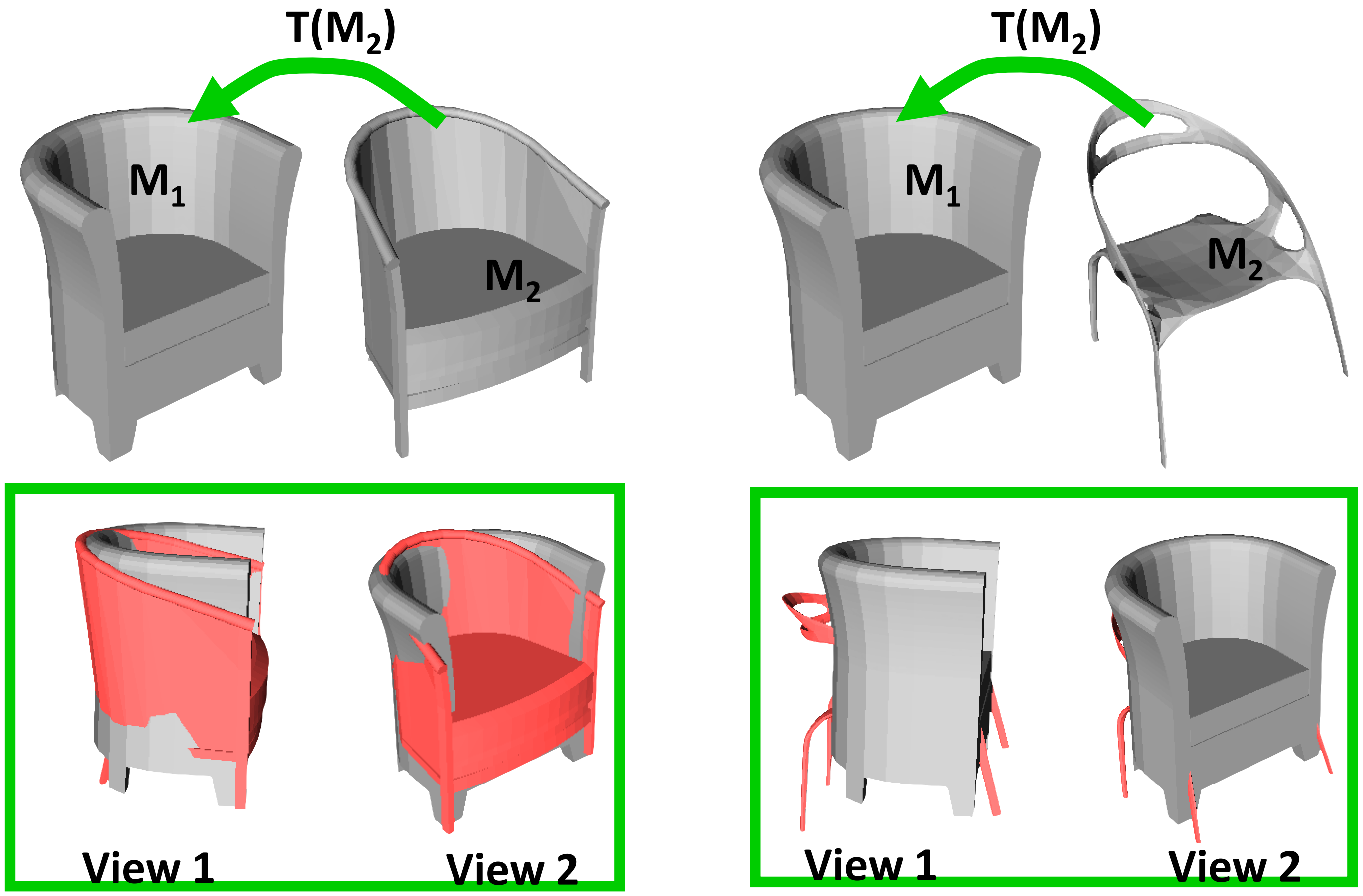
Fundamental Registration Problem



Given two shapes with partially overlapping geometry, find an alignment between them

General Rigid Alignment

Search for a rigid motion that best aligns the shapes, even if the shapes are different



$$M_1 \approx T(M_2)$$
$$T : \text{translation} + \text{rotation}$$

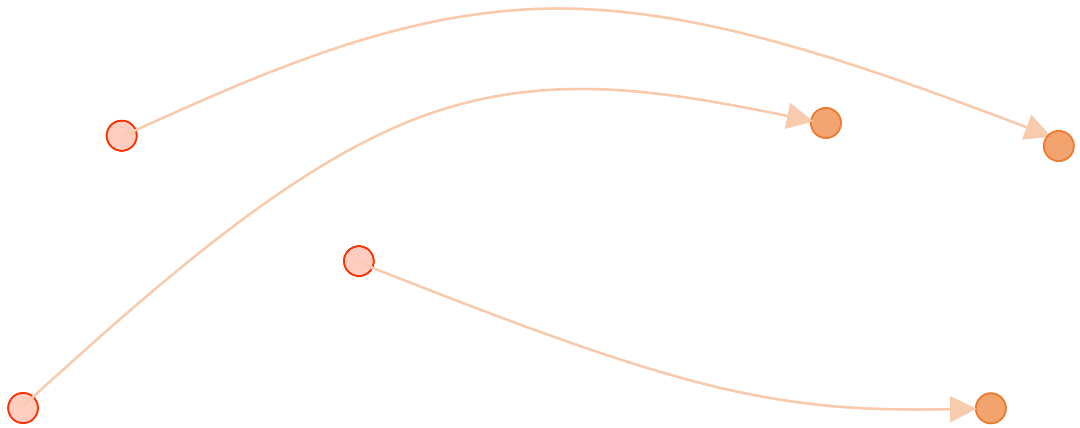
Optimal Rigid Alignment for Points

Problem Formulation:

1. Given two sets points: $\{x_i\}, \{y_i\}, i = 1..n$ in $\mathbb{R}^3$. Find the rigid transform:

   $\mathbf{R}, t$ that minimizes:

$$\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$$



Simplest Case: Rigid Alignment, Given Correspondences

- We are given two sets of corresponding points $x_1, x_2, \ldots, x_n$ and $y_1, y_2, \ldots, y_n$ in $\Re^3$. We wish to compute the rigid transform T that best aligns x_1 to y_1, x_2 to y_2, ..., and x_n to y_n.
- We define the error to be minimized by

$$\min_T \sum_{i=1}^{n} ||T(x_i) - y_i||^2$$

MSE error, RMS distance, ...

- Old Problem:
  - Known and solved as the *orthogonal Procrustes problem* in Factor Analysis (Statistics) [Shönemann, 1966]
  - Known and solved as the *absolute orientation problem* in Photogrammetry [Horn, 1986]
  - Also in robotics, graphics, medical image analysis, statistical theories of shape, etc ...

The Optimization

- A rigid motion T is a combination of a translation a and a rotation R, so that $T(x) = R(x) + a$.
- The quantity to be minimized is:

$$\min_{a,R}\left(\sum_{i=1}^{n}|R(x_i)+a-y_i|^2\right)$$

The unknowns

Separability

- A rigid motion *T* is a combination of a translation *a* and a rotation *R*, so that *T*(*x*) = *R*(*x*) + *a*.
- If we place the origin of our coordinate system at the mean of the x_i's, then the quantity to be minimized simplifies to (up to some constants):

$$\min_{a,R}\left(\sum_{i=1}^{n}|y_i - a|^2 - 2\sum_{i=1}^{n}\langle R(x_i), y_i\rangle\right)$$

- Note that the translational and rotational parts separate. The translational part *a* can easily be seen to be optimized by

$$a = \frac{1}{n}\sum_{i=1}^{n} y_i$$

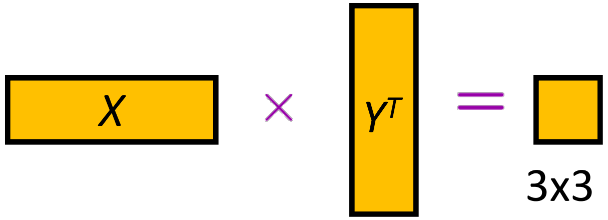
The centroids of the two point sets have to be aligned!

The Rotation Part via SVD

- Define

$$
\begin{aligned}
\overline{x} &= \frac{1}{n}\sum_{i=1}^{n} x_i \\
\overline{y} &= \frac{1}{n}\sum_{i=1}^{n} y_i \\
X &= [x_1 - \overline{x}, \ldots x_n - \overline{x}]^T \\
Y &= [y_1 - \overline{y}, \ldots y_n - \overline{y}]^T
\end{aligned}
$$

- Here X and Y are 3 by n matrices.



*SVD = singular value decomposition

- Now compute the SVD*

$$XY^T = UDV^T \quad (3 \times 3)$$

- U and V are 3 by 3 orthogonal matrices, and D is a diagonal matrix with decreasing non-negative entries along the diagonal (the singular values).

- Define S by

$$S = \begin{cases} I, & \text{if } \det U \det V = 1 \\ \text{diag}(1, \ldots, 1, -1), & \text{otherwise} \end{cases}$$

- Then $$R = USV^T$$

$O(n)$ algorithm!

Optimal Transformation Summary

Problem Formulation:

1. Given two sets points: $\{x_i\}, \{y_i\}, i = 1..n$ in $\mathbb{R}^3$. Find the rigid transform: $\mathbf{R}, t$ that minimizes:
$$\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$$
2. Closed form solution:
   1. Construct: $C = \sum_{i=1}^{N}(y_i - \mu^Y)(x_i - \mu^X)^T$, where $\mu^X = \frac{1}{N}\sum_i x_i,$ $\mu^Y = \frac{1}{N}\sum_i y_i$
   2. Compute the SVD of C: $C = U\Sigma V^T$
      1. If $\det(UV^T) = 1, R_{\text{opt}} = UV^T$
      2. Else $R_{\text{opt}} = U\tilde{\Sigma}V^T, \tilde{\Sigma} = \text{diag}(1, 1, \ldots, -1)$
   3. Set $t_{\text{opt}} = \mu^Y - R_{\text{opt}}\mu^X$

Note that C is a 3x3 matrix. SVD is very fast.

Arun et al., Least-Squares Fitting of Two 3-D Point Sets

How to Get Correspondences?

A chicken-and-egg problem: if we knew the optimal aligning transform, then we could get correspondences by proximity (possibly with the aid of some global adjustment, e.g., dynamic programming)

Transform ⟷ Correspondences

Guess one, estimate the other, and *iterate!*

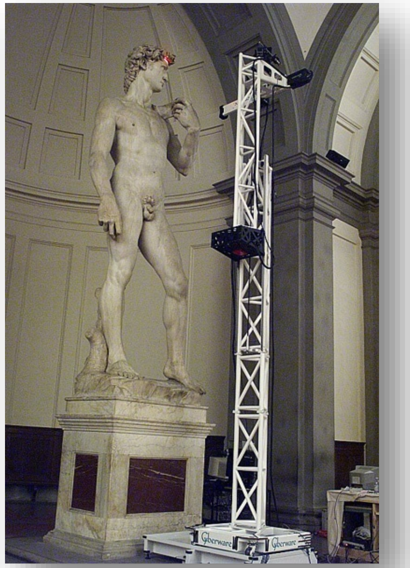
EM like

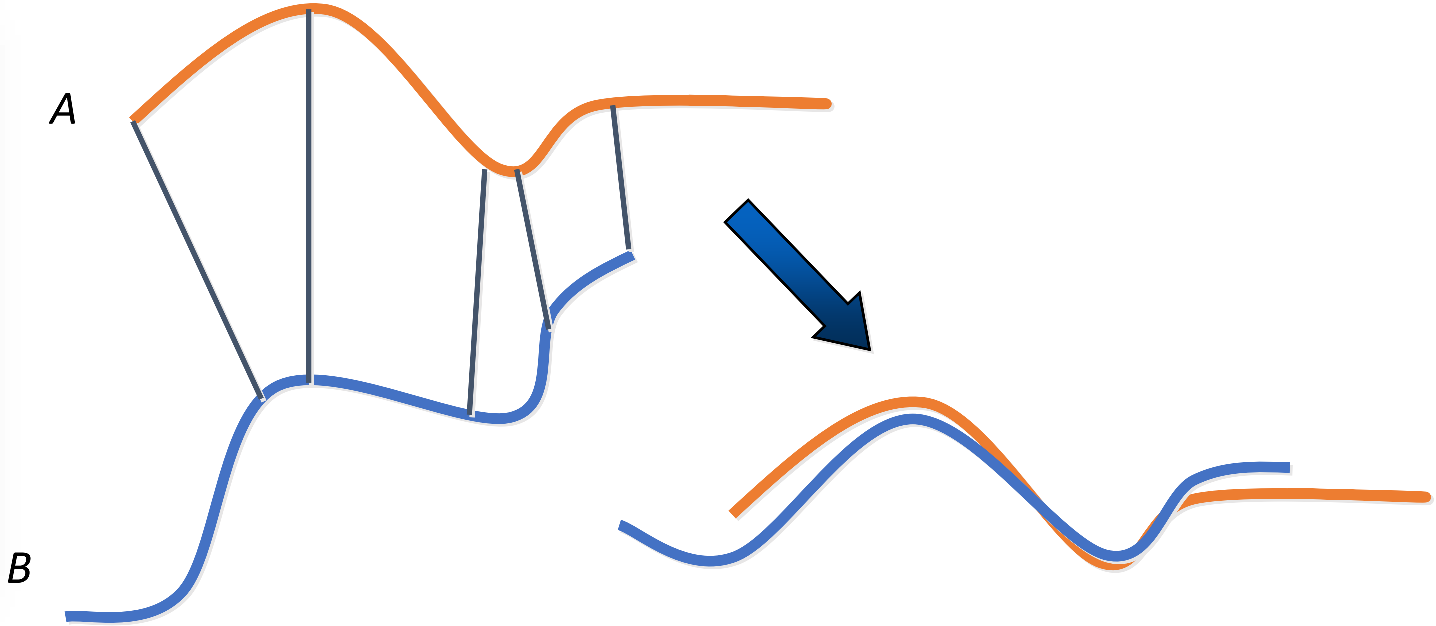
- Correspondences from proximity (Iterated Closest Pair)
- Correspondences from local shape descriptors (Shape Features)
- Transform from voting schemes (RANSAC, Geometric Hashing)
- Combinations

Local Methods:
Iterated Closest Pair (ICP)
[Besl, McKay 1992]

Aligning 3D Data

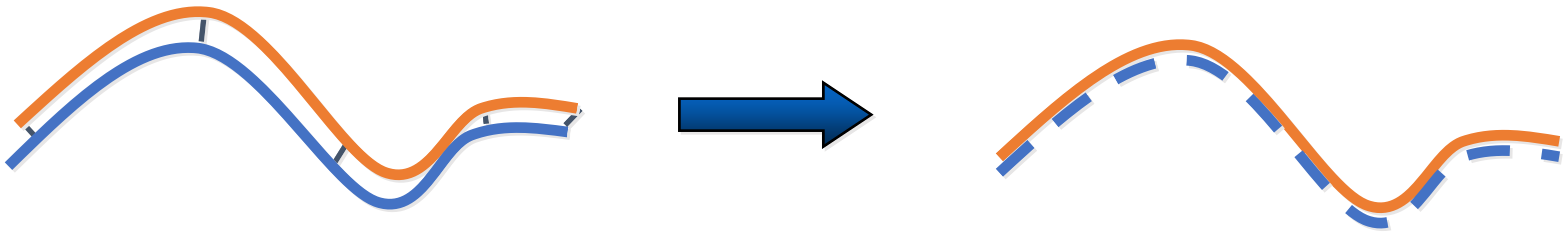
- How to find correspondences: User input? Feature detection? Local shape signatures?
- When *A* and *B* are partial scans of the same stationary object captured in a consistent setting, use the simplest alternative: assume closest points correspond

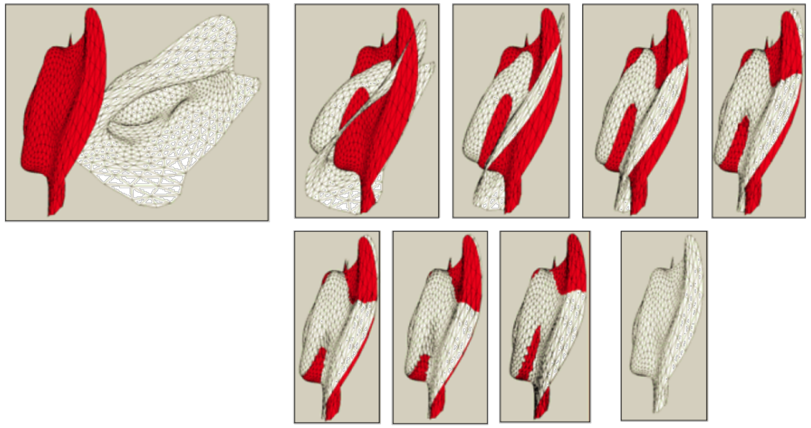




Aligning 3D Data

- Align the *A* points to their closest *B* neighbors, then repeat
- Converges, if starting positions are "close enough"

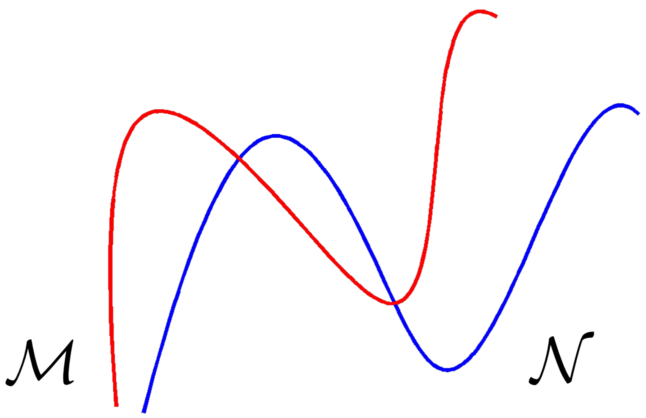




Successive iterations bring the objects closer together

Iterative Closest Point (ICP)

- Classical approach: iterate between finding correspondences and finding the transformation:



example in 2D

Given a pair of shapes, $\mathcal{M}$ and $\mathcal{N}$, iterate:

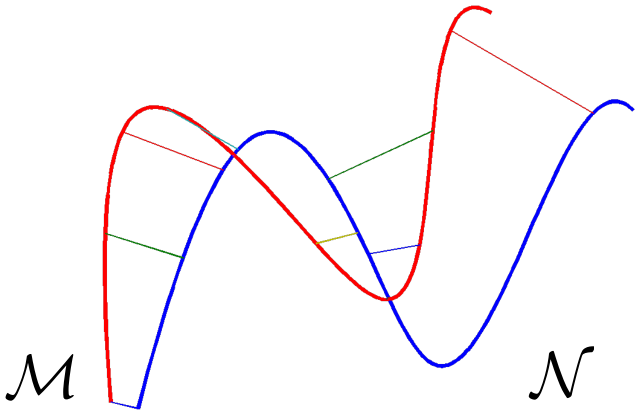
1. For each $x_i \in \mathcal{M}$ find **nearest** neighbor $y_i \in \mathcal{N}$
2. Find optimal transformation $\mathbf{R}, t$ minimizing:

$$\arg\min_{R,t} \sum_i \|Rx_i + t - y_i\|_2^2$$

Classic problem, solvable by SVD

Iterative Closest Point (ICP)

- Classical approach: iterate between finding correspondences and finding the transformation:



example in 2D

Given a pair of shapes, $\mathcal{M}$ and $\mathcal{N}$, iterate:

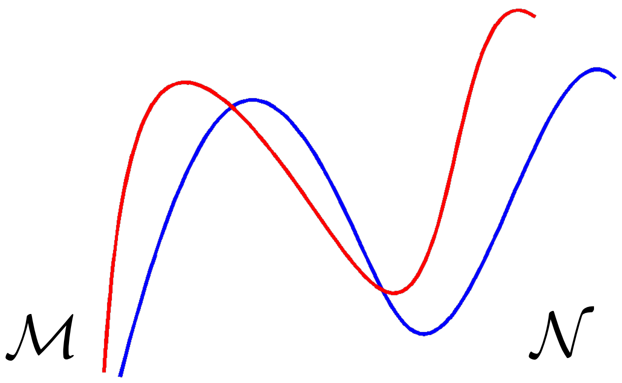
1. For each $x_i \in \mathcal{M}$ find **nearest** neighbor $y_i \in \mathcal{N}$
2. Find optimal transformation $\mathbf{R}, t$ minimizing:

$$\arg\min_{R,t} \sum_i \|Rx_i + t - y_i\|_2^2$$

Classic problem, solvable by SVD

Iterative Closest Point (ICP)

- Classical approach: iterate between finding correspondences and finding the transformation:



example in 2D

Given a pair of shapes, $\mathcal{M}$ and $\mathcal{N}$, iterate:

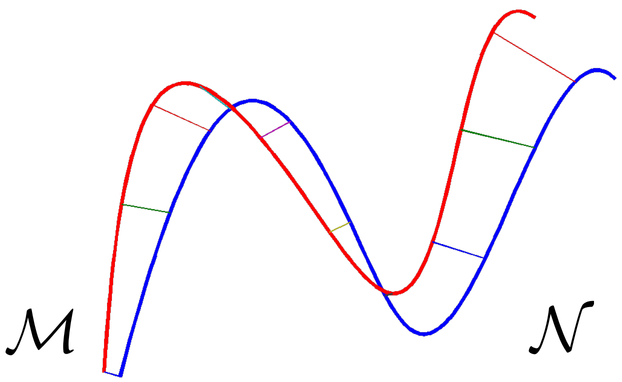
1. For each $x_i \in \mathcal{M}$ find **nearest** neighbor $y_i \in \mathcal{N}$
2. Find optimal transformation $\mathbf{R}, t$ minimizing:

$$\arg\min_{R,t} \sum_i \|Rx_i + t - y_i\|_2^2$$

Classic problem, solvable by SVD

Iterative Closest Point (ICP)

- Classical approach: iterate between finding correspondences and finding the transformation:



example in 2D

Given a pair of shapes, $\mathcal{M}$ and $\mathcal{N}$, iterate:

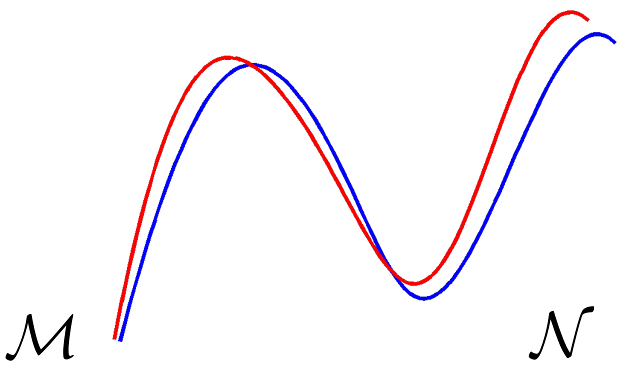
1. For each $x_i \in \mathcal{M}$ find **nearest** neighbor $y_i \in \mathcal{N}$
2. Find optimal transformation $\mathbf{R}, t$ minimizing:

$$\arg\min_{R,t} \sum_i \|Rx_i + t - y_i\|_2^2$$

Classic problem, solvable by SVD

Iterative Closest Point (ICP)

- Classical approach: iterate between finding correspondences and finding the transformation:



example in 2D

Given a pair of shapes, $\mathcal{M}$ and $\mathcal{N}$, iterate:

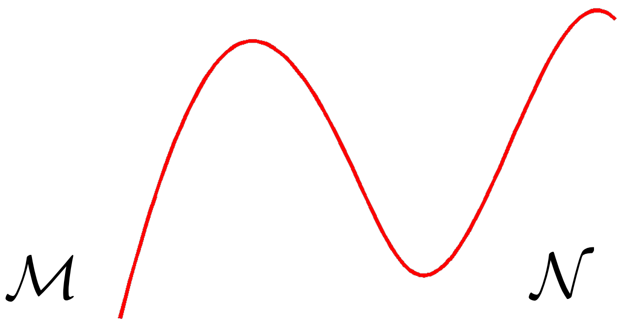
1. For each $x_i \in \mathcal{M}$ find **nearest** neighbor $y_i \in \mathcal{N}$
2. Find optimal transformation $\mathbf{R}, t$ minimizing:

$$\arg\min_{R,t} \sum_i \|Rx_i + t - y_i\|_2^2$$

Classic problem, solvable by SVD

Iterative Closest Point (ICP)

- Classical approach: iterate between finding correspondences and finding the transformation:



example in 2D

Given a pair of shapes, $\mathcal{M}$ and $\mathcal{N}$, iterate:

1. For each $x_i \in \mathcal{M}$ find **nearest** neighbor $y_i \in \mathcal{N}$
2. Find optimal transformation $\mathbf{R}, t$ minimizing:

$$\arg\min_{R,t} \sum_i \|Rx_i + t - y_i\|_2^2$$

Classic problem, solvable by SVD

Convergence Theorem

- The ICP algorithm always converges monotonically to a local minimum, with respect to the MSE distance objective function
  - Correspondence step improves error bound – because of nearest neighbor computation
  - Rotation/Translation step improves error bound – because of transform optimization

Time Analysis

Each iteration includes three main steps

A. Finding the closest points:

   $O(N_M)$ per point

   $O(N_M * N_S)$ total

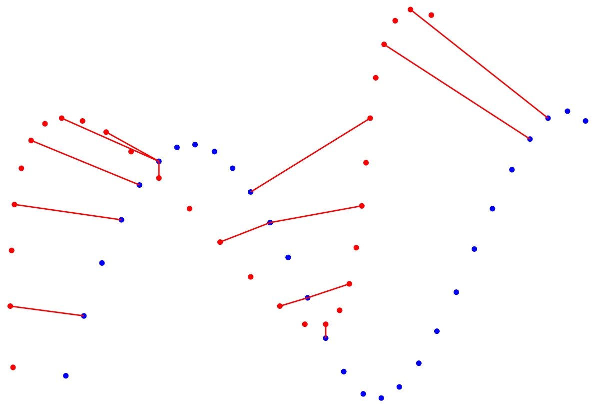
B. Calculating the optimal alignment: $O(N_S)$

C. Updating the scene: $O(N_S)$

Fast nearest-neighbor data structures can be very helpful here, e.g., a *k-d* tree

Variations of ICP

1. Selecting source points (from one or both scans): sampling
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. the transformation

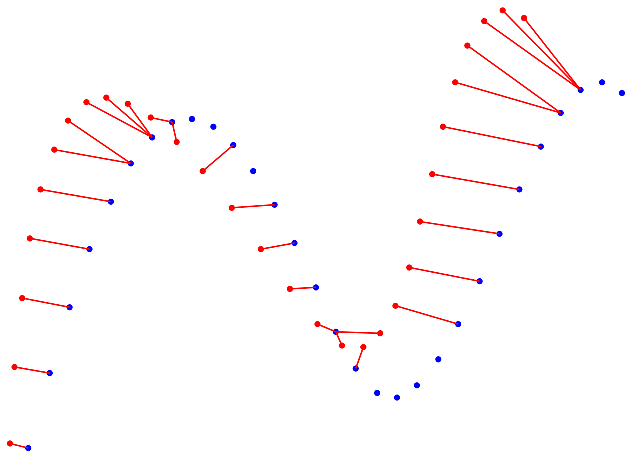


Iterative Closest Point

Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing:

$$\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$$



Ideally, most correspondences are 1-1

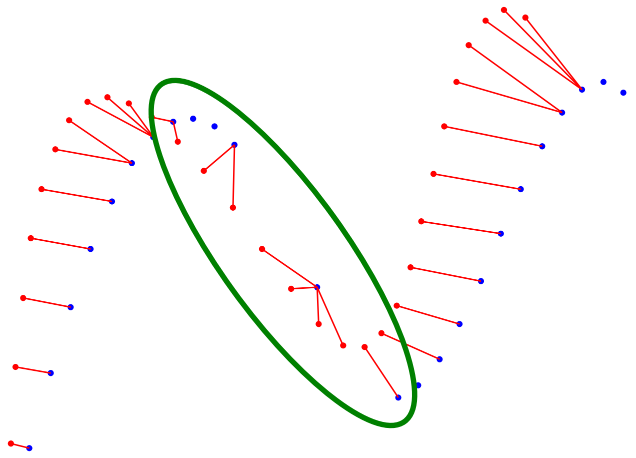
Iterative Closest Point

Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing:

$$\sum_{i=1}^{N} \|\mathbf{R}x_i + t - y_i\|_2^2$$

Problem:
uneven sampling



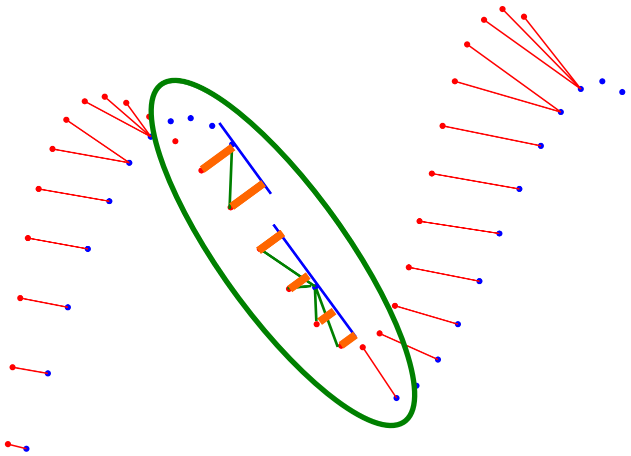
Iterative Closest Point

Given a pair of shapes, X and Y, iterate:

1. For each $x_i \in X$ find **nearest** neighbor $y_i \in Y$.
2. Find deformation $\mathbf{R}, t$ minimizing:

$$\sum_{i=1}^{N} d(\mathbf{R}x_i + t, P(y_i))^2 = \sum_{i=1}^{N} \left((\mathbf{R}x_i + t - y_i)^T \mathbf{n}_{y_i}\right)$$

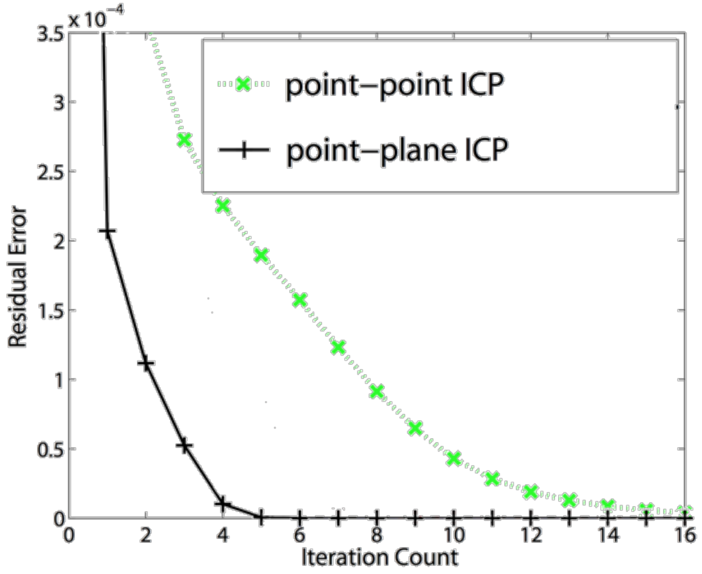
Solution:
Minimize distance to the tangent plane

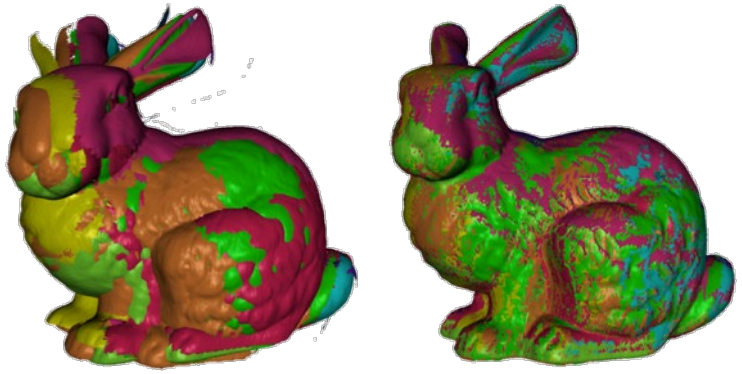


No longer a closed-form solution

Chen, Medioni, '91

Iterative Closest Point





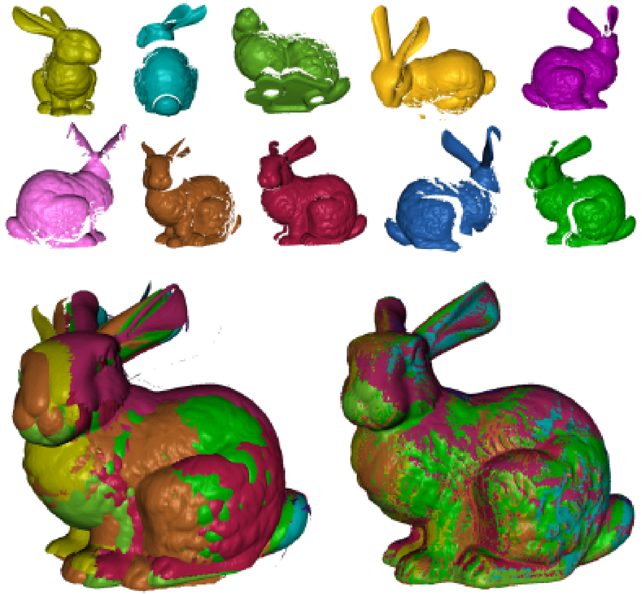
Aligning the bunny to itself:

Point-to-plane always wins in the end-game.

Global Matching Methods

Global Matching

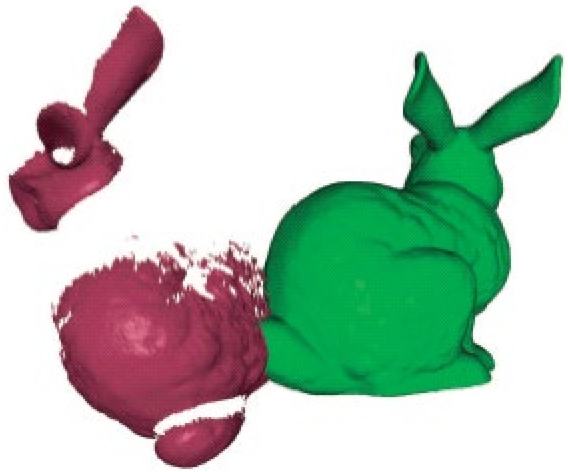
Given shapes in *arbitrary* positions, find their alignment:



Robust Global Registration
Gelfand et al. SGP 2005

Can be approximate, since will refine later using e.g. ICP

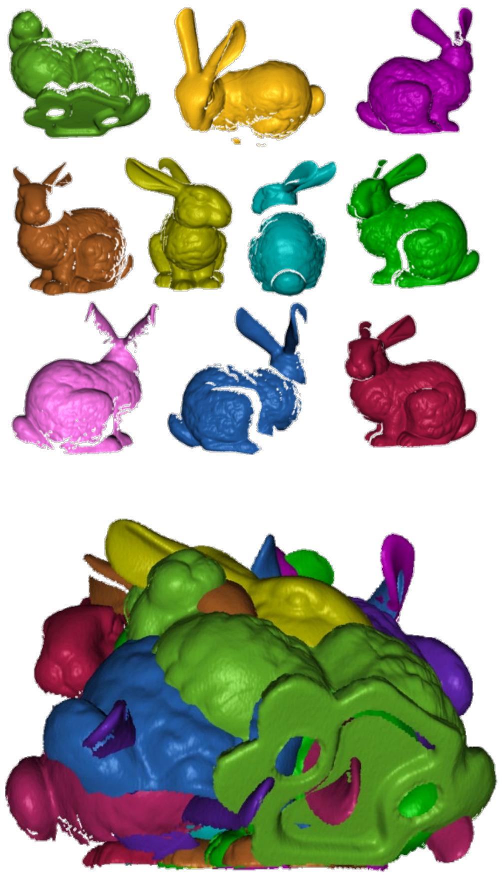
Partial Alignment



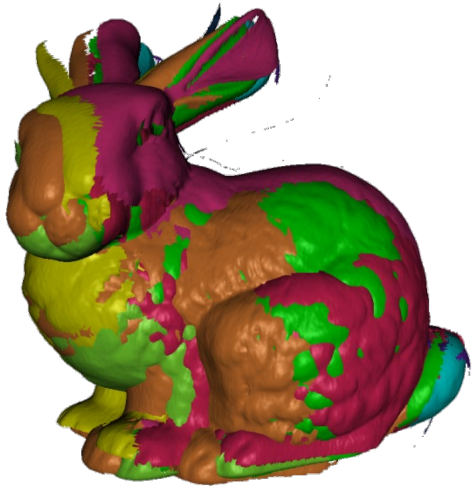
After 6 iterations

Multiple Alignment Results

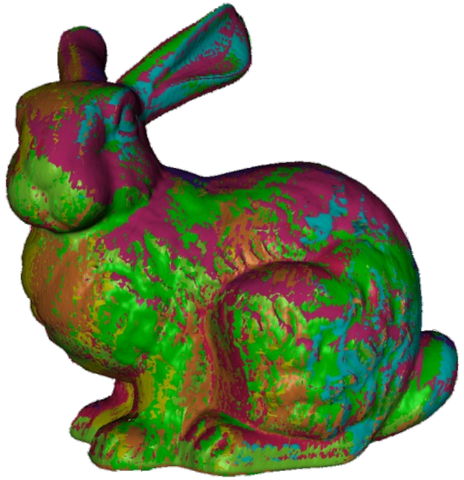
Bundle adjustment: build a scan graph
keep re-aligning pairs so as to reduce global error



Input: 10 scans



Approximate alignment



Refined by ICP

Global Matching – Approaches

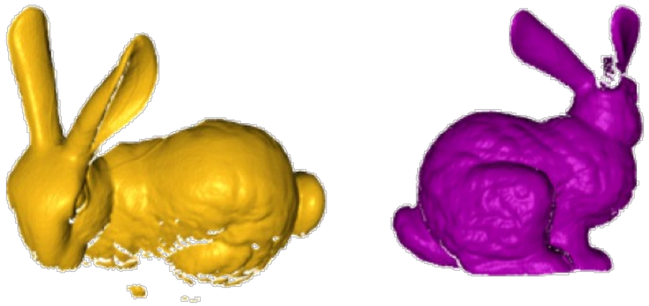
Several classes of approaches:

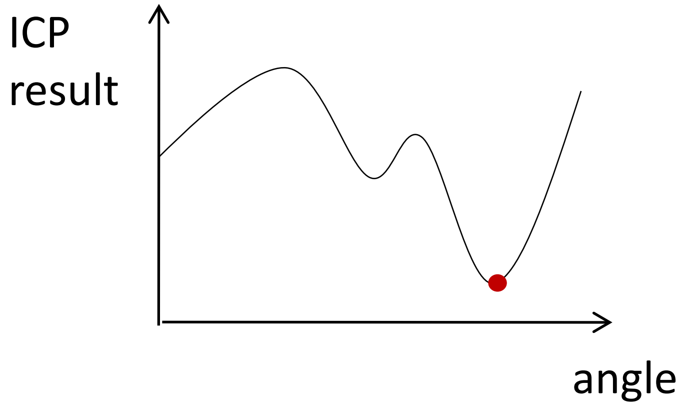
1. Exhaustive Search
2. Normalization
3. Random Sampling
4. Invariance

Exhaustive Search:

Compare (ideally) all alignments

- Sample the space of possible initial alignments.
- Correspondence is determined by the alignment at which models are closest.





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

Exhaustive Search:

Compare at all alignments

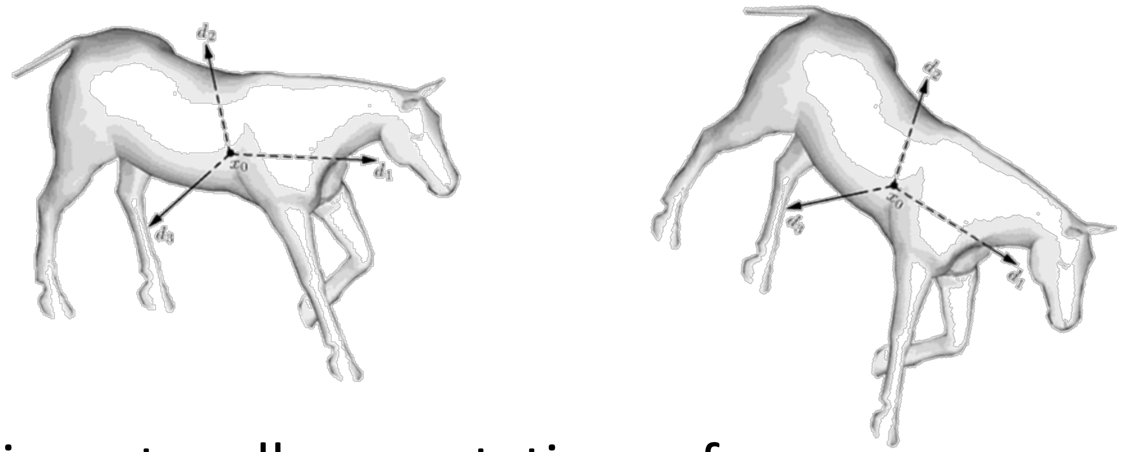
- Sample the space of possible initial alignments
- Correspondence is determined by the alignment at which models are closest

- Provides optimal result
- Can be unnecessarily slow
- Does not generalize well to non-rigid deformations

Normalization – Canonical Poses

There are only a handful of initial configurations that are important.

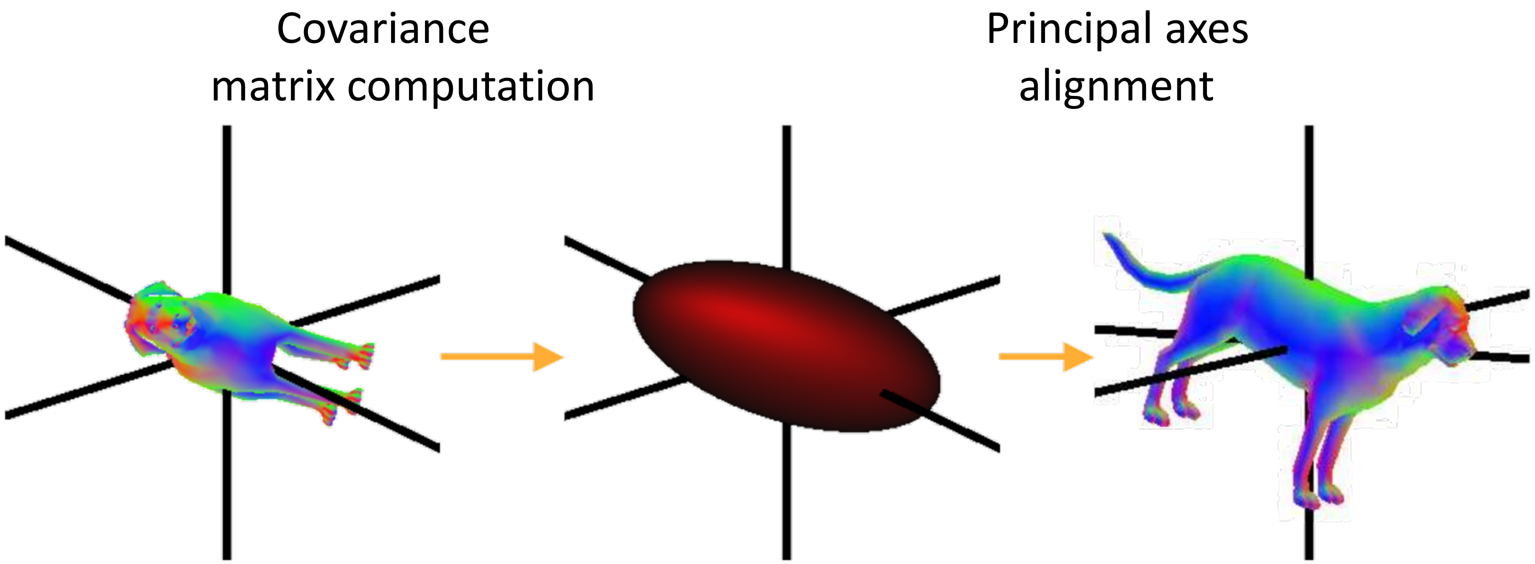
Can center all shapes at the origin and use PCA to find the principal directions of the shape.



In addition sometimes try all permutations of x-y-z.

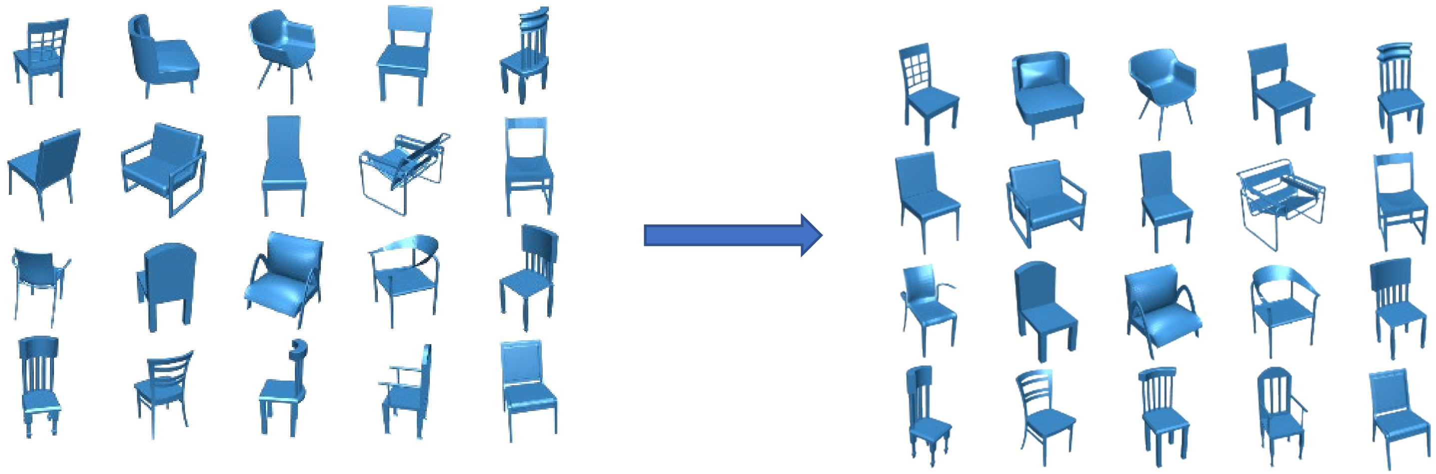
PCA-Based Alignment

- Use PCA to place models into canonical coordinate frames
- Then align those frames



Normalization – Canonical Poses

There are only a handful of initial configurations that are important.

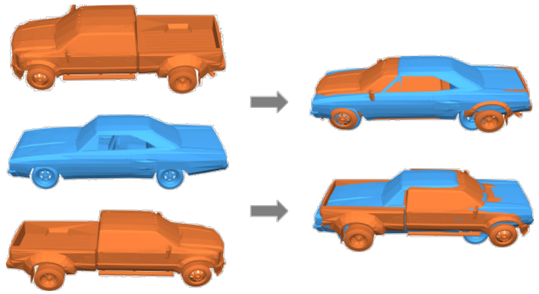


Works well if we have complete shapes and no noise.

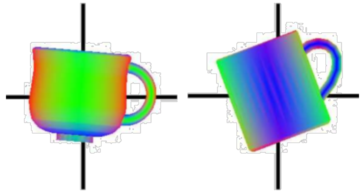
Fails for partial scans, outliers, high noise, etc.

Problems with PCA

- Principal axes are not consistently oriented
  - Symmetries cause problems



- Axes are unstable when principal values are similar



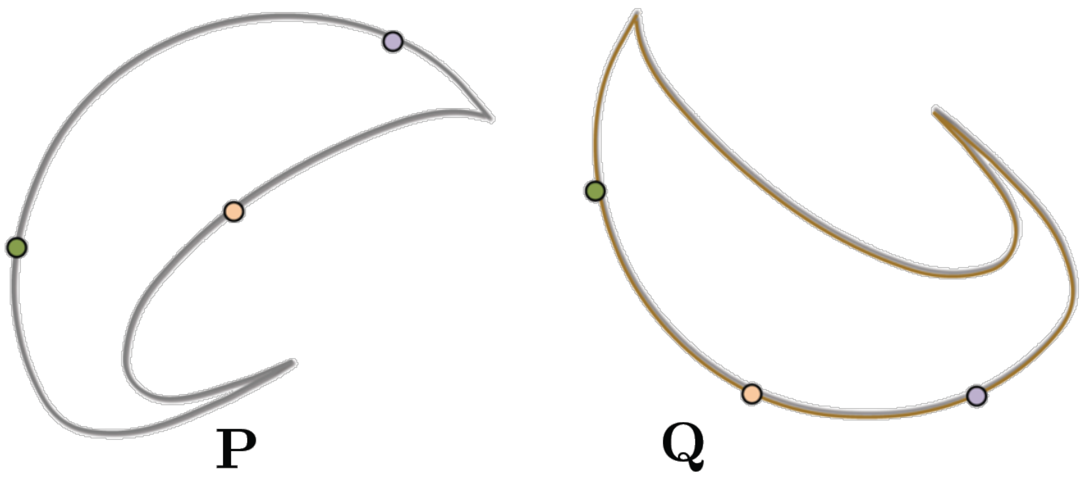
- Partial similarity

Random Sampling (RANSAC)

ICP only needs 3 point pairs! – Rigid motion space is 6-dimensional.

Robust and Simple approach. Iterate between:

1. Pick a random pair of 3 points on model & scan
2. Estimate alignment, and check for error.



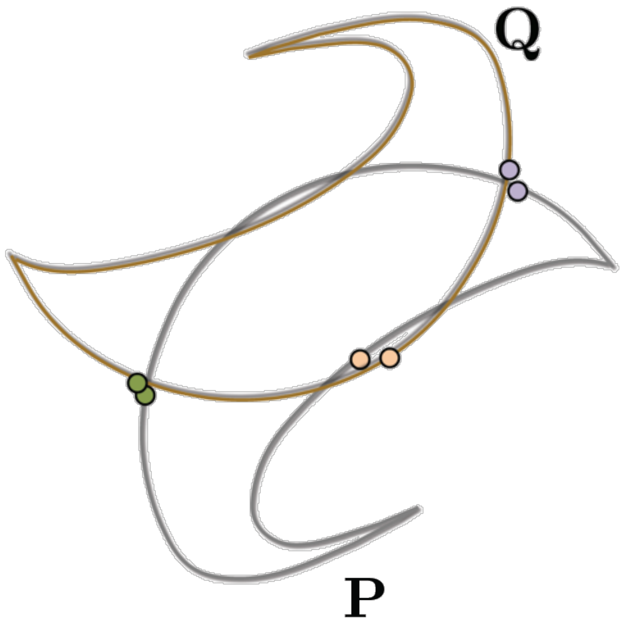
Guess and verify

Random Sampling (RANSAC)

ICP only needs 3 point pairs! – Rigid motion space is 6-dimensional.

Robust and simple approach. Iterate between:

1. Pick a random pair of 3 points on model & scan
2. Estimate alignment, and check for error.



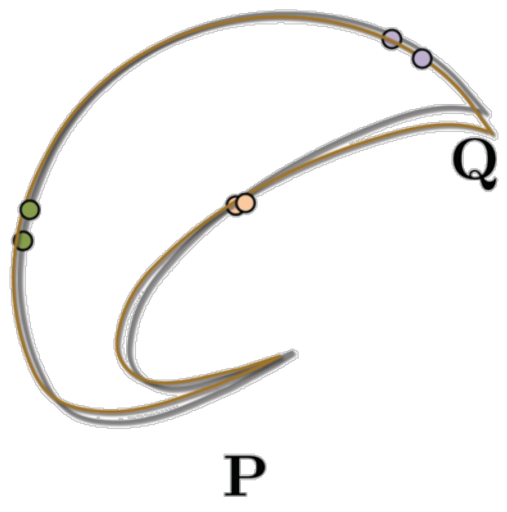
Guess and verify

Random Sampling (RANSAC)

ICP only needs 3 point pairs! – Rigid motion space is 6-dimensional.

Robust and simple approach. Iterate between:

1. Pick a random pair of 3 points on model & scan
2. Estimate alignment, and check for error.



Can also refine the final result. Picks don't have to be exact.

Can be expensive!

Guess and verify

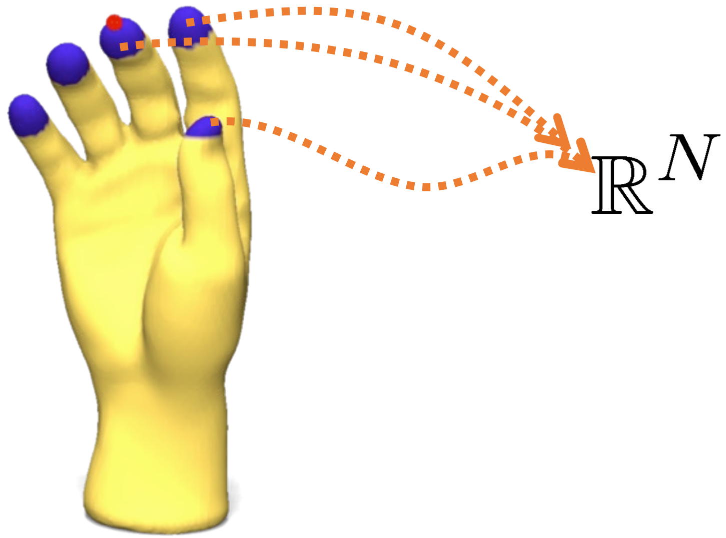
Global Matching – Invariant Features

Try to characterize the shape using properties that are invariant under the desired set of transformations.

Conflicting interests – invariance vs. informativeness.

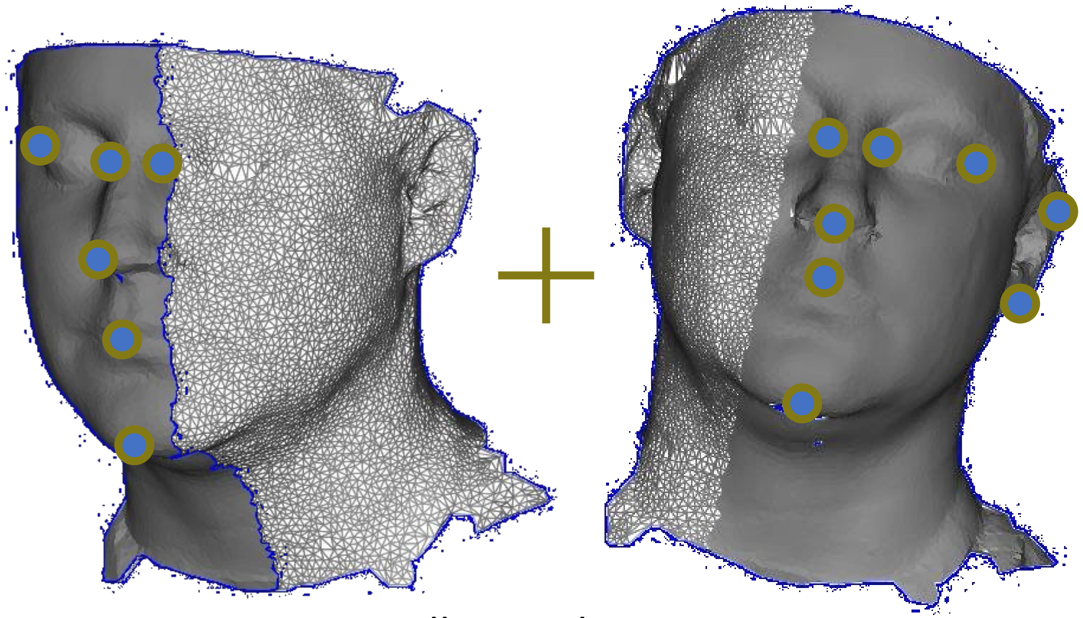
The most common pipeline:

1. identify salient feature points
2. compute informative and commensurable descriptors.



Matching Using Feature Points

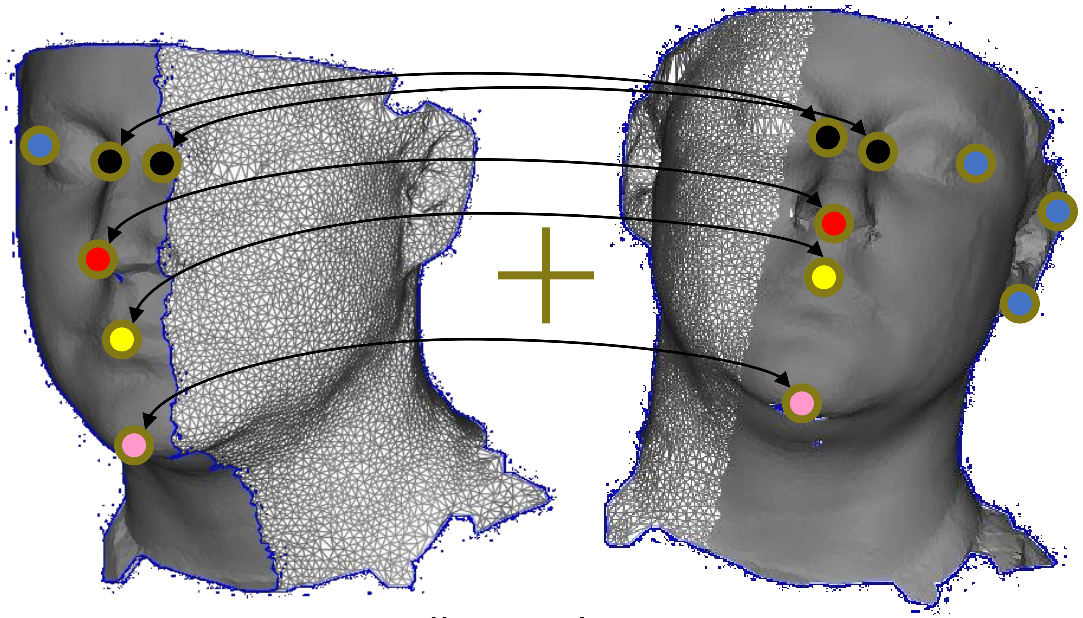
1. Find feature points on the two scans (we'll come back to that issue)



Partially Overlapping Scans

Approach

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

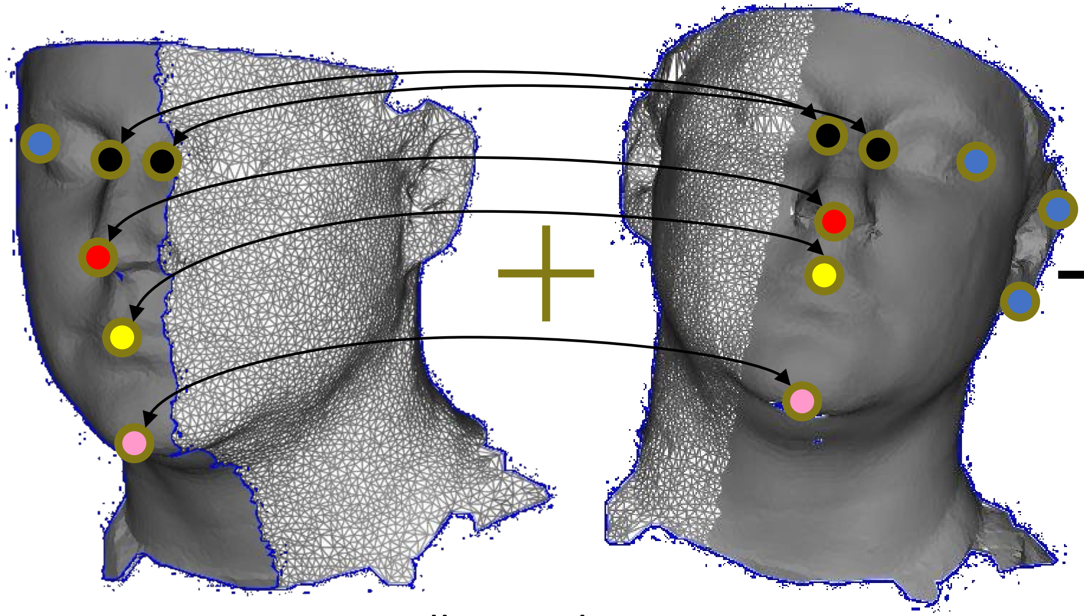


Partially Overlapping Scans

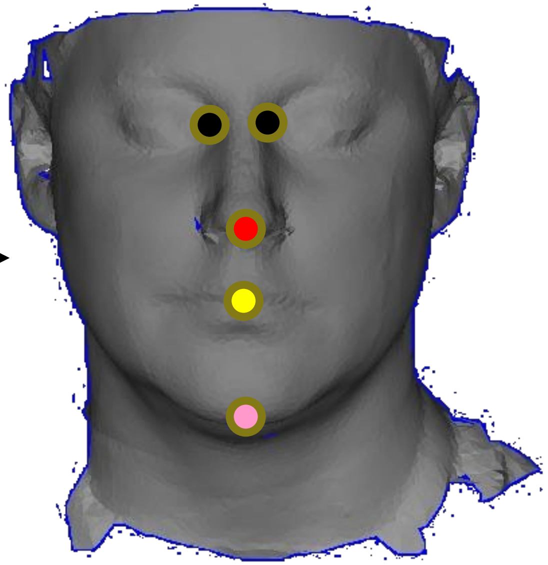
Approach

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

Preserve features
Various regularizers



Partially Overlapping Scans

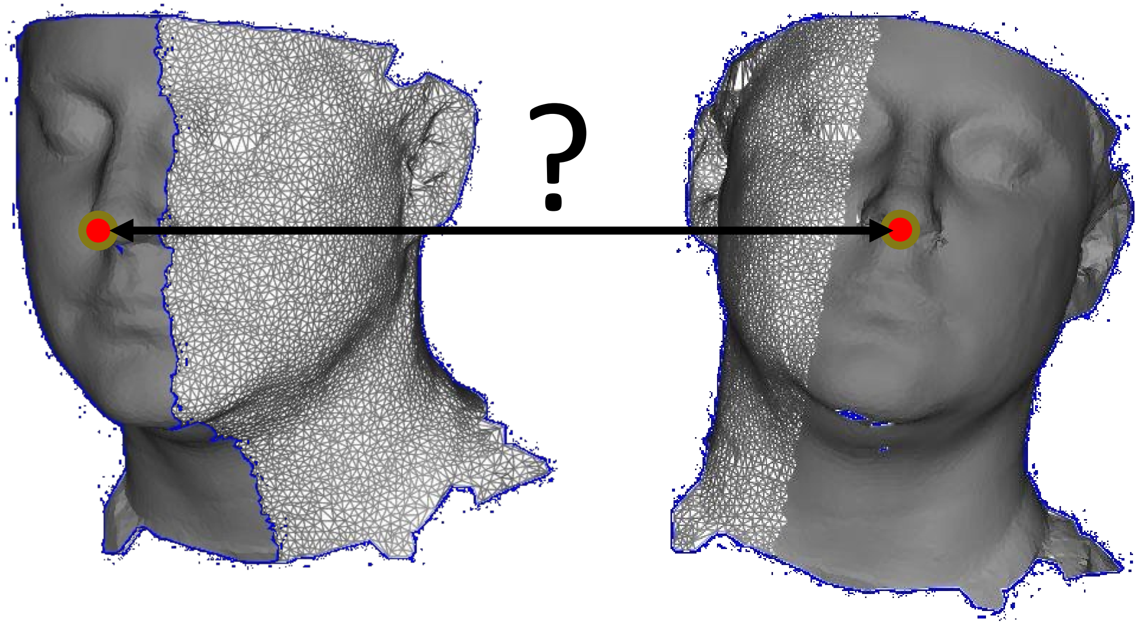


Aligned Scans

Correspondence

Goal:

Identify when two points on different scans represent the same feature

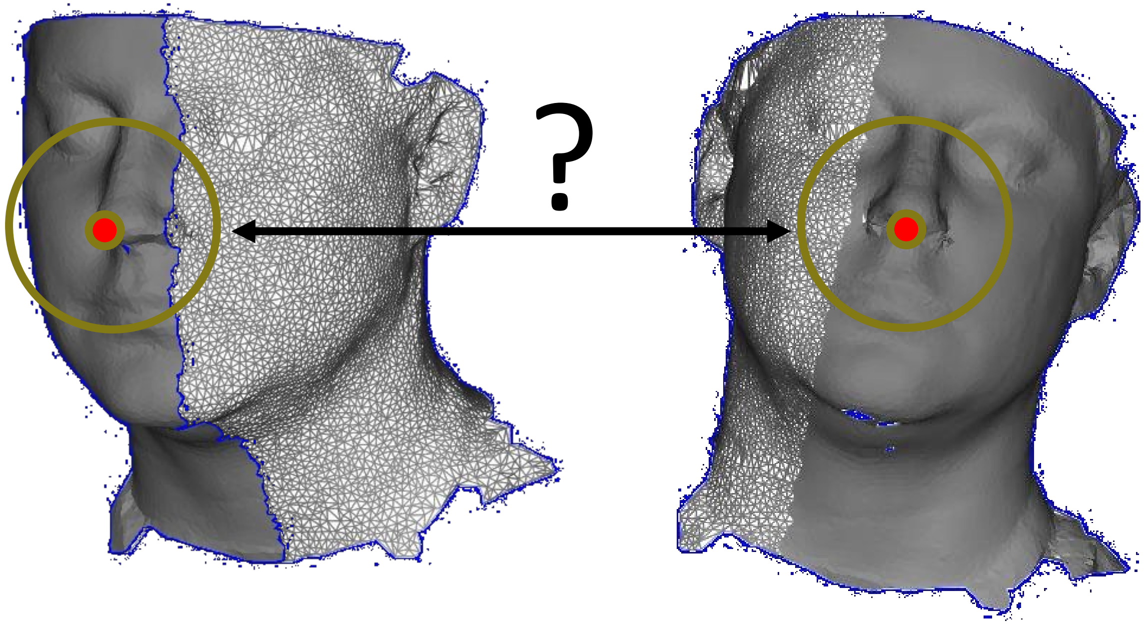


Correspondence

<u>Goal</u>:

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

Are the surrounding regions similar?

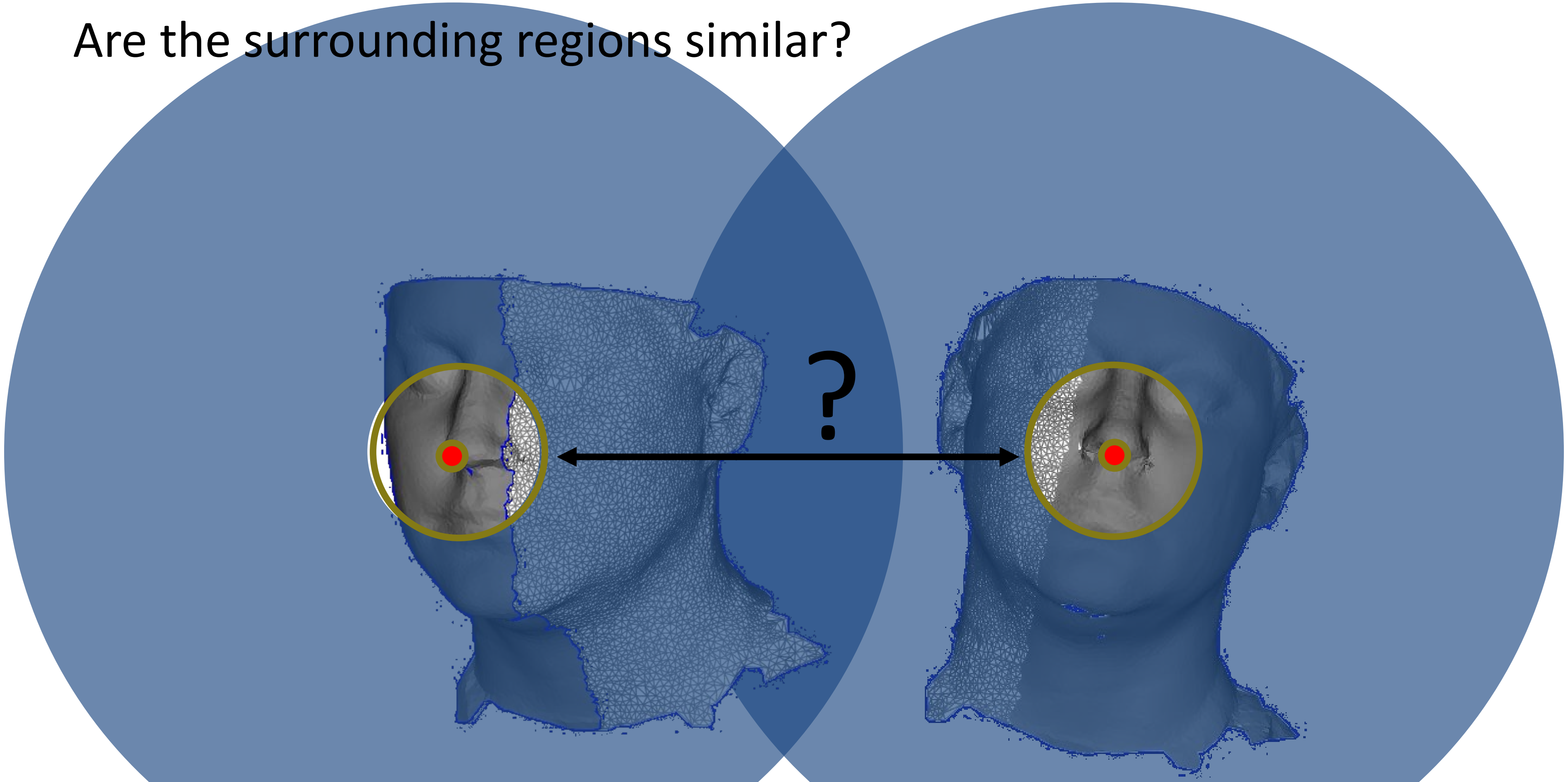


Correspondence

Goal:

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

Are the surrounding regions similar?

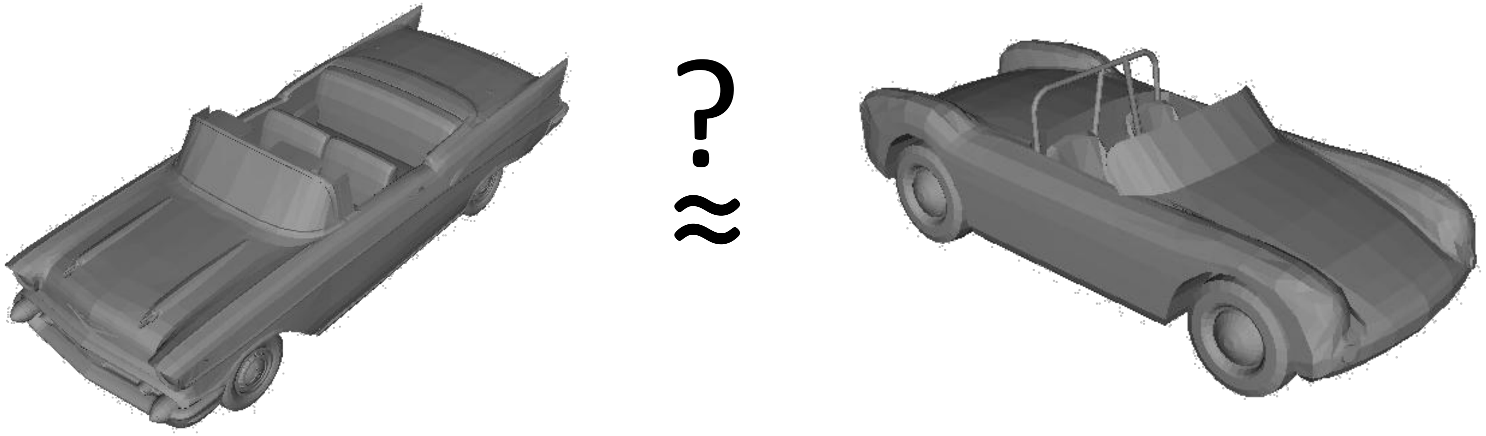


Shape Descriptors

Global Shape Similarity

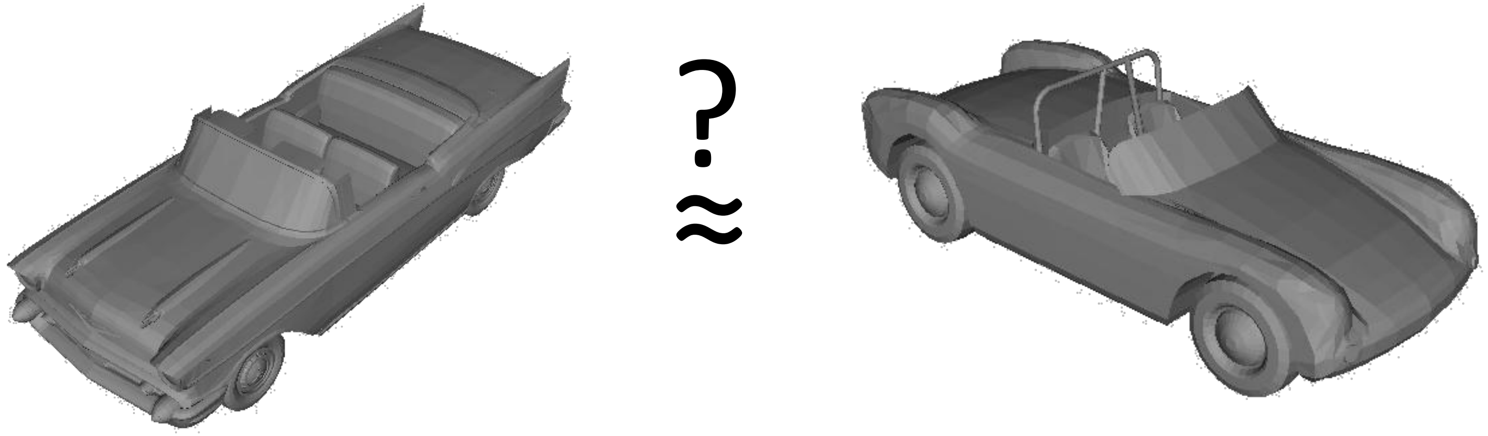
Global Similarity

Given two 3D models, determine if they represent the same/similar shapes



Global Similarity

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



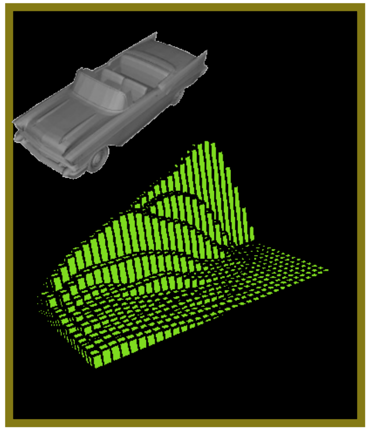
Models can have different:
representations, tessellations, topologies, etc.

Global Similarity

Approach:

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

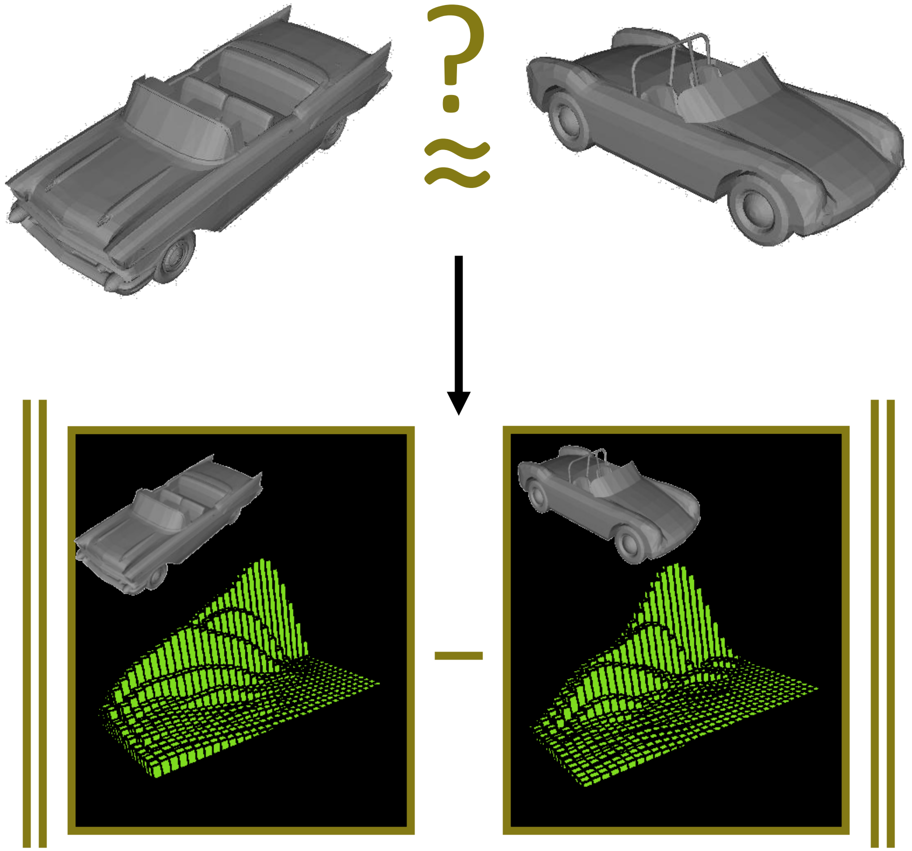
Typically a
high-dimensional vector



Global Similarity

Approach:

1. Represent each model by a shape descriptor
2. Compare shapes
   by comparing their
   shape descriptors
3. Is the descriptor designed, or learned?



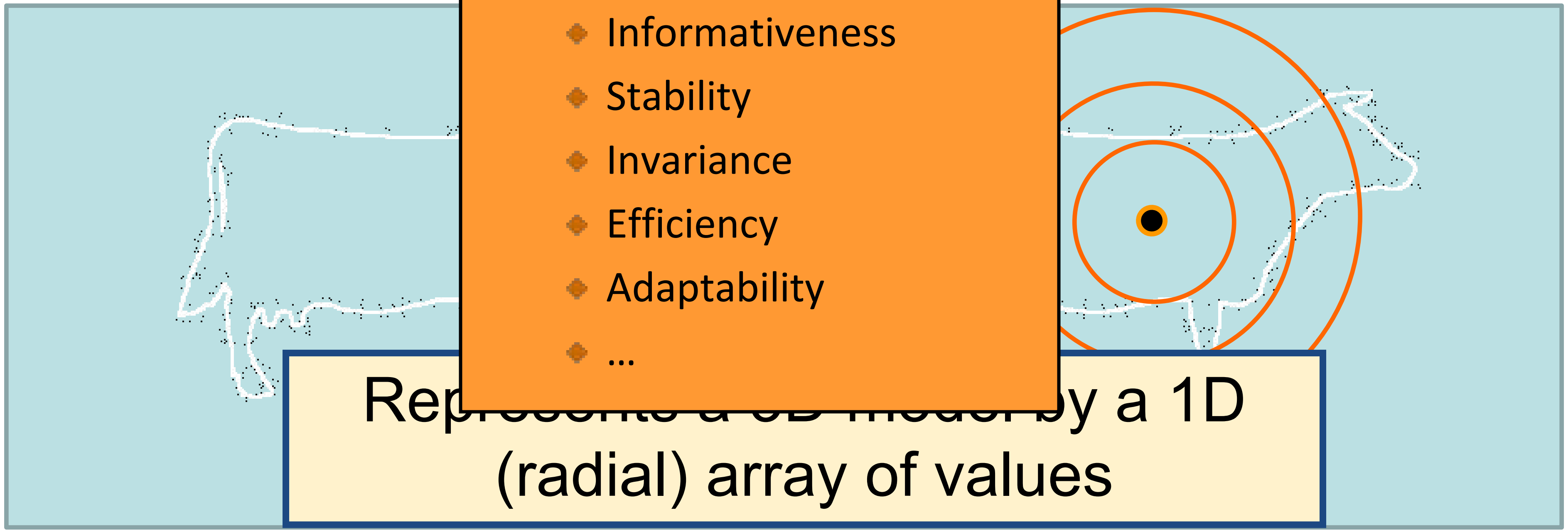
Shape Descriptors: Examples

Shape Histograms

[Ankerst *et al*. 1999]

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

- Feature desiderata
  - Informativeness
  - Stability
  - Invariance
  - Efficiency
  - Adaptability
  - …

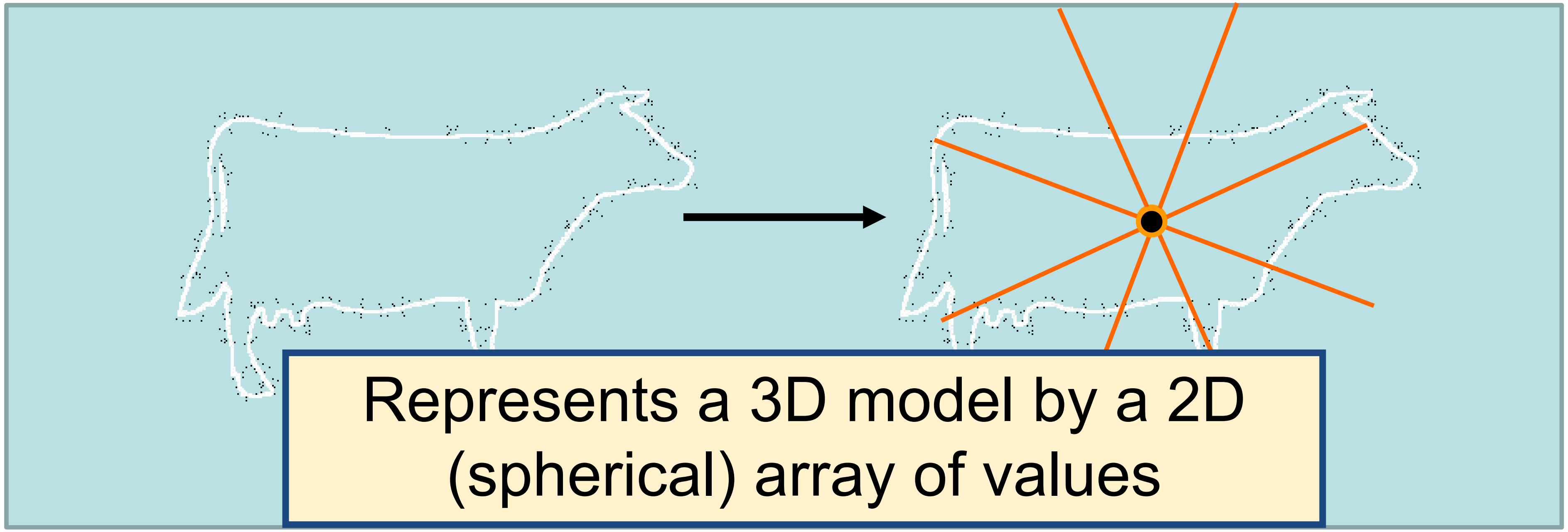


Shape Descriptors: Examples

Shape Histograms

[Ankerst *et al*. 1999]

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



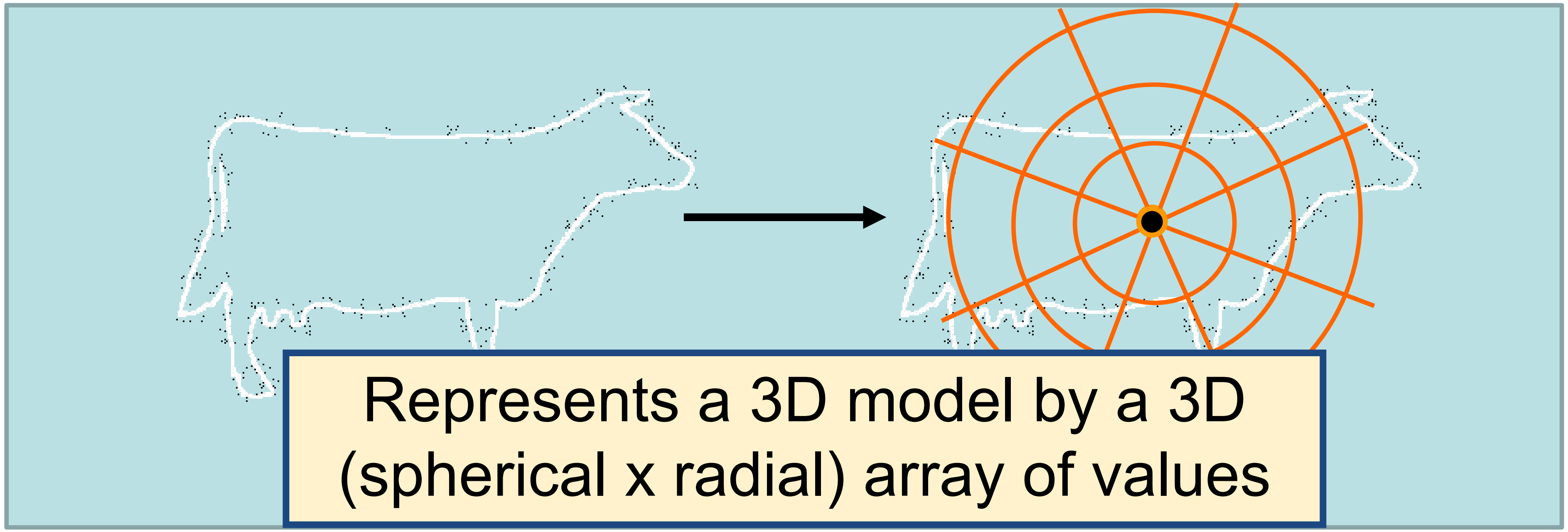
Represents a 3D model by a 2D (spherical) array of values

Shape Descriptors: Examples

Shape Histograms

[Ankerst *et al.* 1999]

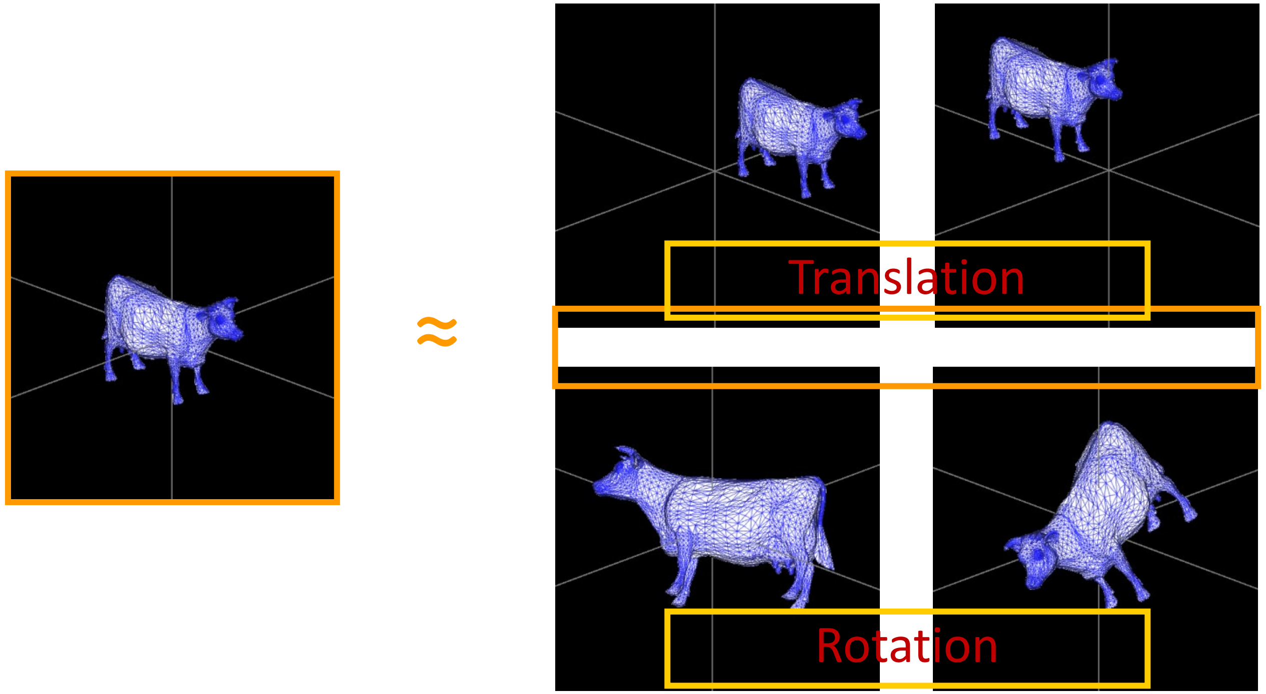
Shape descriptor stores a histogram of how much surface area resides within different shells and sectors in space



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

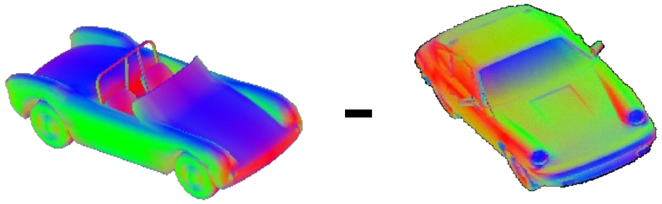
Shape Descriptors: Challenge

The descriptor must not change when a rigid body transformation (e.g., translation, rotation) is applied to the model



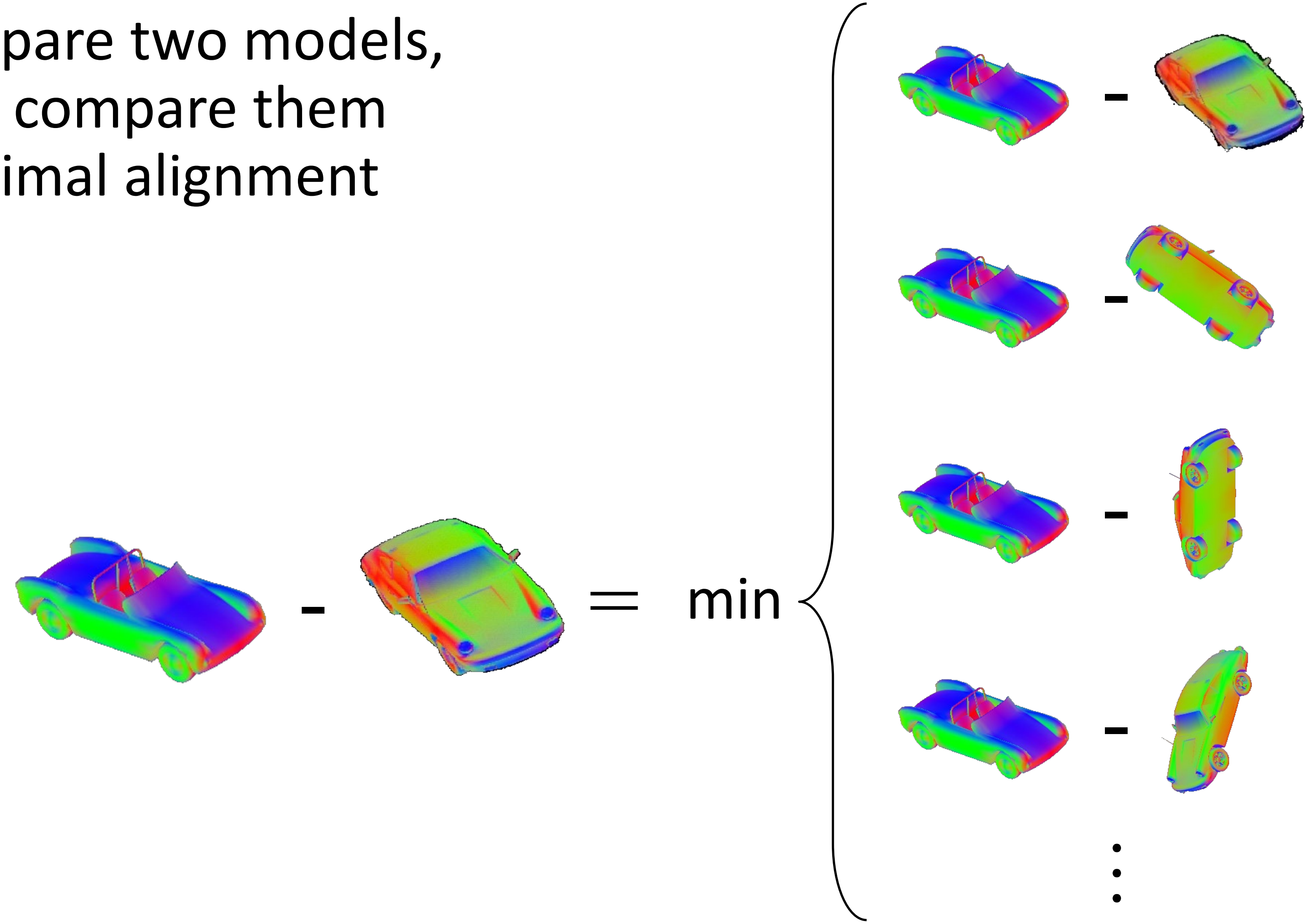
Shape Descriptors: Challenge

In order to compare two 3D models, we must either make descriptors invariant to rigid motions, or we need to compare the shapes at their optimal alignment



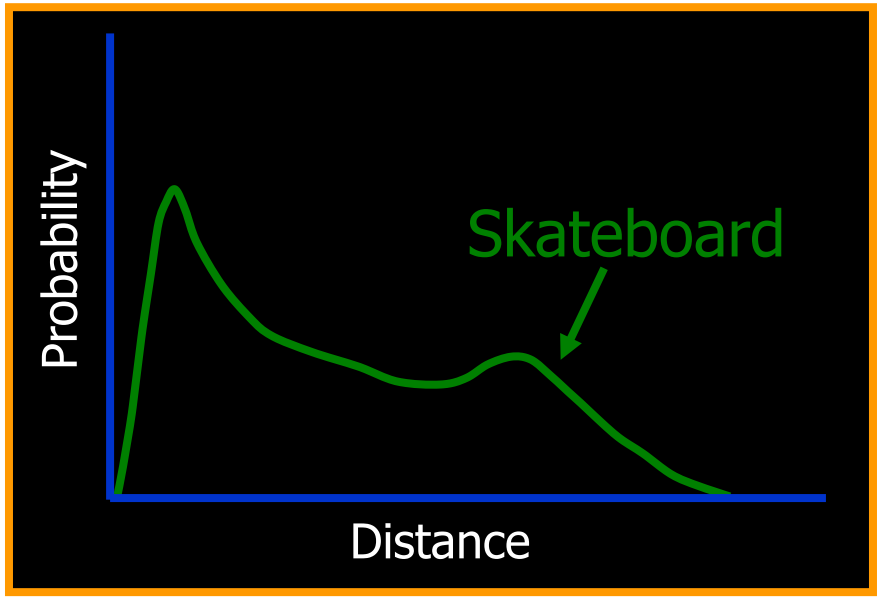
Shape Descriptors: Challenge

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



D2 Shape Distributions

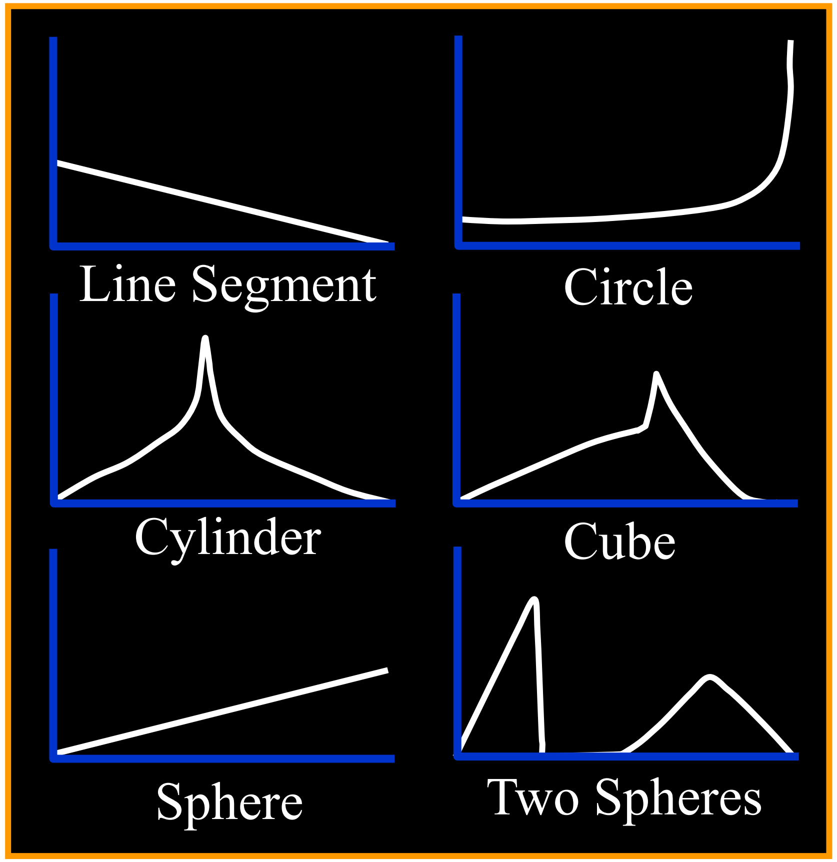
- Properties
  - Concise to store?
  - Quick to compute?
  - Invariant to transforms?
  - Efficient to match?
  - Insensitive to noise?
  - Insensitive to topology?
  - Robust to degeneracies?
  - Invariant to deformations?
  - Discriminating?



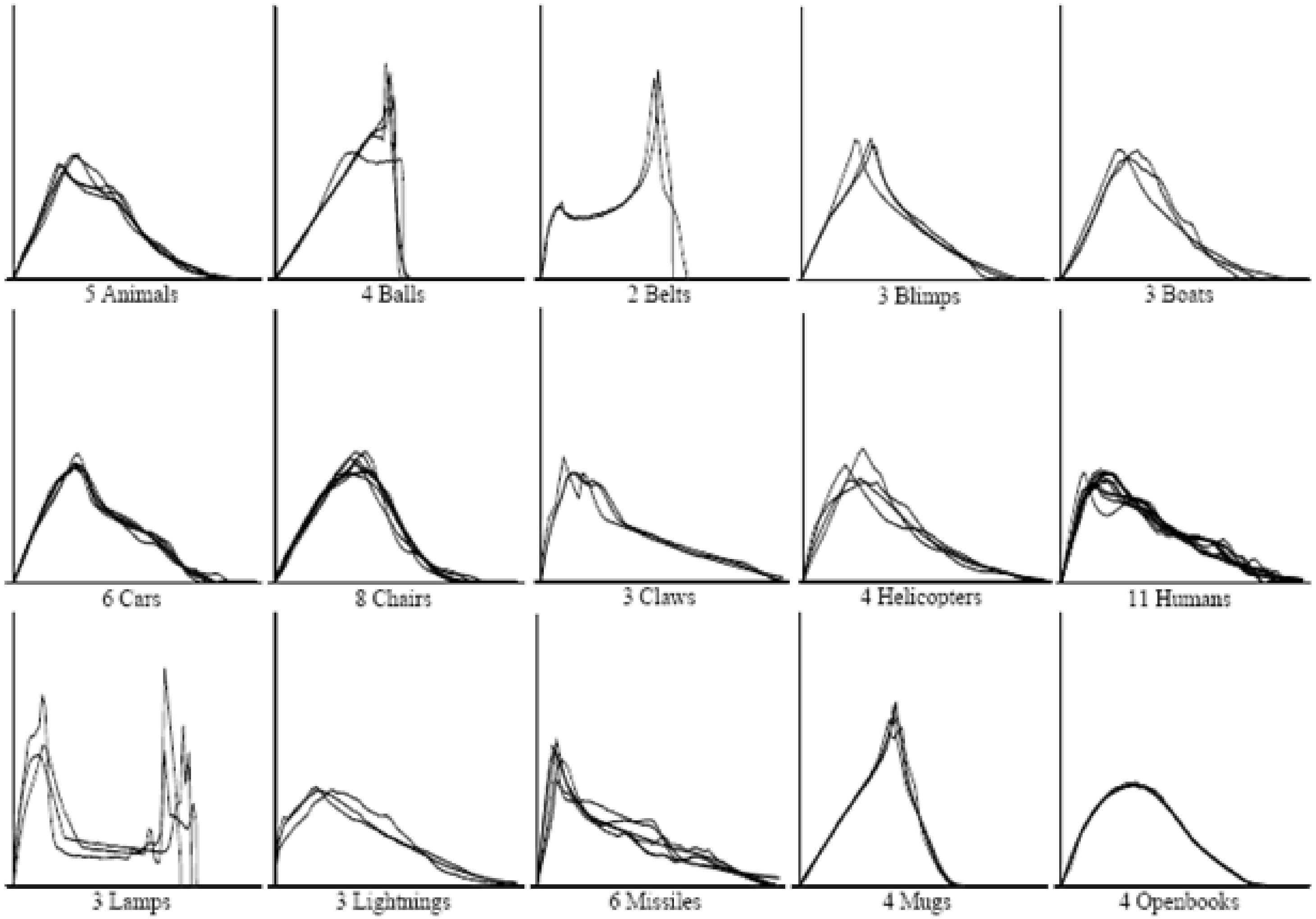
512 bytes (64 values)

0.5 seconds (10^6 samples)

Discriminating?

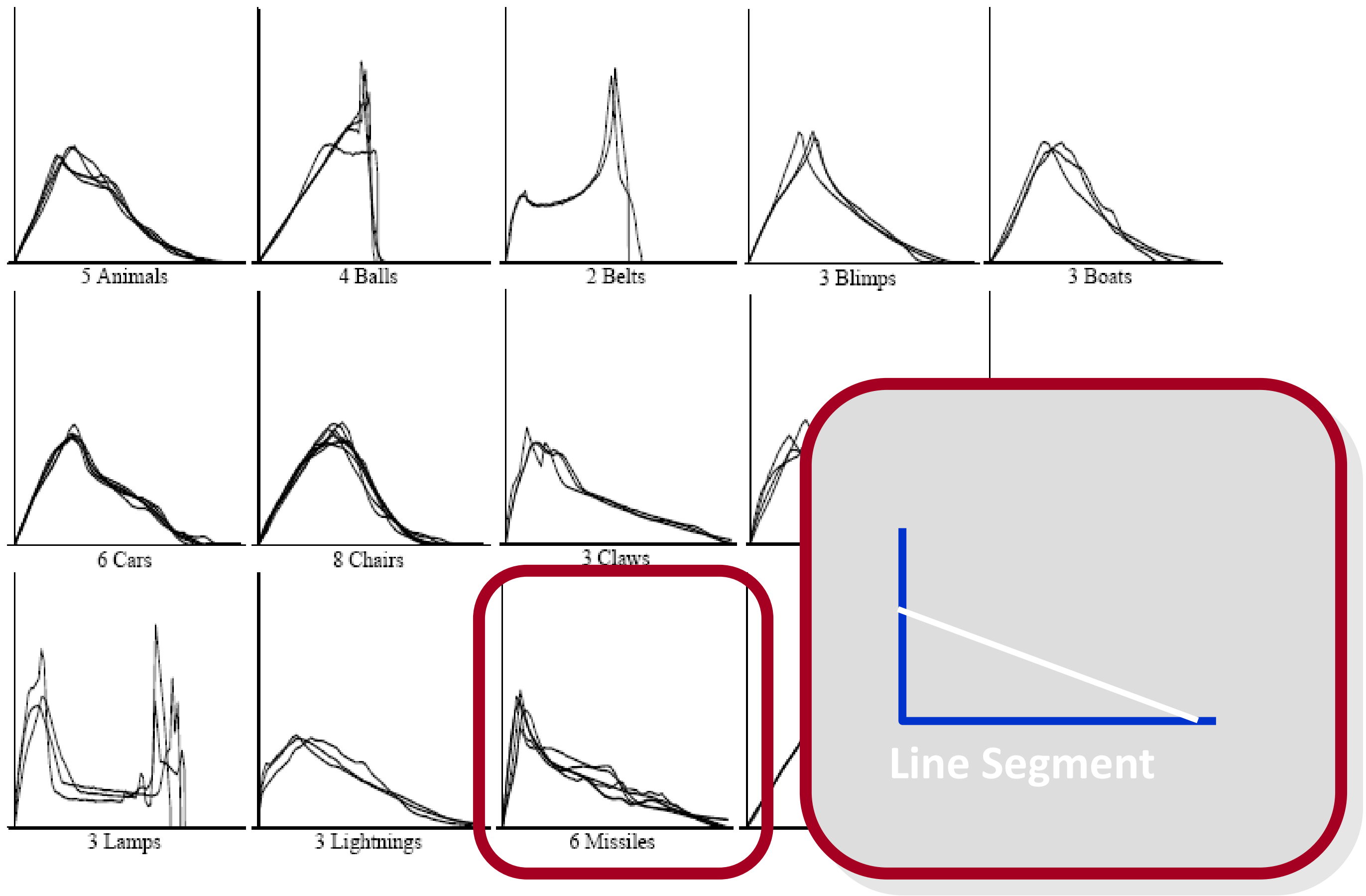


Stable?

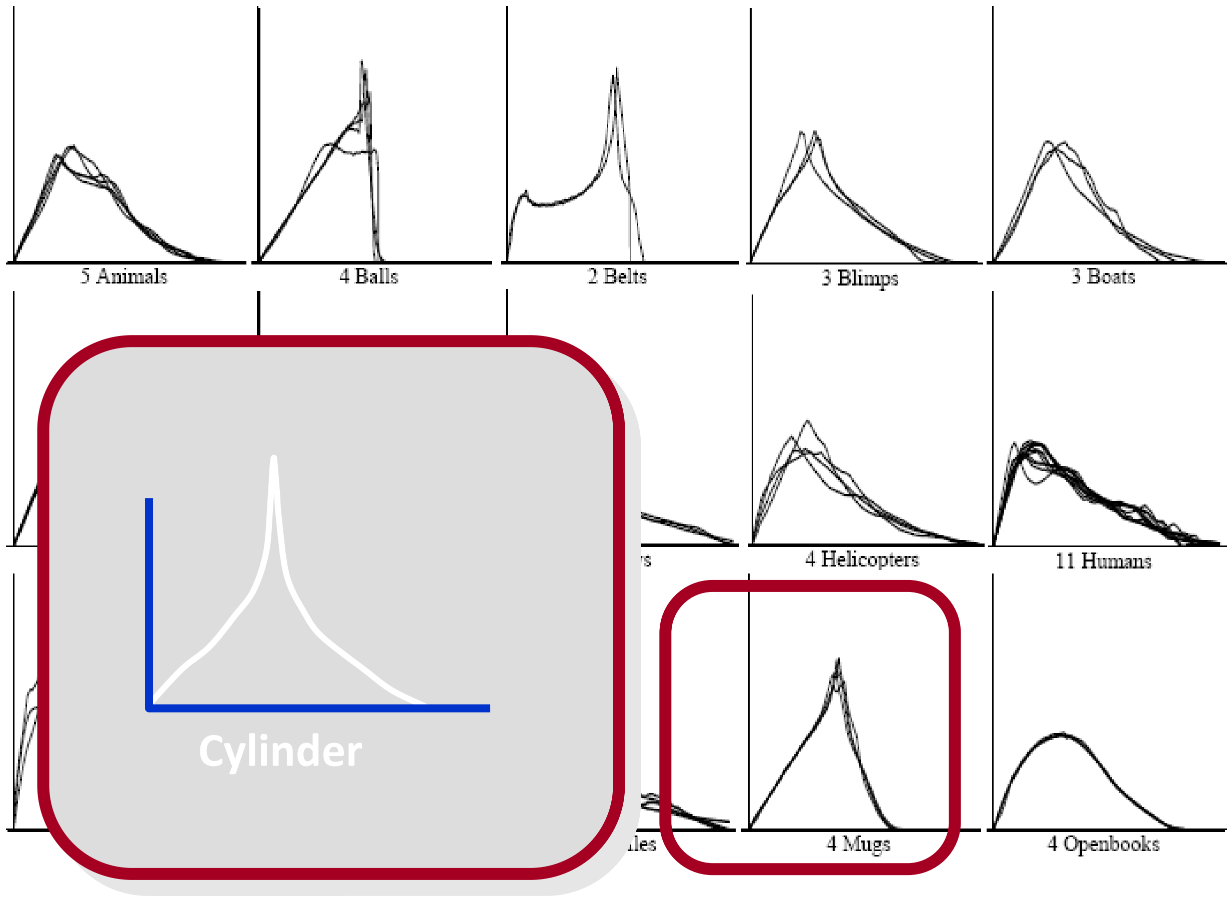


5 Animals
4 Balls
2 Belts
3 Blimps
3 Boats
6 Cars
8 Chairs
3 Claws
4 Helicopters
11 Humans
3 Lamps
3 Lightnings
6 Missiles
4 Mugs
4 Openbooks

Stable and Discriminating?



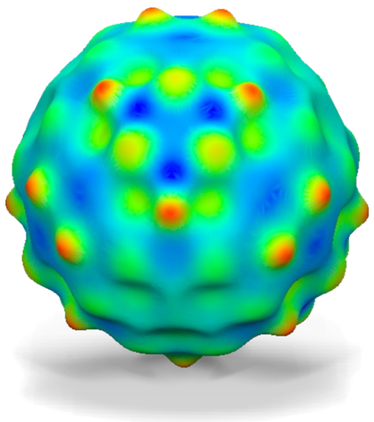
Stable and Discriminating



Local Features and Local Shape Similarity

Classical Curvature

- Differential features can be noisy on meshes

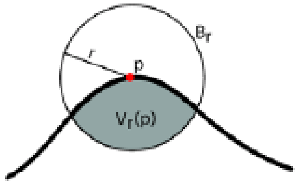


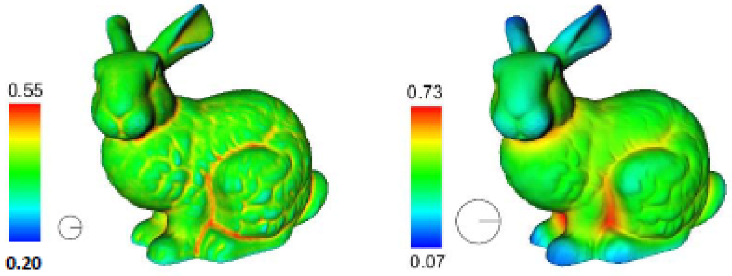
Curvature

Integral Volume Descriptor

Integral invariant signatures, Manay et al. ECCV 2004
Integral Invariants for Robust Geometry Processing, Pottmann et al. 2007-2009

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





Relation to mean curvature

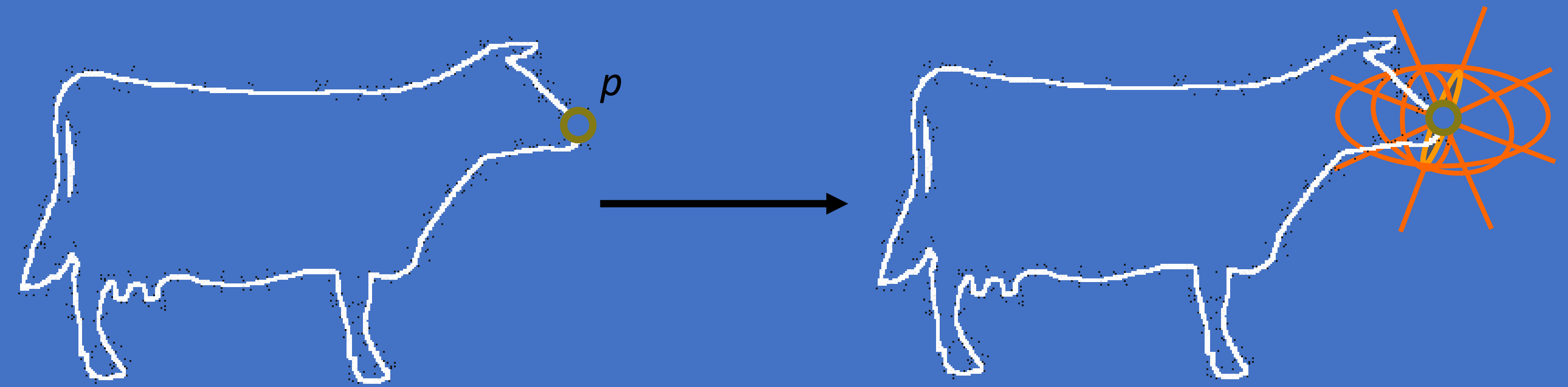
$$V_r(\mathbf{p}) = \frac{2\pi}{3} r^3 - \frac{\pi H}{4} r^4 + O(r^5)$$

Robust Global Registration,
Gelfand et al. 2005

From Global to Local

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

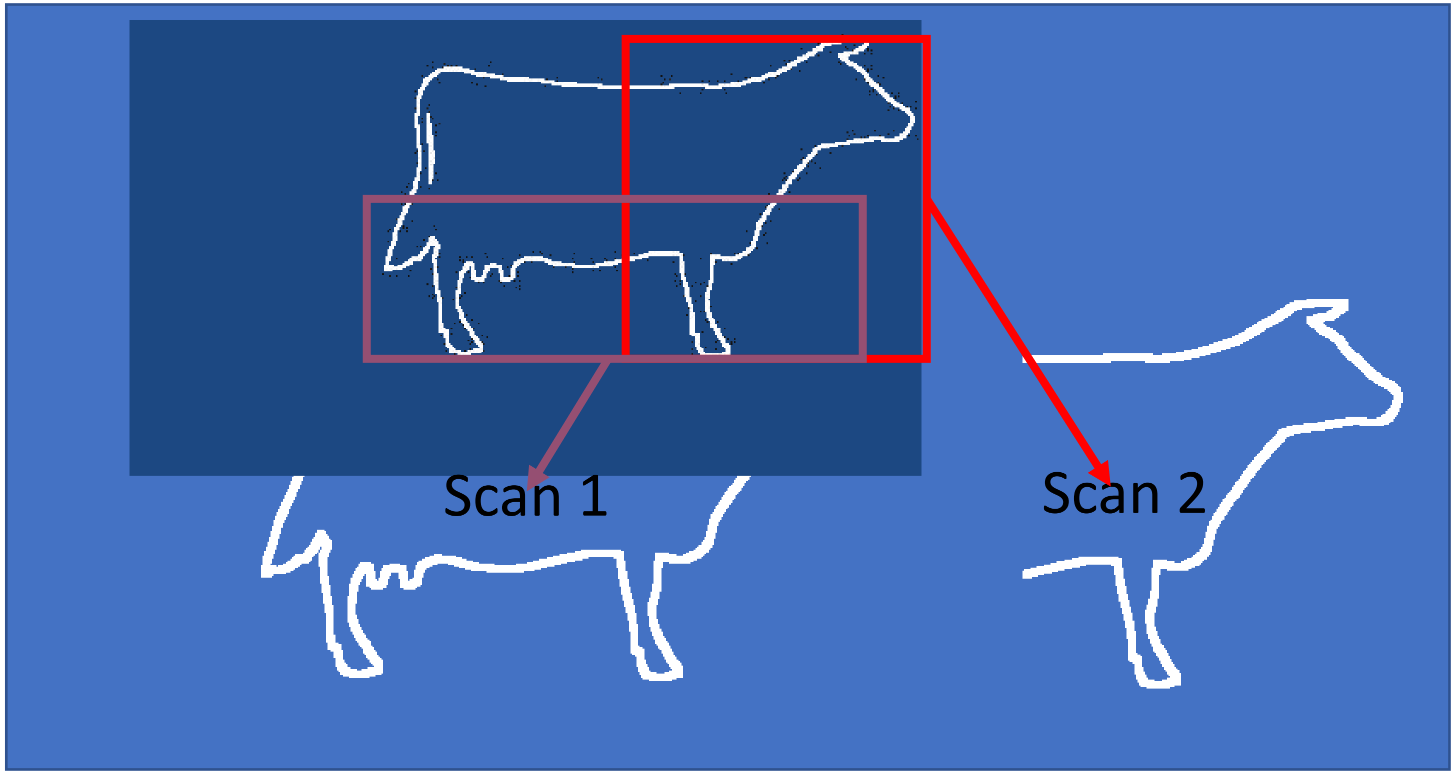
- center it about *p* (instead of the COM), and
- restrict the extent to a small region about *p*.



Shape histograms as local shape descriptors

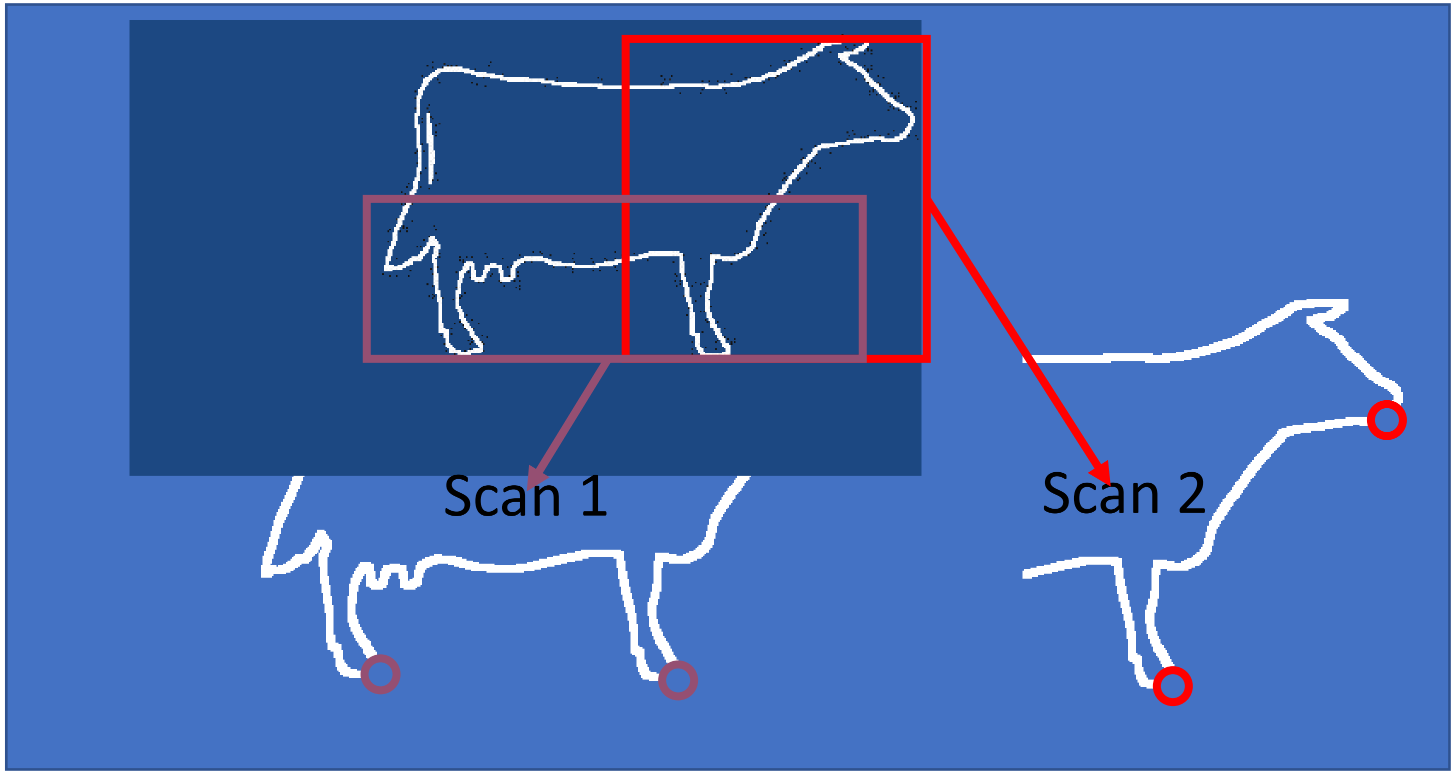
From Global to Local

Given scans of a model:



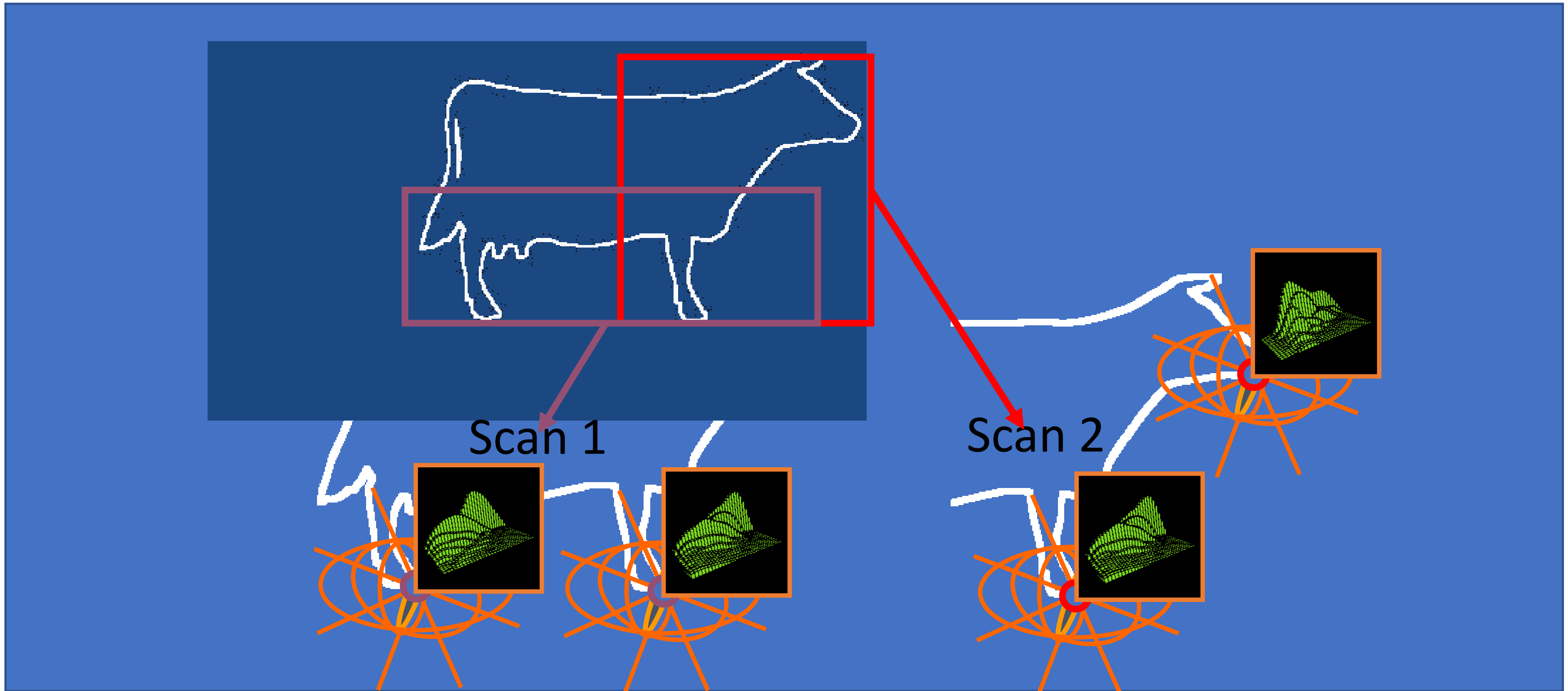
From Global to Local

- Identify the feature points



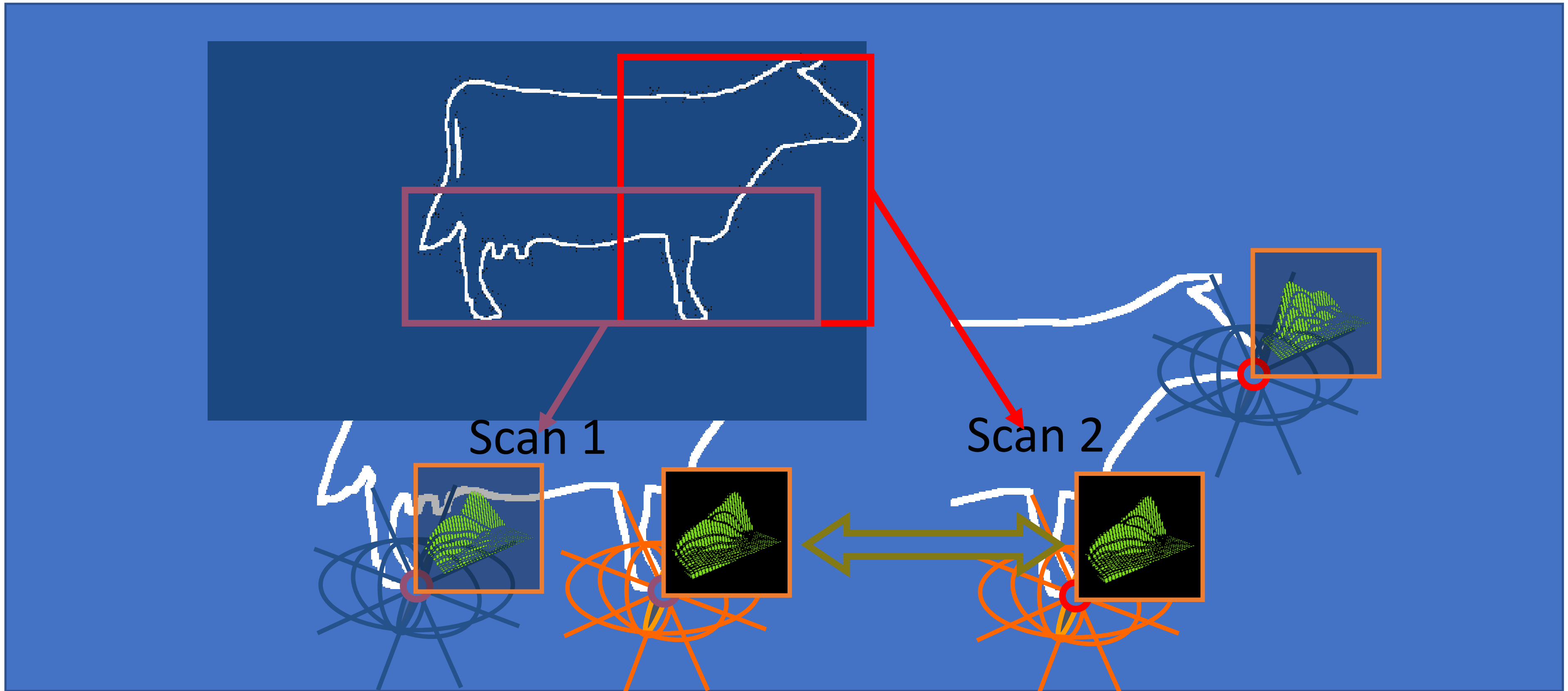
From Global to Local

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



From Global to Local

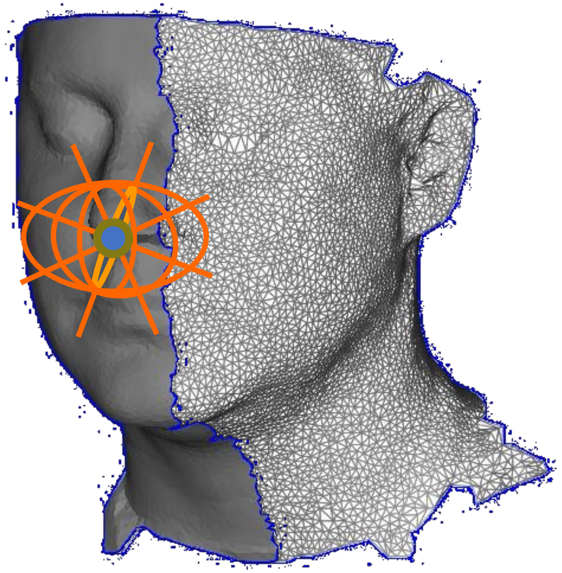
- Identify the features
- Compute a local descriptor for each feature
- For features correspond → descriptors are similar

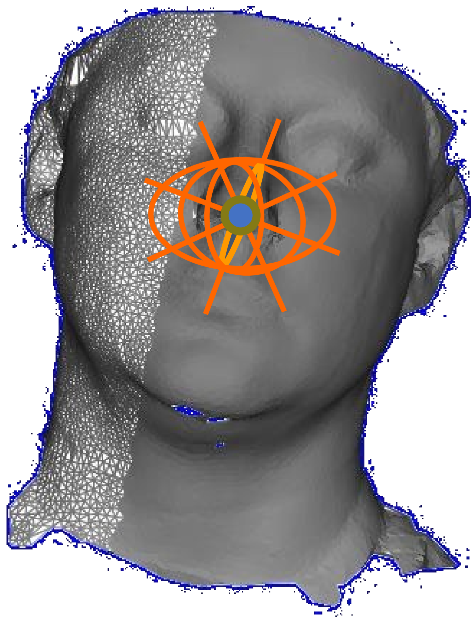


Pose Normalization

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.

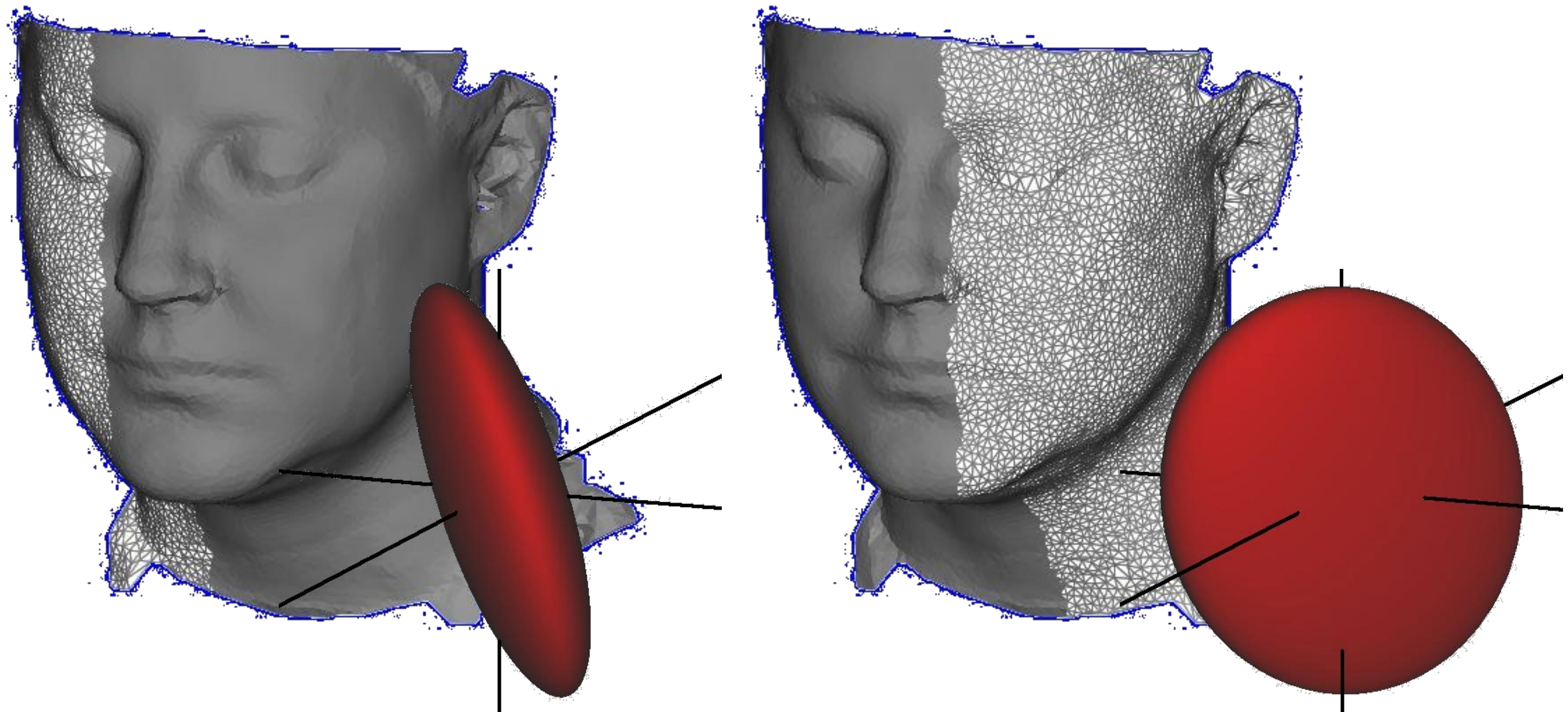




Pose Normalization

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

Spin Images

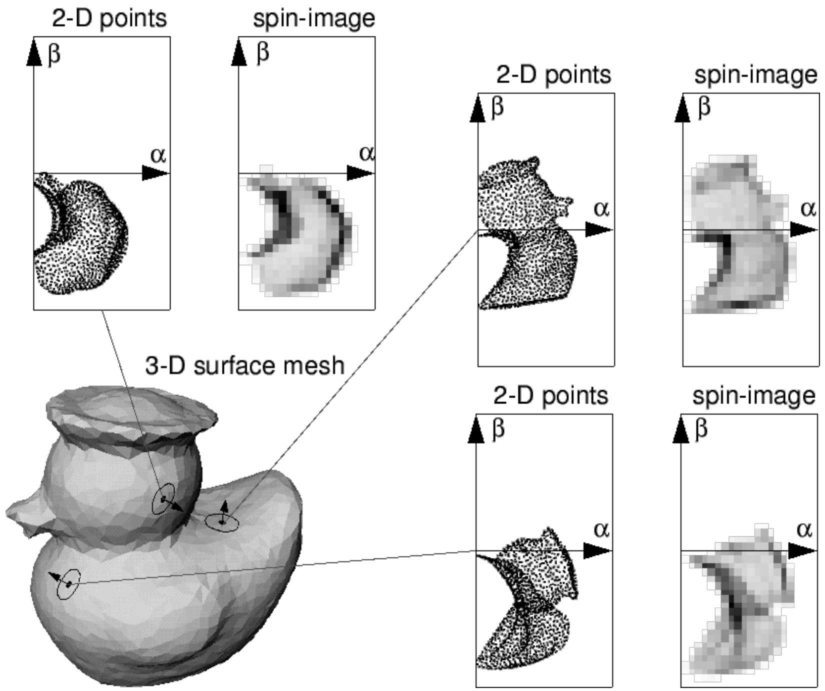
Creates an image associated with a neighborhood of a point.

Compare points by comparing their *spin images* (2D).

Given a point and a normal, every other point is indexed by two parameters:

β distance to tangent plane

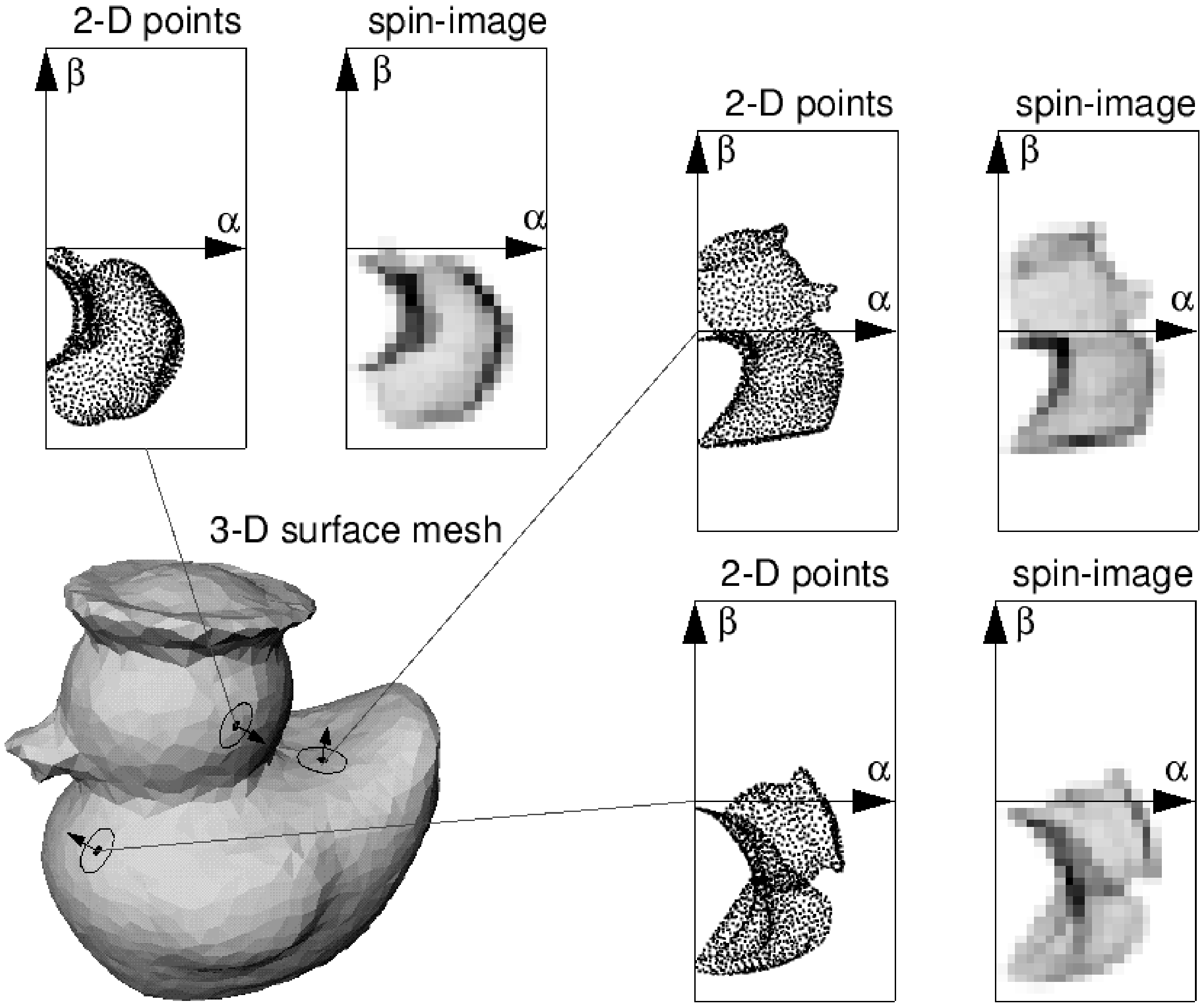
α distance to normal line



Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes
Johnson et al, PAMI 99

Spin Images

α - radial dist.
β - elevation



Main Question

How to compare regions on the shape in an invariant manner?

A large variety of *descriptors* have been suggested.

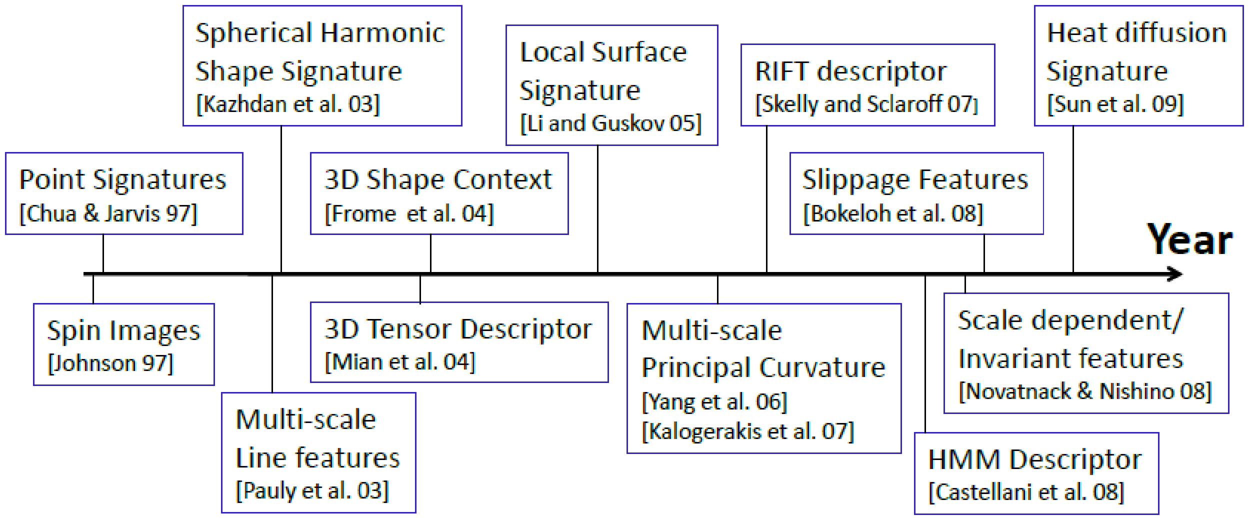


table by Will Chang

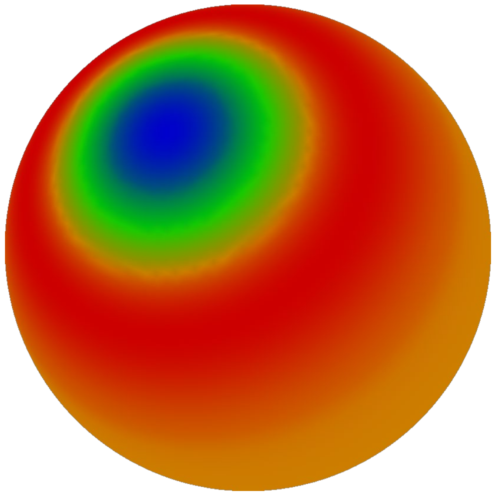
Intrinsic Descriptors for Deformable Matching

Laplace-Beltrami Operator

- Analog of Fourier transform on the sphere, but now on a general 2D manifold
- LB is an operators that can be applied to functions on manifolds to yield other functions

$$\Delta : C^\infty(M) \rightarrow C^\infty(M)$$

$$\Delta f = \mathrm{div} \nabla f$$



LB Eigen-decomposition

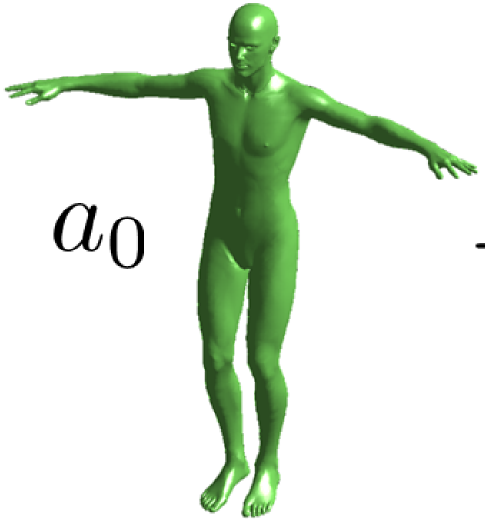
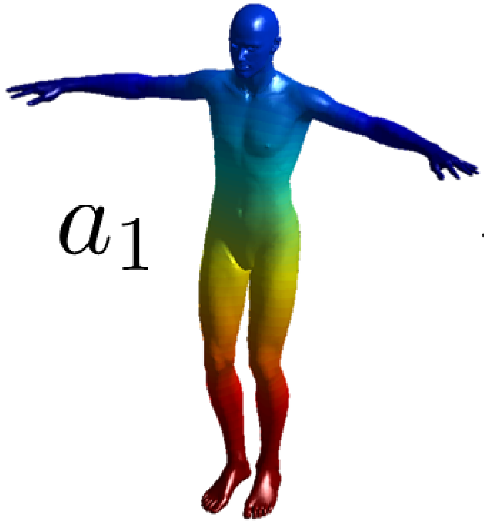
- The Laplace-Beltrami operator Δ has an eigendecomposition

$$\Delta \phi_i = \lambda_i \phi_i$$

$\lambda_0 = 0$ $\lambda_1 = 2.6$ $\lambda_2 = 3.4$ $\lambda_3 = 5.1$ $\lambda_4 = 7.6$

Multiscale Basis for a Function Space

$f : M \to \mathbb{R}$

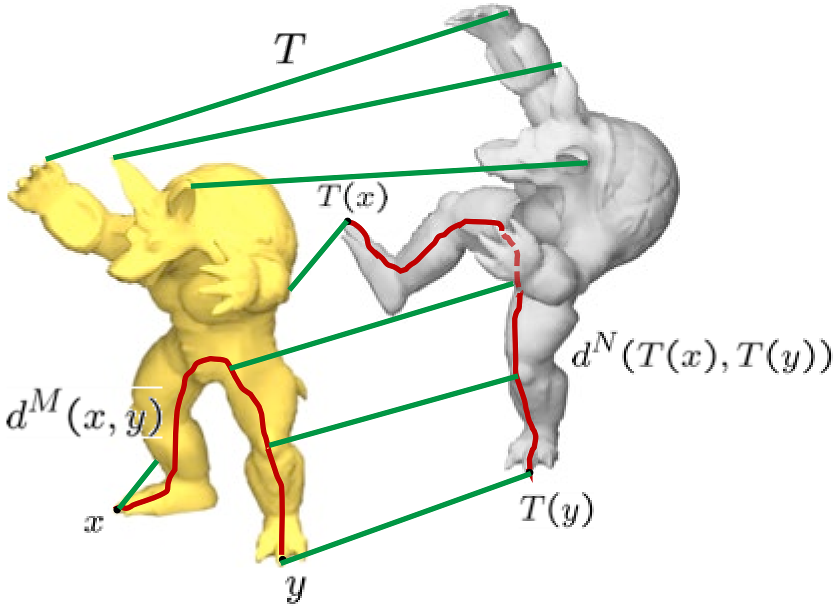
$= \; a_0$  $+ \; a_1$  $+ \; \dots$

$$f = \sum_{i=0}^{\infty} a_i \phi_i \qquad a_i = \int_M f(x)\phi_i(x)d\mu$$

Laplace-Beltrami – Isometry Invariant

Isometry invariance (LB is intrinsic)

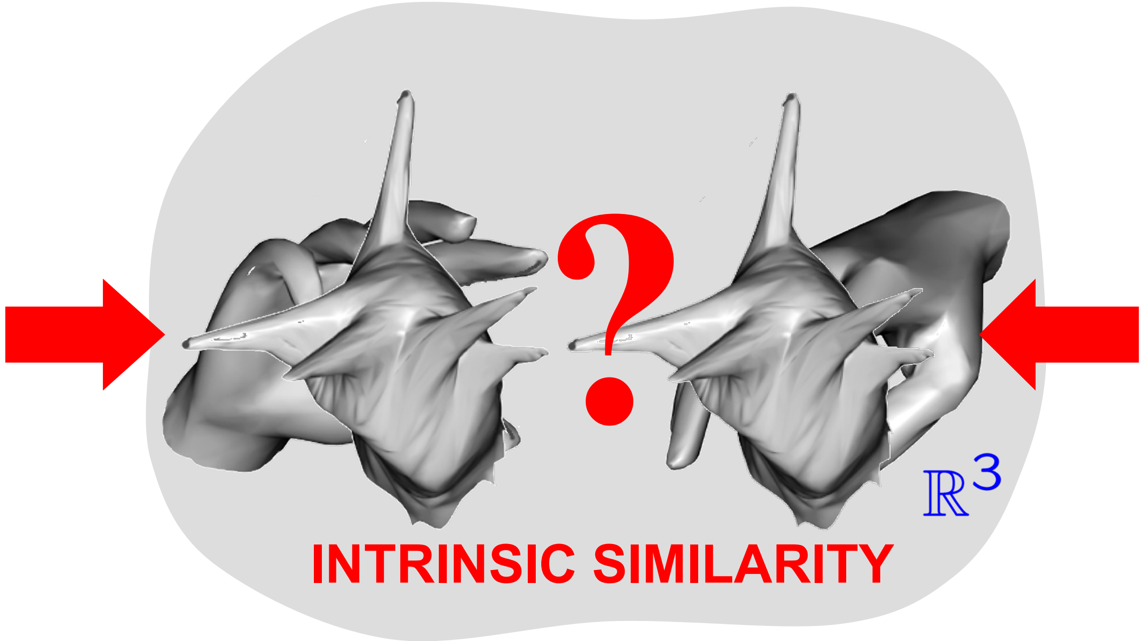
If two shapes are isometric then their LB operators agree.



Any quantity derived from the LB operator has to be invariant to isometries.

Global Point Signature (GPS)

$$GPS(p) = \left(\frac{1}{\sqrt{\lambda_1}}\phi_1(p), \frac{1}{\sqrt{\lambda_2}}\phi_2(p), \frac{1}{\sqrt{\lambda_3}}\phi_3(p), \cdots \right)$$



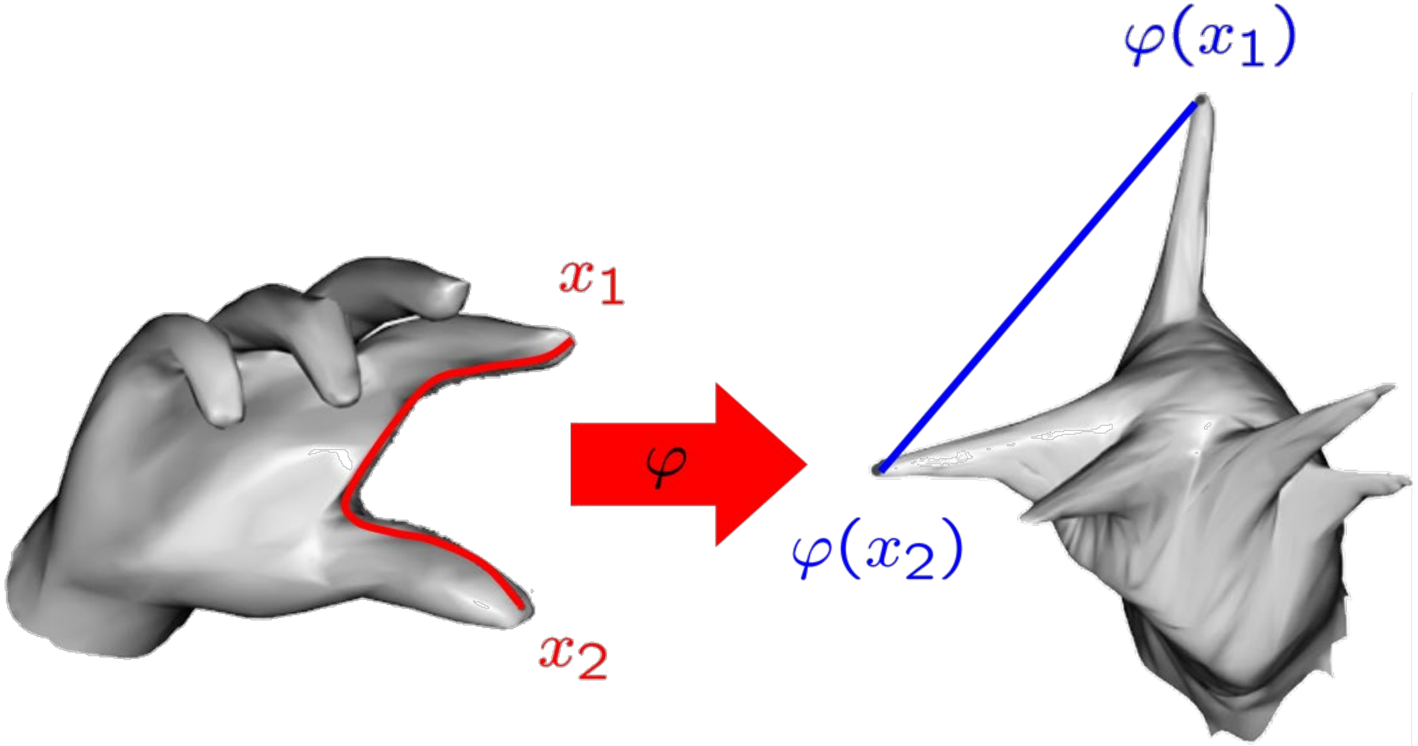
but there is a sign ambiguity

Rustamov et al. 2007

Global Point Signature

almost invariant under isometries – but not completely canonical

$$GPS(p) = \left(\frac{1}{\sqrt{\lambda_1}}\phi_1(p), \frac{1}{\sqrt{\lambda_2}}\phi_2(p), \frac{1}{\sqrt{\lambda_3}}\phi_3(p), \cdots \right)$$



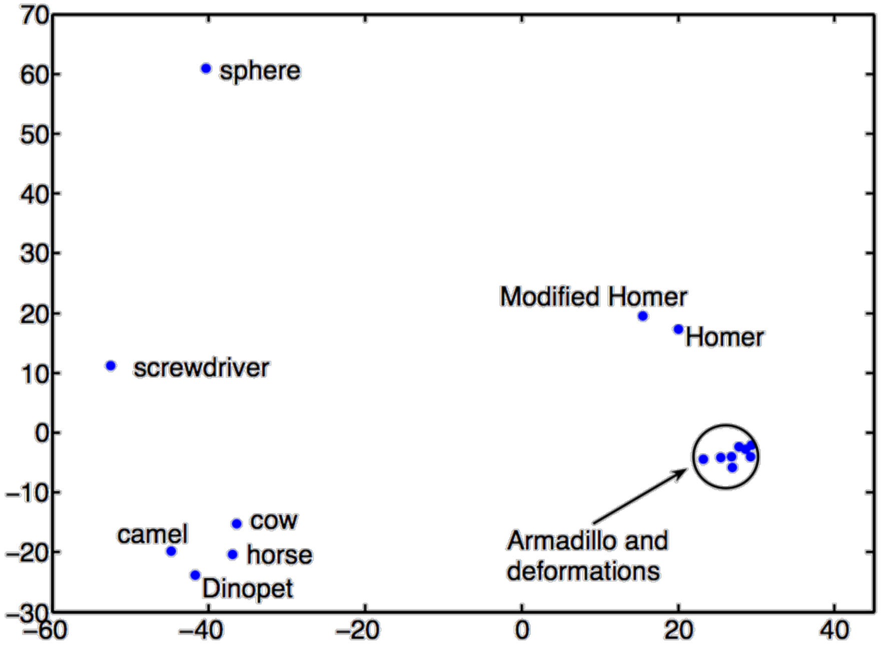
Diffusion distances are also intrinsic and also canonical

Rustamov et al. 2007

Global Point Signature

$$GPS(p) = \left(\frac{1}{\sqrt{\lambda_1}}\phi_1(p), \frac{1}{\sqrt{\lambda_2}}\phi_2(p), \frac{1}{\sqrt{\lambda_3}}\phi_3(p), \cdots \right)$$

Figure 4: *Armadillo and its deformations.*



Similar to D2, but use histograms in embedded space (rather than Euclidean)

Rustamov et al. 2007

Global Point Signature

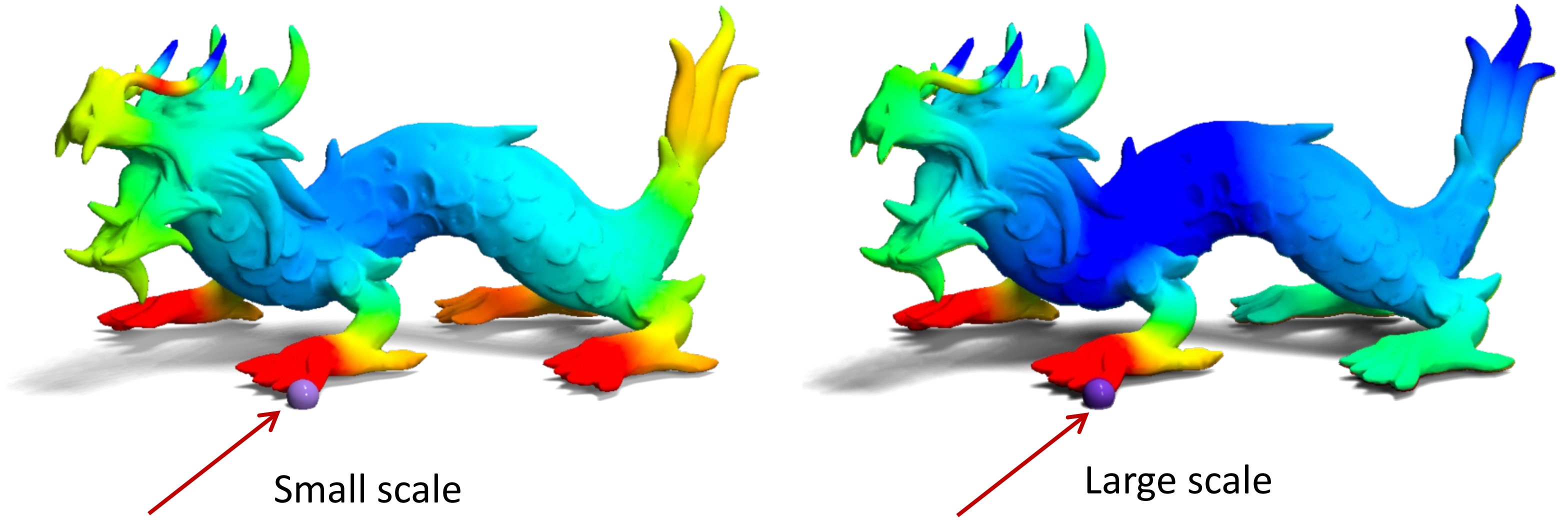
$$GPS(p) = \left(\frac{1}{\sqrt{\lambda_1}}\phi_1(p), \frac{1}{\sqrt{\lambda_2}}\phi_2(p), \frac{1}{\sqrt{\lambda_3}}\phi_3(p), \cdots \right)$$

- Pros
  - Isometry-invariant
  - Nearly canonical
- Cons
  - Eigenfunctions may flip sign
  - Eigenfunctions might change positions due to deformations
  - Only global – a point descriptor depends on the entire shape

Rustamov et al. 2007

The Issue of Scale

- Given a point (●) on a shape, find other points with "similar" neighborhoods



- Inherently multiscale question: on a manifold, locally all points are the same. Need a meaningful way to compare point neighborhoods at different scales
- At what scale do neighborhoods become unique?

(Heat) Diffusion on Manifolds

- Heat diffusion on a Riemannian manifold:

  If $u(x,t)$ is the amount of heat at point x at time t,

  then

$$\frac{\partial u}{\partial t} = \Delta u$$

Δ : Laplace-Beltrami Operator (div grad)

- Given an initial distribution $f(x)$. After time t:

$$f(x,t) = e^{-t\Delta} f$$

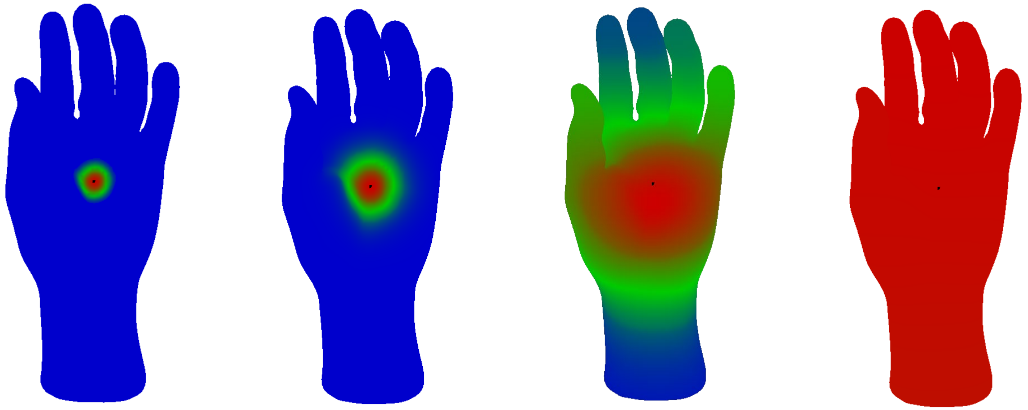
H_t heat operator

The Heat Kernel

- Heat kernel $k_t(x, y)$

$$f(x,t) = \int_{\mathcal{M}} k_t(x,y) f(y) dy$$

$k_t(x, y)$: amount of heat transferred from x to y in time t.
How well x and y are connected at scale t



$t = 0.001$ $t = 0.02$ $t = 3$ $t = 40$

Background

- Heat Kernel $k_t(x,y)$. Also the probability density function of Brownian motion on $\mathcal{M}$:

$$\mathbb{P}\left(W_x^t \in C\right) = \int_C k_t(x,y)dy$$

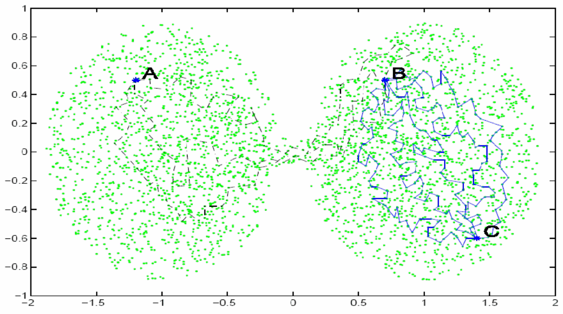
- Intuitively: weighted average over all paths possible between x and y in time t

- Related to Diffusion Distance:

  $D_t(x,y) = k_t(x,x) - 2k_t(x,y) + k_t(y,y)$

  a robust multi-scale measure

  of proximity



Coifman, Lafon

Heat Kernel Properties

Basic Properties

- $k_t(x, y) = k_t(y, x)$
- $k_{t+s}(x, y) = \int_M k_t(x, z) k_s(z, y) dz$
- $k_t(x, y) = \sum_{i=0}^{\infty} e^{-\lambda_i t} \phi_i(x) \phi_i(y)$

Eigenfunctions of LB

Heat Kernel Properties

- Invariant under isometric deformations

  If $T : X \to Y$ is an isometry, then:

$$k_t(X, Y) = k_t(T(x), T(y))$$

- Conversely: it characterizes the shape up to isometry.

  If $k_t(X, Y) = k_t\,(T(x), T(y)) \;\; \forall\; x, y, t$ then:

  T is an isometry.

This is because:

$$\lim_{t \downarrow 0} \left(t \log k_t(x, y)\right) = -\frac{1}{4} d^2_{\mathcal{M}}(x, y) \;\; \forall\; x, y$$

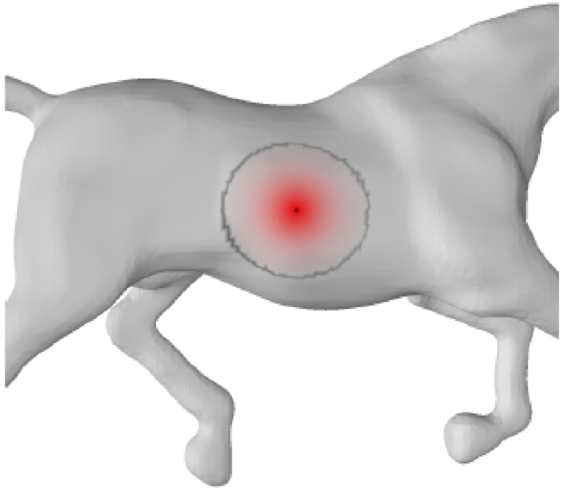
where $d_{\mathcal{M}}(\cdot, \cdot)$ is the geodesic distance

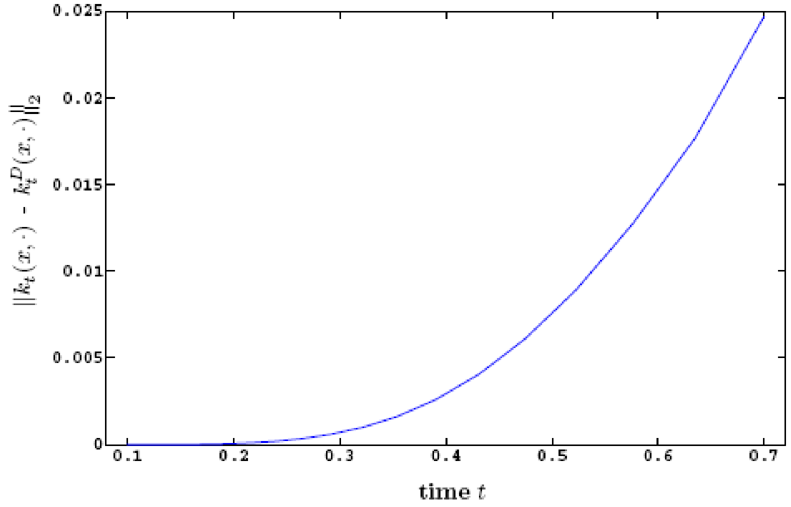
Heat Kernel Properties

- Multiscale:

  For a fixed x , as t increases, heat diffuses to larger and larger neighborhoods

  Therefore, $k_t(x, \cdot)$ is determined by (reflects the properties of) a neighborhood that grows with t

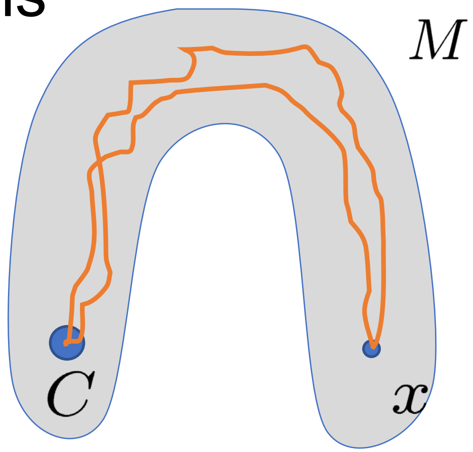




Heat Kernel Properties

- Robustness:

  $k_t(x, \cdot)$ is the probability density function of a Brownian motion, a weighted average over all paths, which is generally not very sensitive to local perturbations

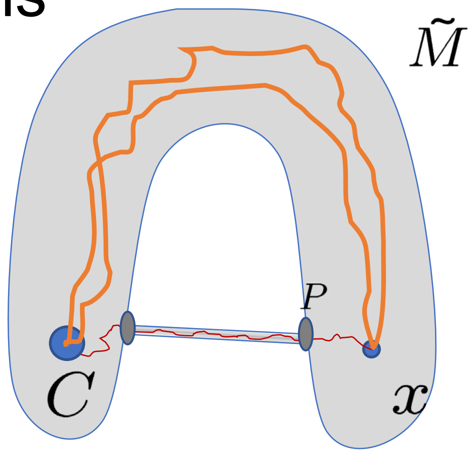


$$k_t^M(x, C) = \mathbb{P}(W_x^t \in C)$$

Heat Kernel Properties

- Robustness:

  $k_t(x, \cdot)$ is the probability density function of a Brownian motion, a weighted average over all paths, which is generally not very sensitive to local perturbations



$$k_t^{\tilde{M}}(x, C) = \mathbb{P}(\tilde{W}_x^t \in C)$$

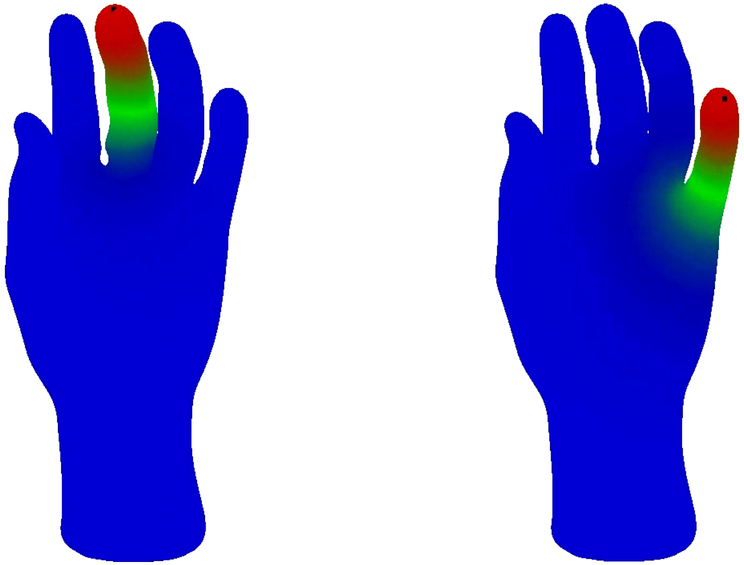
Only paths through the modified area P will change

Defining a Signature

- Let $k_t(x, \cdot)$ be the signature of x at scale t

  The heat kernel has all the properties we want

  Except easy comparison …



- $k_t(x, \cdot)$ is a function on the entire manifold
- Nontrivial to align the domains of such functions across different shapes, or even for different points of the same shape

The Heat Kernel Signature

- Let $k_t(x, \cdot)$ be the signature of x at scale t

  The heat kernel has all the properties we want.

  Except easy comparison …

- We define the Heat Kernel Signature (HKS), by restricting to the diagonal:

$$\text{HKS}(x) = \{k_t(x, x), t \in \mathbb{R}^+\}$$

- Now HKSs of two points can be easily compared since they are defined on a common domain (time)

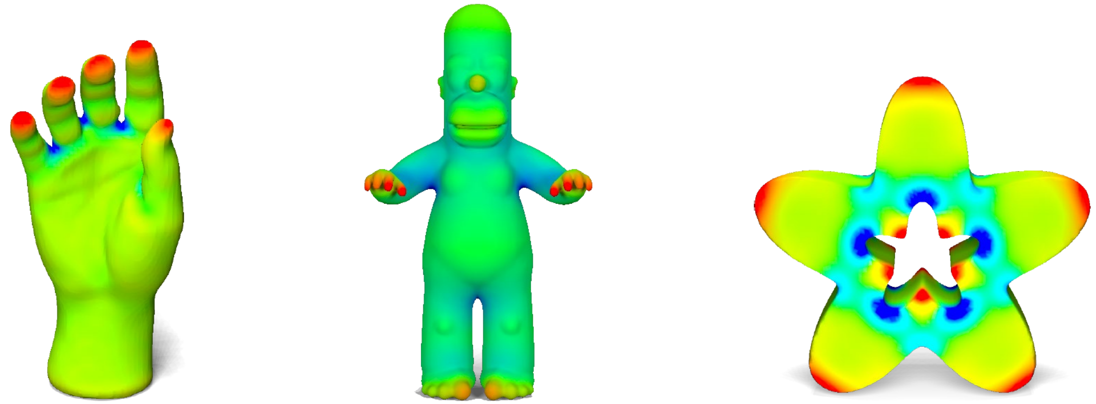
[Sun, Ovsjanikov, G., 2009]

Defining a Signature

- Since HKS is a restriction of the heat kernel, it is:
  - Robust
  - Multiscale

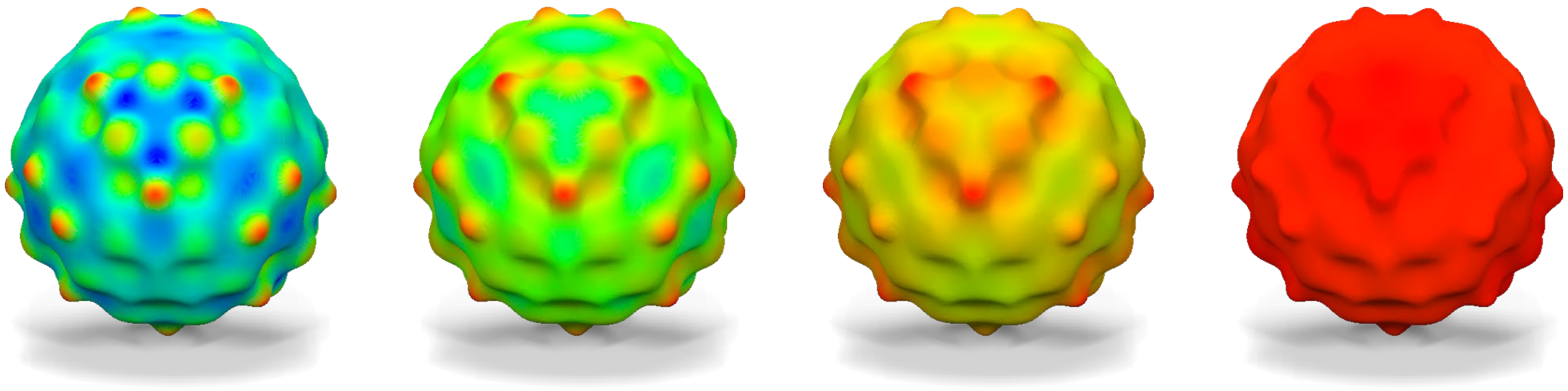
- Question: How informative is it?
  - Related to Gaussian curvature for small t :

$$k_t(x,x) = \frac{1}{4\pi t} \sum_{i=0}^{\infty} a_i t^i \qquad a_0 = 1, a_1 = \frac{1}{6} K$$



Defining a Signature

- HKS can be interpreted as a multiscale, robust, intrinsic curvature:



$t = 0.004$ $t = 0.008$ $t = 0.02$ $t = 2$

Theory Perspective: Informative Theorem

- The set of all HKSs on a shape almost always defines it up to isometry!

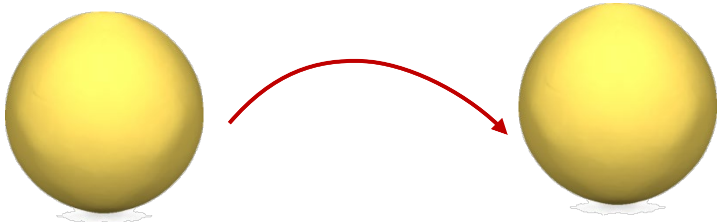
- Theorem: If X and Y are two compact manifolds, such that Δ_X and Δ_Y have only non-repeating eigenvalues, then a homeomorphism $T : X \to Y$ is an isometry if and only if, for all x

$$\mathsf{HKS}(x) = \mathsf{HKS}(T(x))$$

- The set of all HKSs characterizes the intrinsic structure of the manifold

Theory Perspective: Informative Theorem

- How general is the theorem?
  - If there are repeated eigenvalues, it does not hold:



On the sphere, $\text{HKS}(x) = \text{HKS}(y) \ \forall \ x, y$ but there are non-isometric maps between spheres.

- Uhlenbeck's Theorem (1976): for "almost any" metric on a 2-manifold X, the eigenvalues of Δ_X are non-repeating

Informative Theorem

- Heat kernel is related to the eigenvalues and eigenfunctions of the LB-operator:

$$\mathrm{HKS}(x,t) = \sum_{i=0}^{\infty} e^{-\lambda_i t} \phi_i^2(x)$$

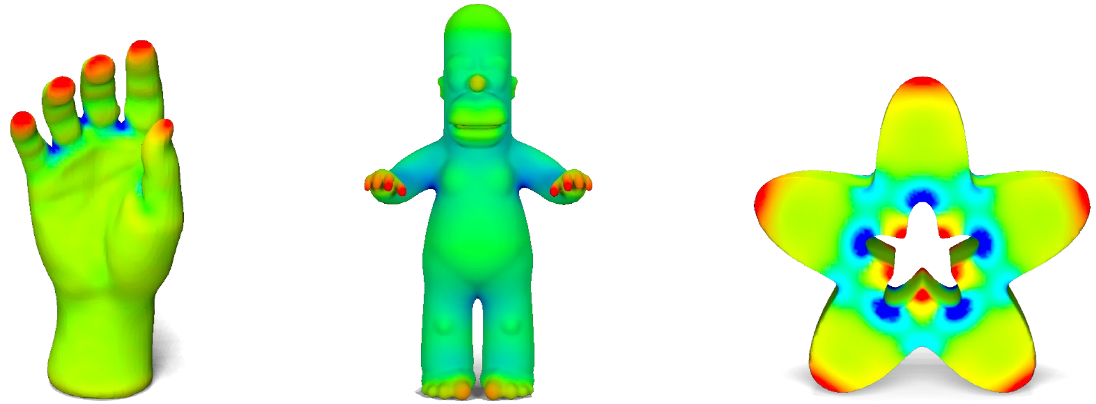
- Invariant to rotations within the eigenspace

Informative Property

- Conclusion:
  - HKS is informative for individual points
  - And, as a set, for the entire shape

Can be used both for multiscale point matching and for shape comparison

$$\text{HKS}(x) = \{k_t(x,x), t \in \mathbb{R}^+\}$$



Multiscale Matching

- Two heuristics for making HKSs commensurable:
  - For a fixed point x, sample HKS on a logarithmic scale at times t_i
  - For a fixed time t scale each HKS, by the sum over all points of *M*

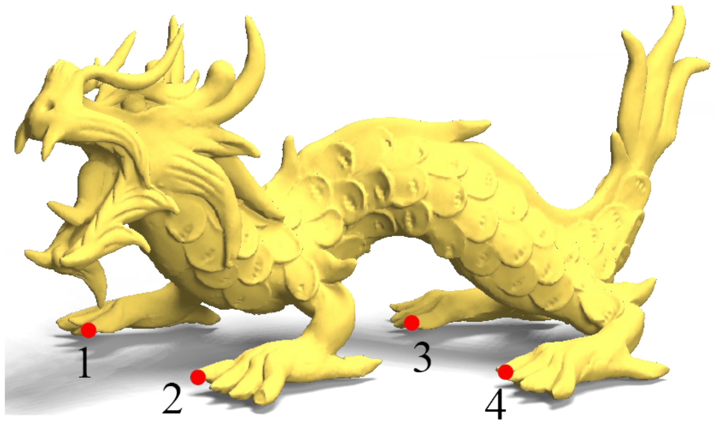
$$\text{HKS}(x) = \left\{ \frac{k_{t_i}(x,x)}{\sum_j e^{-t_i \lambda_j}}, i \in 1, 2, .., 100 \right\}$$

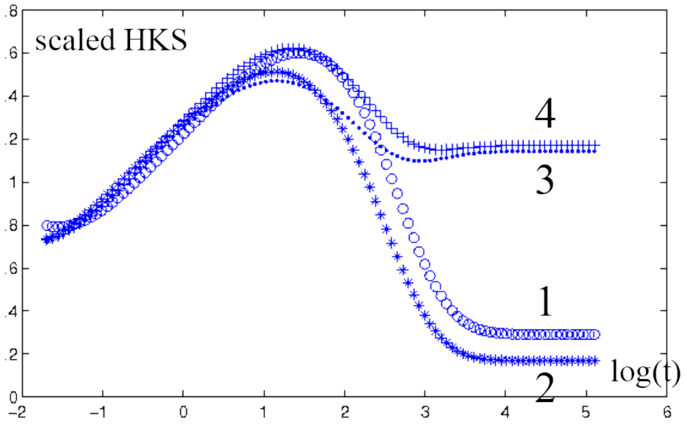
$$t_i = \alpha^i t_0$$

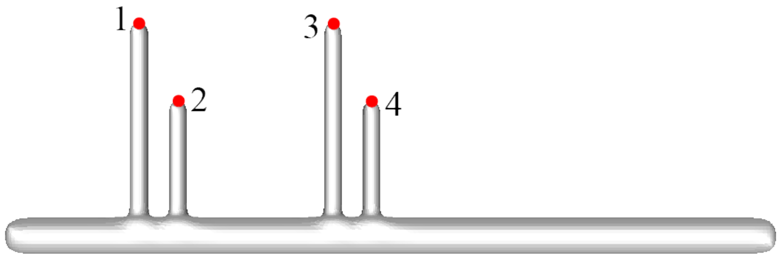
- Compare using L2 norm of the HKS vectors.

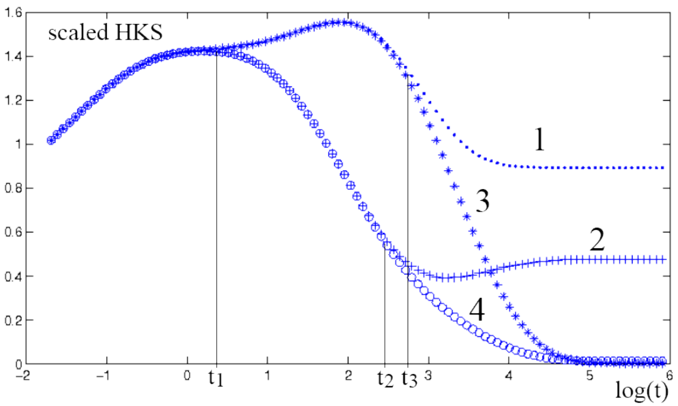
Multiscale Matching

- Comparing points through their HKS signatures:





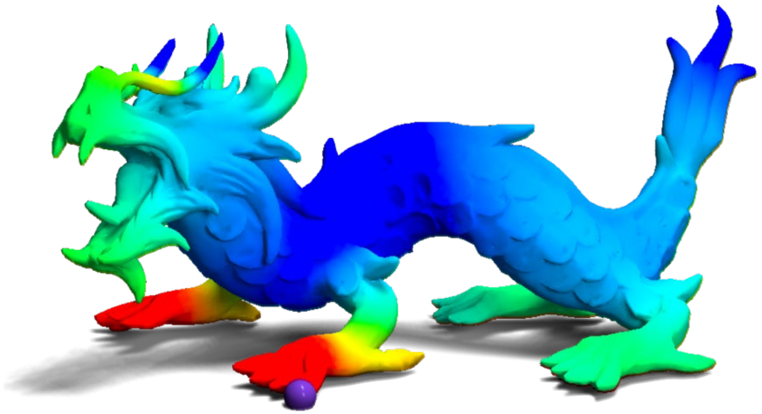




Multiscale Matching

- Comparing points through their HKS signatures:

Medium scale



Full scale

Multiscale Matching

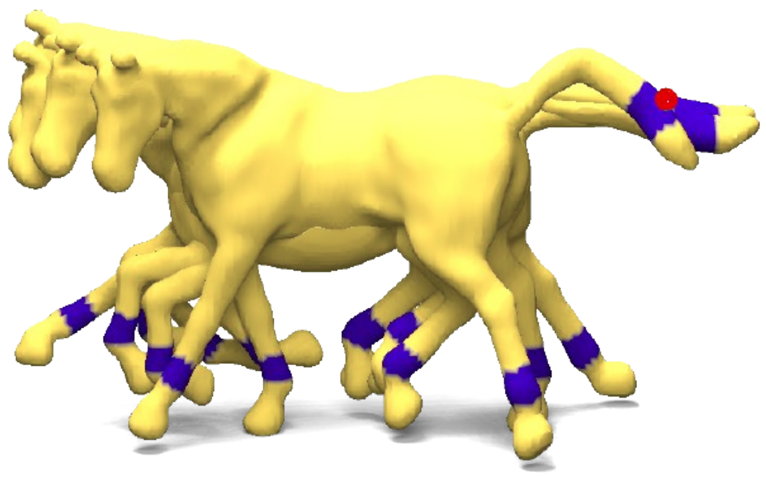
- Finding similar points – robustly:

Medium scale

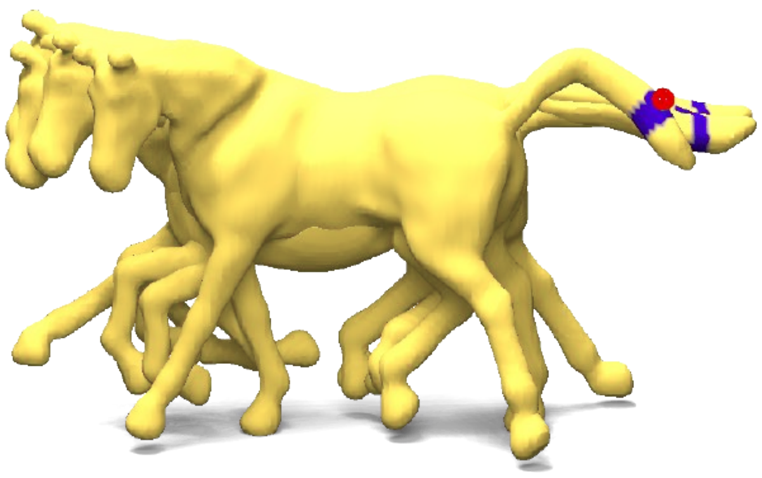
Full scale

Multiscale Matching

- Finding similar points across multiple shapes:



Medium scale



Full scale

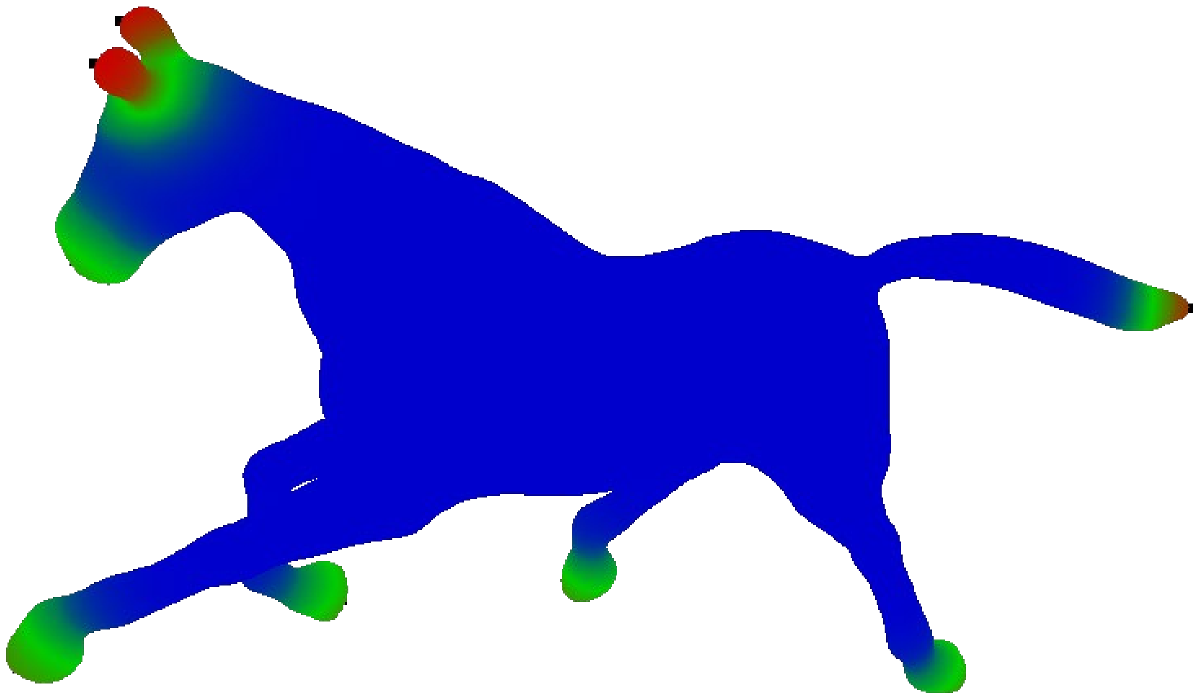
Feature Detection

- Persistent feature detection:
  - Intuition: heat diffuses slower at points with high curvature. Heat will tend to concentrate in “hot spots” – extremities of the surface
  - Approach: track the local maximum of the heat kernel for increasing t

Feature Detection

- Persistent feature detection:
  - Find points that are long term maxima of their heat kernels:

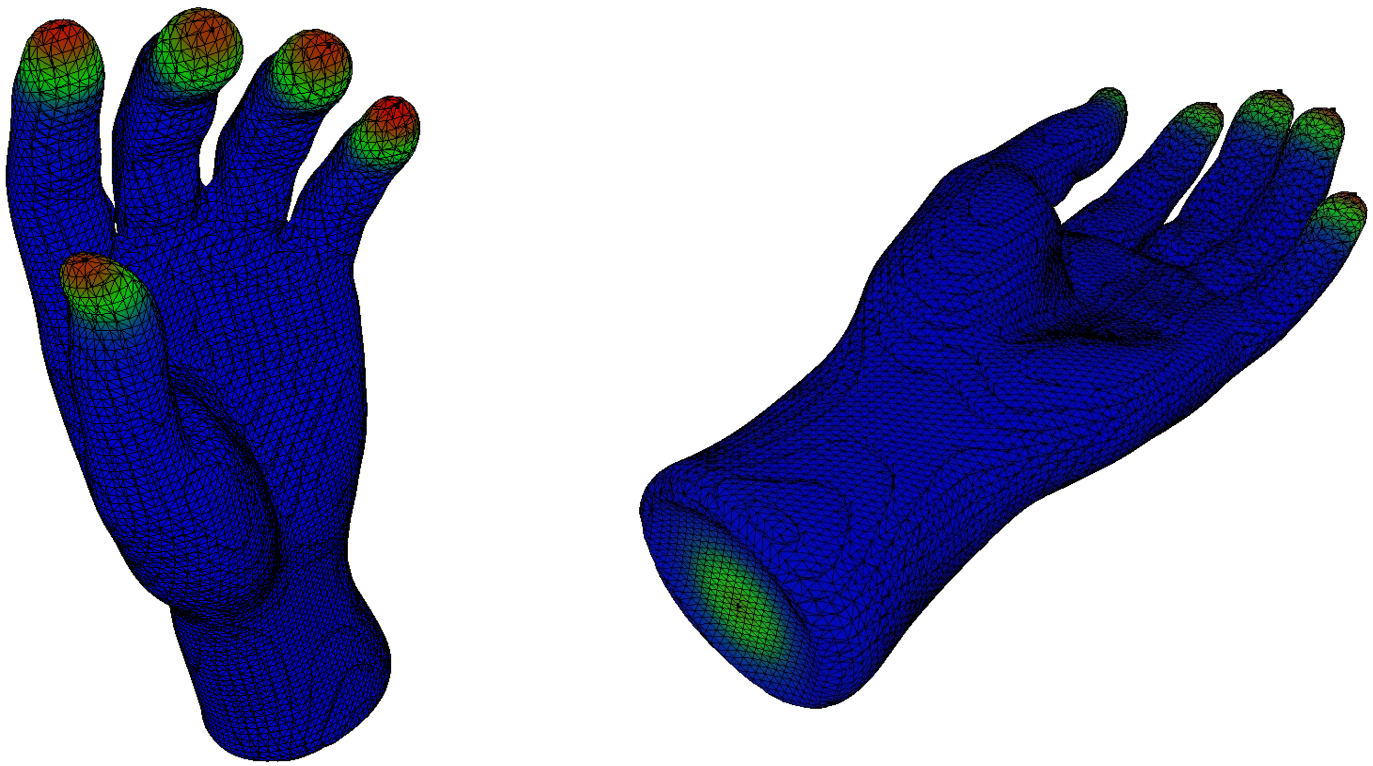
$$k_t(x, \cdot)$$



Feature Detection

- Persistent feature detection:
  - Find points that are long term maxima of their heat kernels:

$$k_t(x, \cdot)$$



Feature Detection

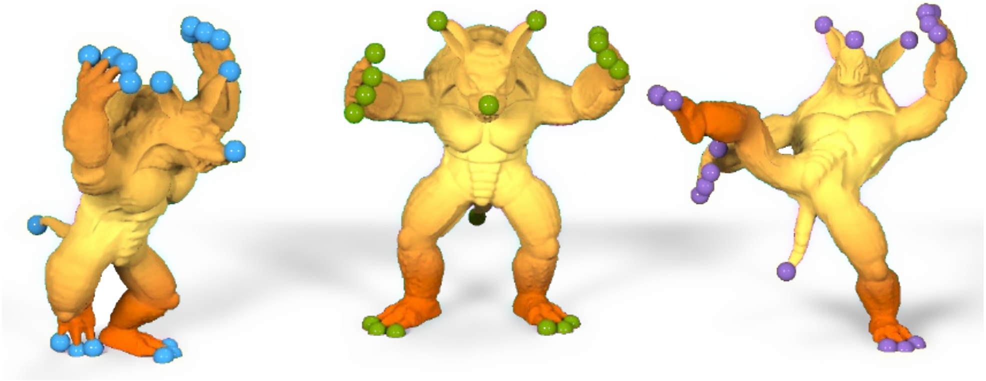
- Persistent feature detection:
  - Find points that are long term maxima of their heat kernels:

$$k_t(x, \cdot)$$

  - This may be expensive since the heat kernel at every point is a function over the whole shape. However, long term behavior at nearby points is similar due to mixing
  - Approximation: find points that are local maxima of

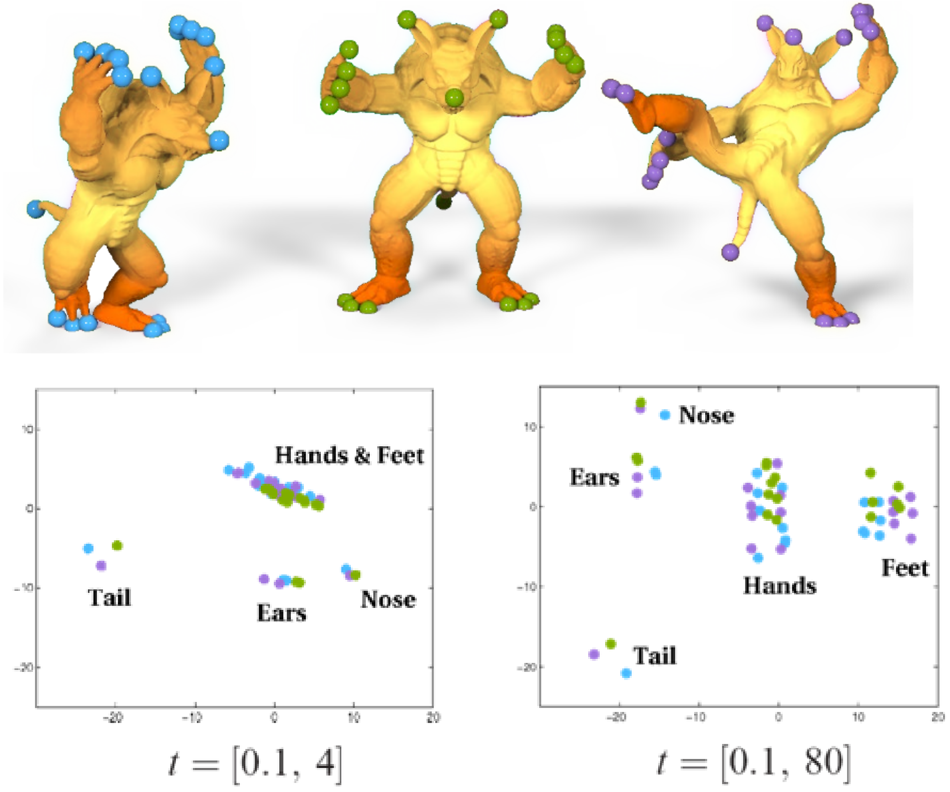
$$k_t(x, x)$$

    for large enough t



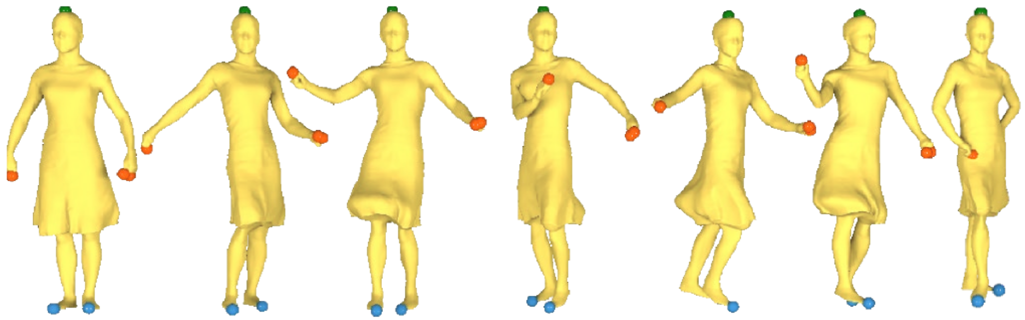
Shared Structure

- 2D MDS embedding of feature points on three shapes according to distances of their HKS

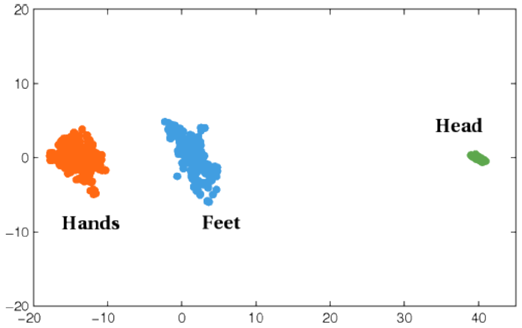


Shared Structure

- 2D MDS embedding of feature points on 175 shapes according to distances of their HKS.

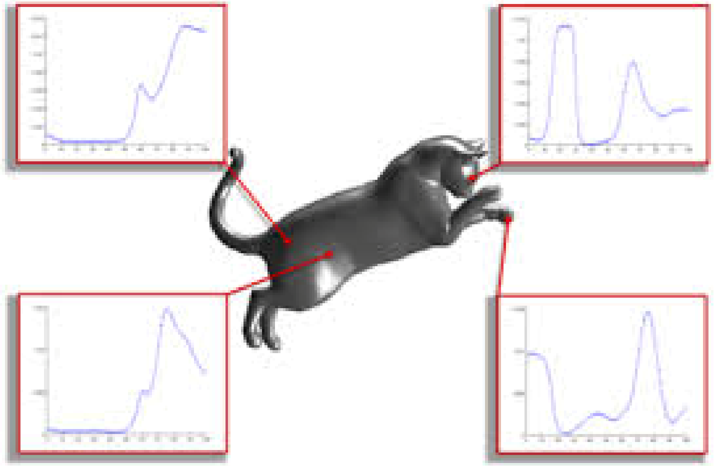


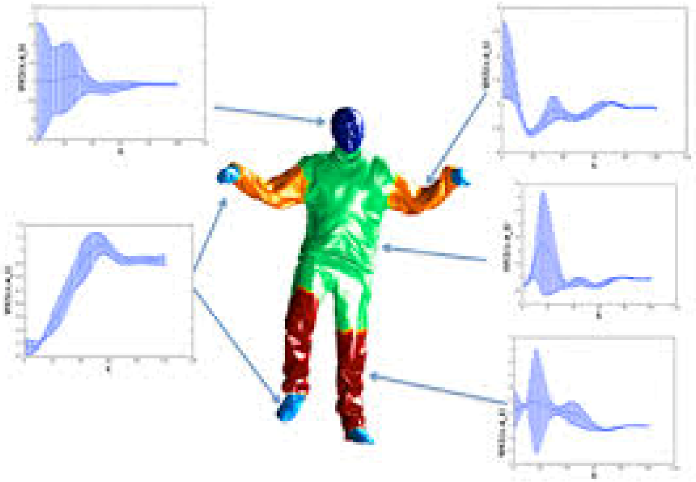
Feature points found on a few poses of the dancer model by Vlasic *et al.*



MDS of features from all 175 poses using a full range of scales

The Wave Kernel Signature (WKS)





Based on solutions of the quantum Schrödinger equation

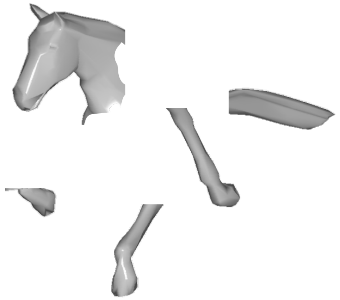
$$\frac{\partial \psi}{\partial t}(x,t) = i\Delta\psi(x,t)$$

Shape Search

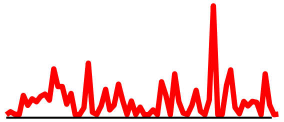
Bag-of-Words Models (BoW)

Visual Object → Bag of ‘words’

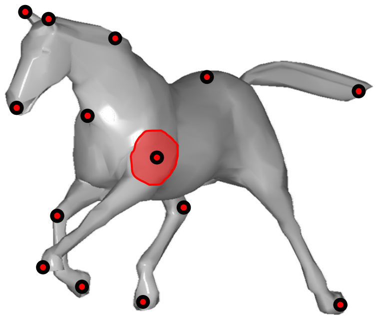
Spatial Words



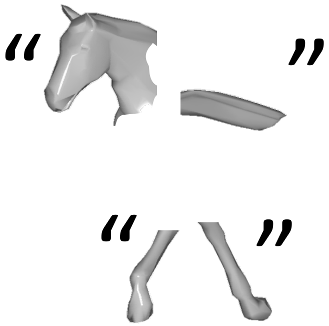
Geometric words



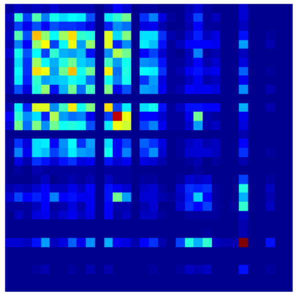
Bag of geometric words



Feature descriptor

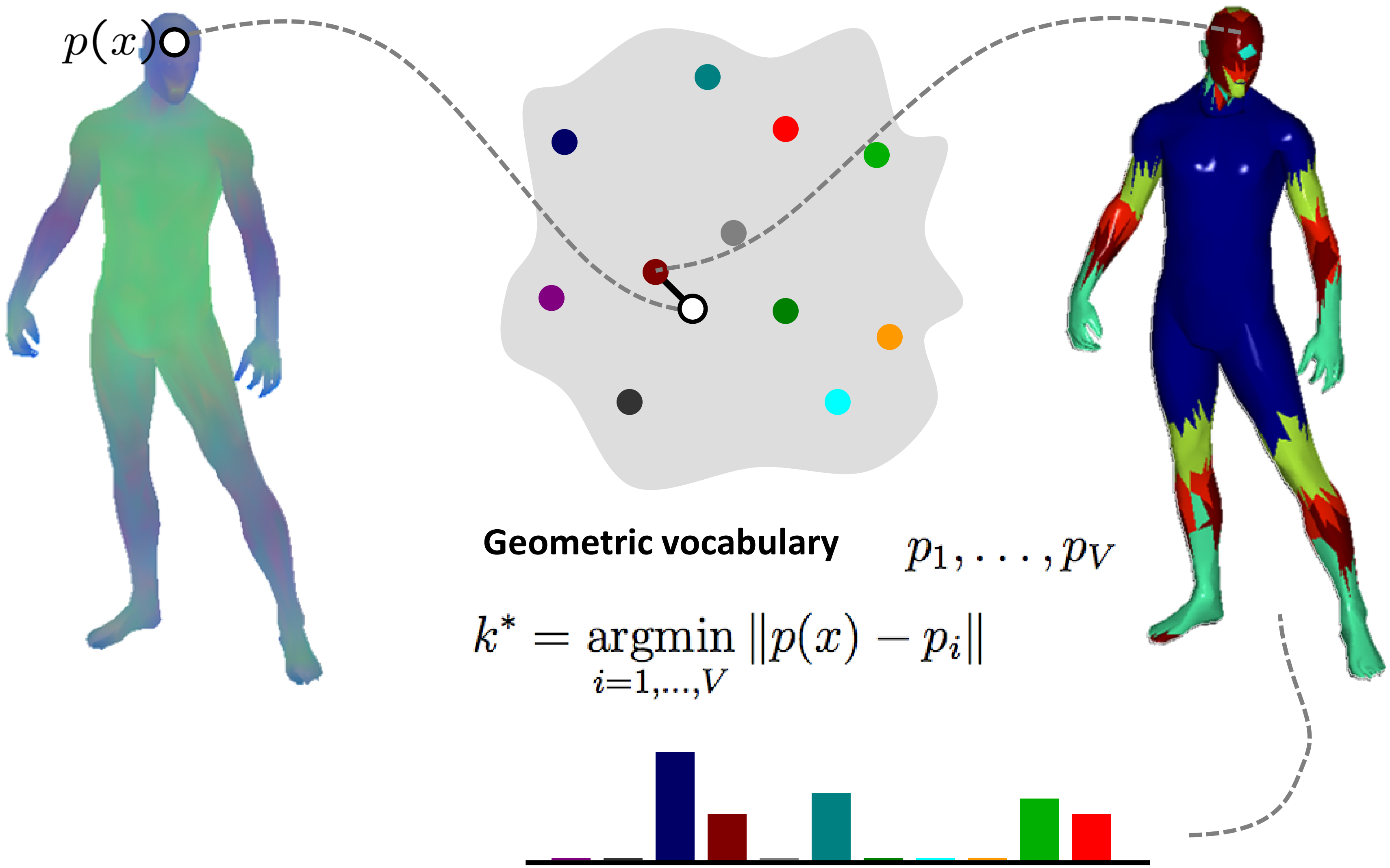


Geometric expressions

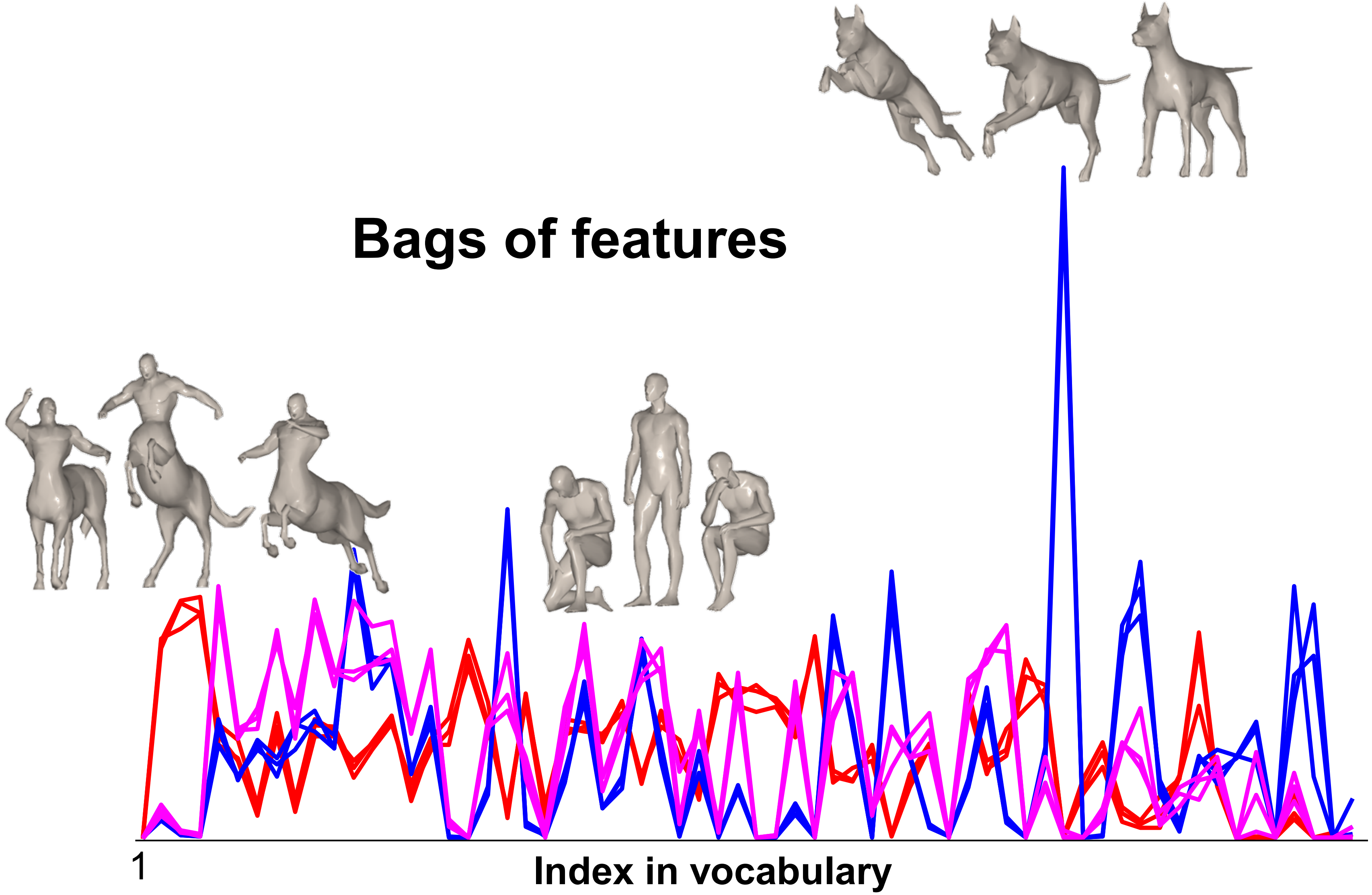


Spatially-sensitive bag of words

ShapeGoogle: HKS-Based BoW Shape Search



Shape Signatures



The End