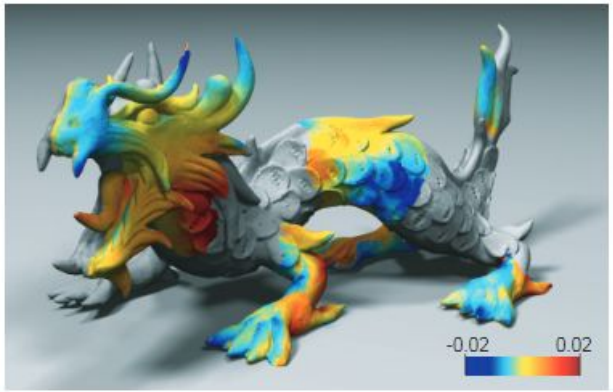
# Partial and Approximate Symmetry Detection for 3D Geometry [2006]

Niloy J. Mitra
Stanford University

Leonidas J. Guibas
Stanford University

Mark Pauly
ETH Zürich

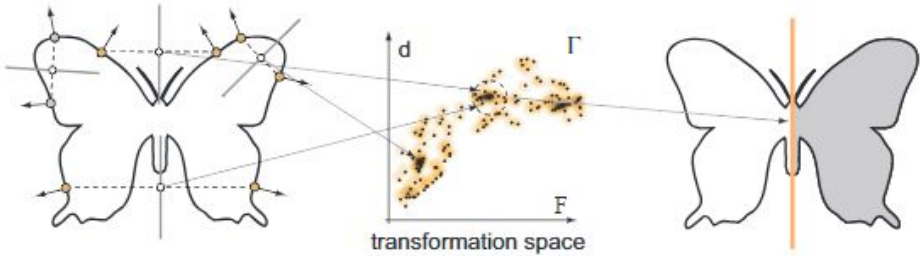


# Outline

1. Introduction
2. Signatures and Transformations
3. Clustering
4. Verification
5. Results and Application
6. Conclusion

# Problem Definition

- Importance of symmetry
- Goal is to extract symmetric relations
- Uses:
  - Scan registration, shape matching
  - Segmentation and skeleton extraction
  - Shape Database Retrieval
  - Advanced modeling and editing
  - Compression



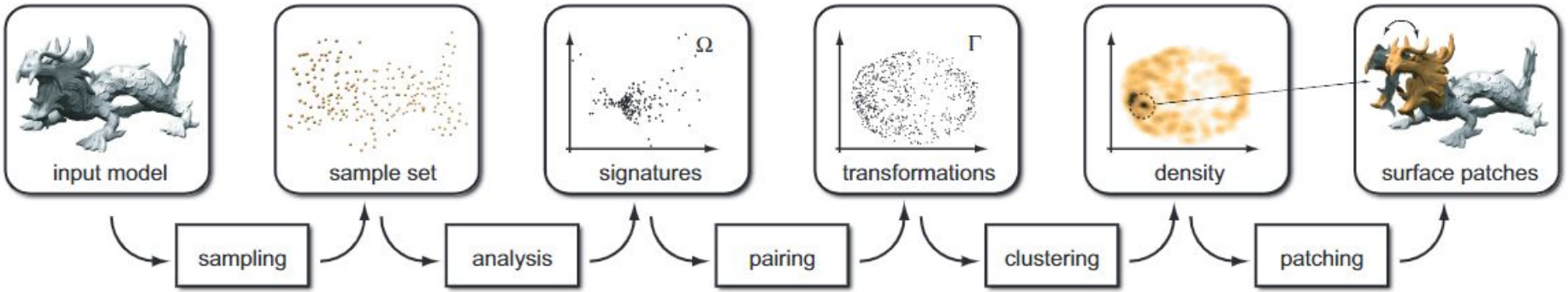
# Problem Definition




# Problem Solution

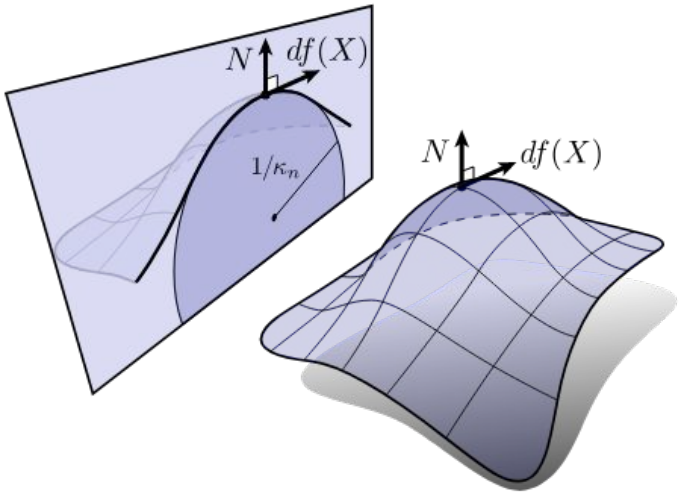
First phase
- Compute transformations of point pairs
- Evidence of symmetry

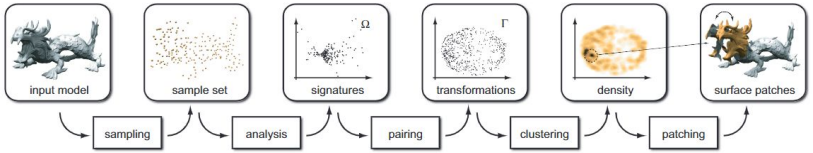
Second phase:
- Use clusters to extract patches
- Bypasses exhaustive search



# Principal Curvature

- Describes how point bends
- Notation k1 and k2
- Noticeable curvature for point mapping
- Invariant under transformations
    - Scale or pointvector

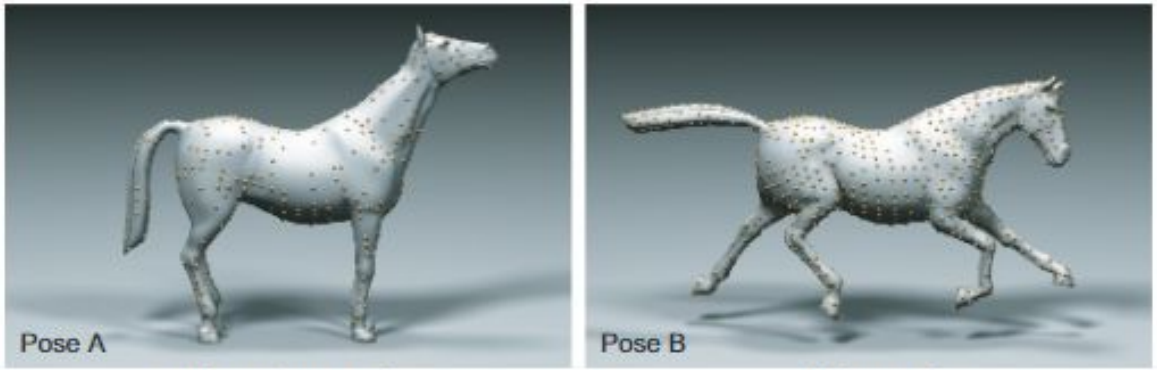




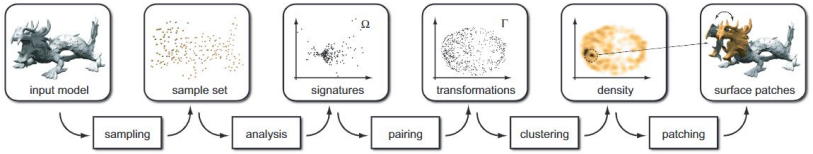



# Sampling

- Random or structured
- Creates sample set



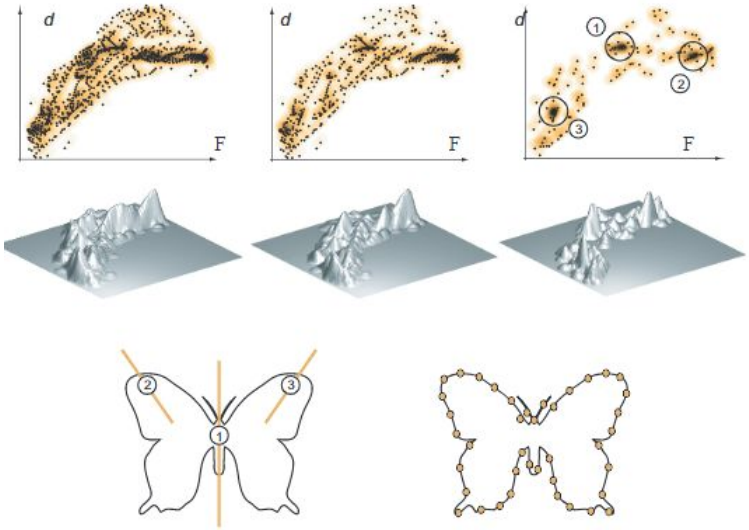

500 random samples (a) 1000 samples

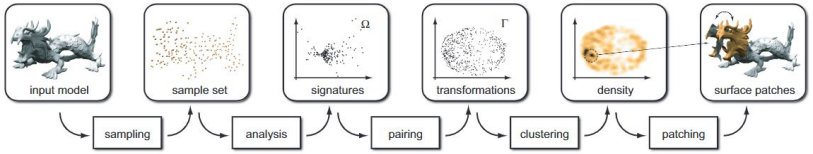





# Analysis

- *"We therefore compute geometry signatures at each sample point p based on the concept of normal cycles[Cohen-Steiner and Morvan 2003]"*
- Generates signatures based on curvature
  - Ratio: Q7(p) = k1 / k2
  - Pair: Q6(p) = (k1,k2)
  - Used for pairing
- Pruning on Q7 < 0.75
  - Avoid flat/uniform surfaces
  - Computational advantage

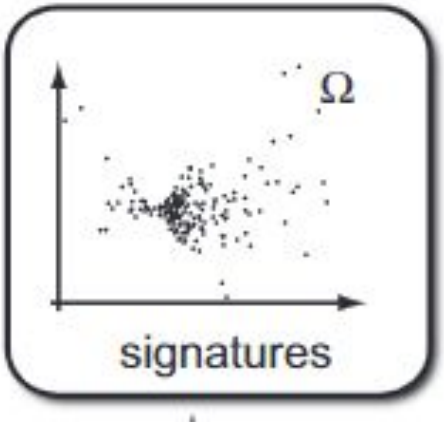


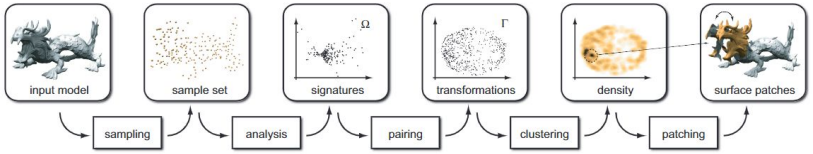





# Pairing

- All neighbours in signature space (curvature space)
  - k1/k2 for scaling, rotation, translation
  - (k1,k2) for rigid transforms
- n'log(n) using spatial proximity structures







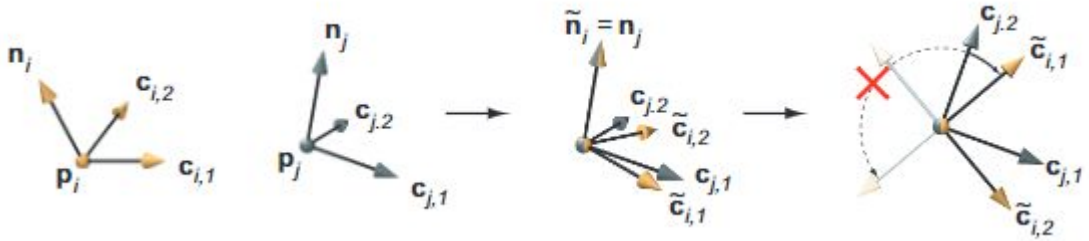

# Transformation space

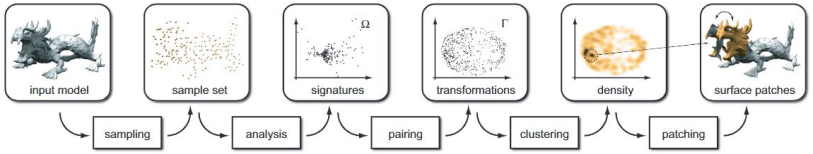
- Potential pairs are mapped
- 7 dimensions of point p(i,j)
  - Curvature scale
  - Rotation [Euler Angles]
  - Translation

$$\mathbf{T}_{ij} = (s_{ij}, R^x_{ij}, R^y_{ij}, R^z_{ij}, t^x_{ij}, t^y_{ij}, t^z_{ij})$$

$$s_{ij} = (\kappa_{i,1} / \tilde{\kappa}_{j,1} + \kappa_{i,2} / \kappa_{j,2})/2,$$

$$\mathbf{t}_{ij} = \mathbf{p}_j - s_{ij}\mathbf{R}_{ij}\mathbf{p}_i.$$





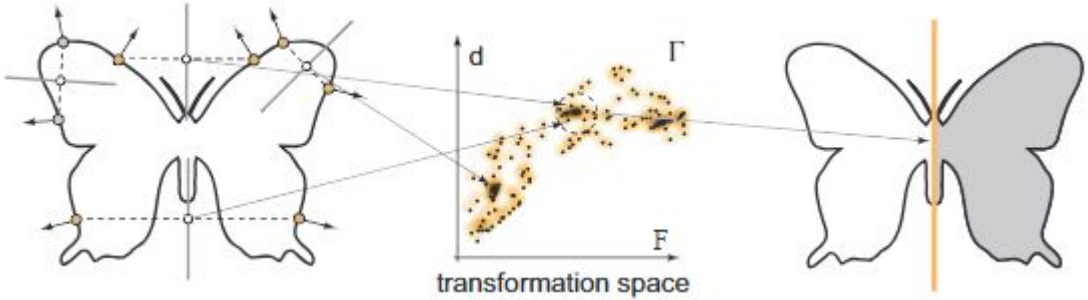


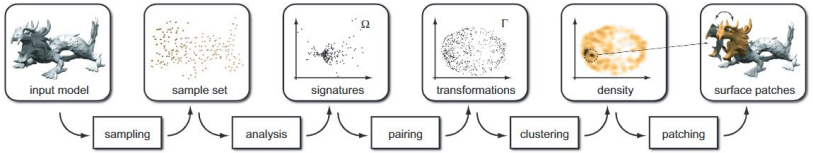

# Clustering

- Goal: Discover clusters of transformation
- Defines metrics:
  - Scale depending on transformation

$$\|\mathbf{T}\|^2 = \beta_1 s^2 + \beta_2 (R_x^2 + R_y^2 + R_z^2) + \beta_3 (t_x^2 + t_y^2 + t_z^2)$$

- Can't use k-means clustering
  - K unknown
  - Clusters not necessarily isotropic



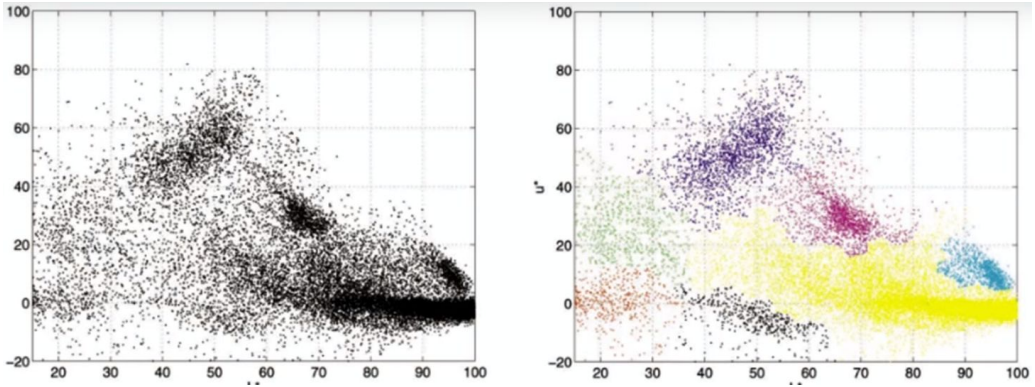


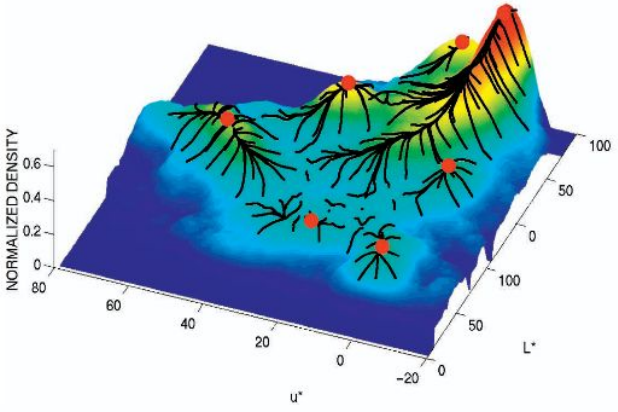


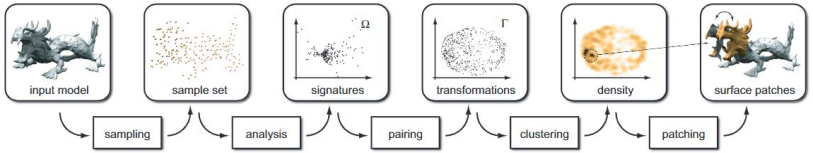

# Clustering

- mean-shift clustering
  - Based on gradient descent
  - Voting scheme
- Outputs numbers of clusters C.

$$\rho(\mathbf{T}) = \sum_i K(\|\mathbf{T} - \mathbf{T}_i\|/h).$$



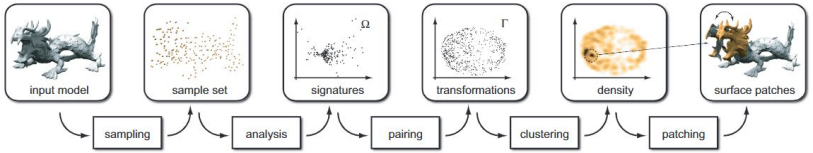







# Verification

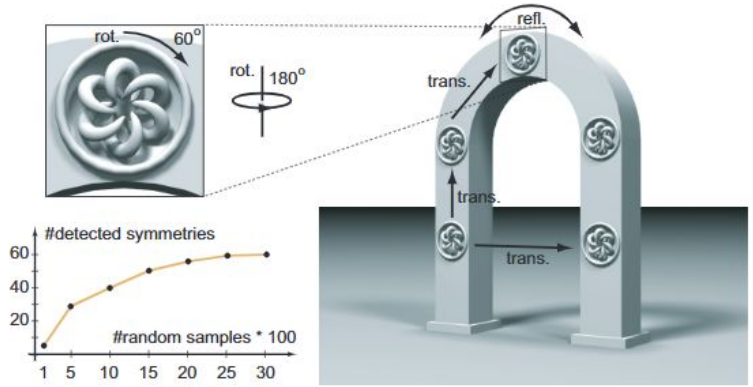
- Evidence of symmetry in clusters
- Second phase begins
- Grows the patches
- Focuses on single cluster at the time
  - Picks point, grows patch based on neighbourhood
  - Removes points added from cluster
  - Iterated Closest Point on T of patch
- Generates weighted graph of Transformations between patches






# Compound Transformations

- Algorithm for compressing the weighted graph
- Set of generators for detected symmetries
  - Advanced editing options
  - High level shape comparisons



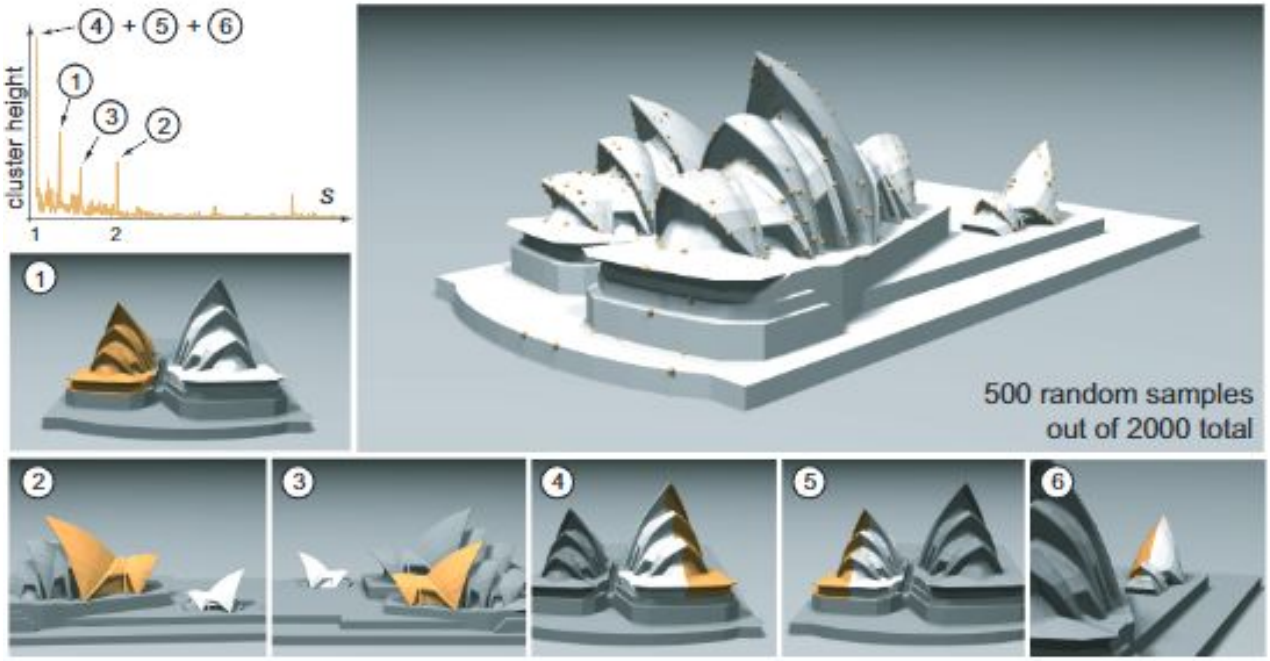
# Implementation

*"An **initial sample set** is created by uniformly sampling the input model. After computing **signatures** using the method of [Alliez et al. 2003], **point pruning** yields the reduced sample set P. We then select a **random subset** $P' \subset P$, find all suitable pairs ($p' \in P'$, $p \in P$) based on the **proximity in signature space**, and compute the corresponding **transformations**. We perform **mean-shift clustering** using the method proposed in [Arya et al. 1998] to efficiently compute neighborhoods in 7D transformation space. **Basis reduction** and **verification** finally yield the **symmetric patches**."*

# 1th figure

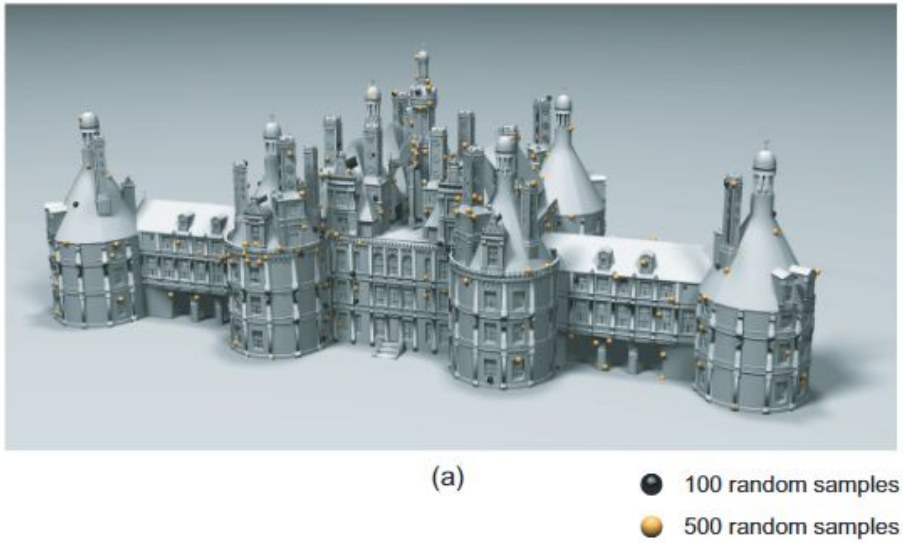
- Approximate symmetry
- Symmetry deviation

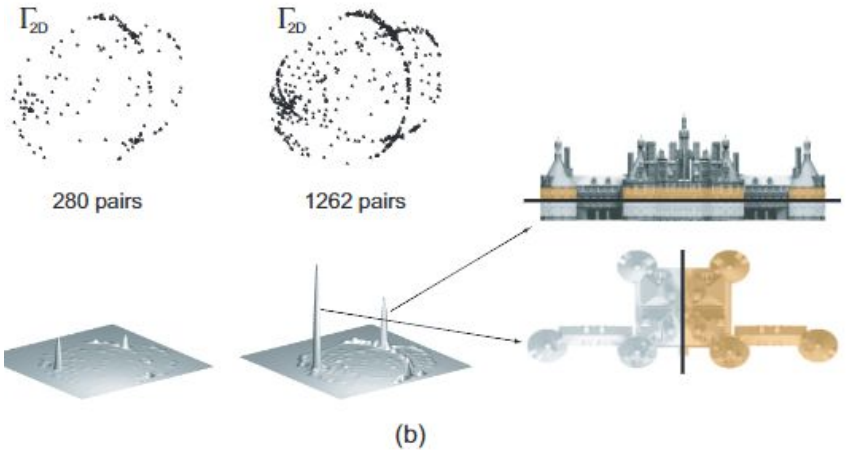
# 2nd figure



# 3rd figure

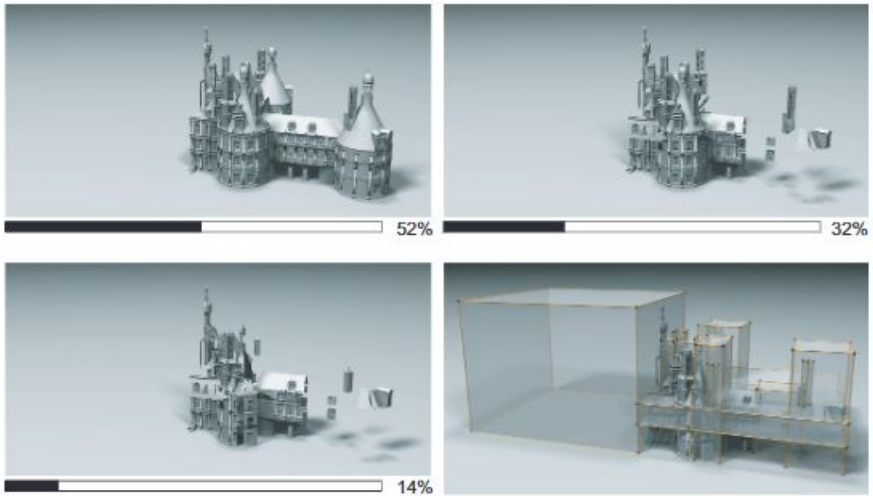
- Automatic model reduction
- Modification on specific parts



# 3rd figure

- Automatic model reduction
- Modification on specific parts

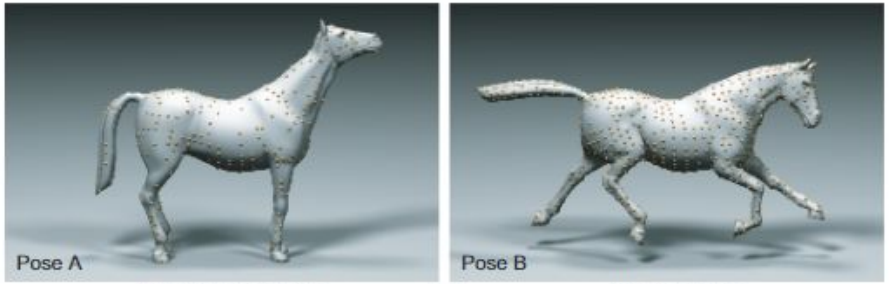


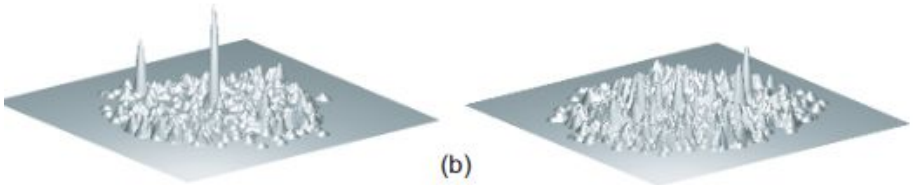

(c)

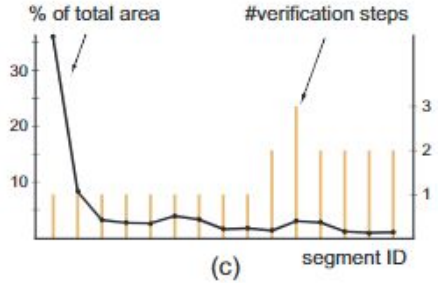
(d)

# 4th figure

- Symmetry between different models



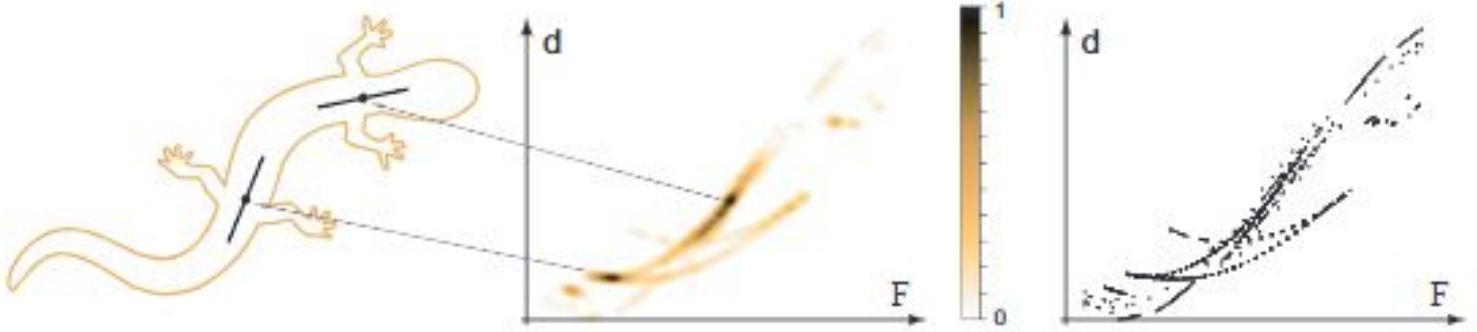






# 5th figure

- Difficulties of continuous symmetries



# Table of comparison

- 2.8 GHz Pentium IV with 2GBytes main memory
- Emphasis on symmetric complexity

| Model | # Vertices | Sign. | Pairing | Cluster. | Verif. |
|---|---|---|---|---|---|
| Dragon | 160,947 | 3.44 | 49.24 | 13.63 | 7.45 |
| Opera | 9,376 | 0.96 | 0.02 | 0.03 | 0.86 |
| Castle | 172,606 | 5.61 | 117.81 | 159.73 | 5.63 |
| Horse | 8,431 | 0.92 | 0.01 | 0.01 | 1.63 |
| Arch | 16,921 | 0.08 | 5.86 | 26.89 | 2.42 |

# Conclusion

- Briefly revisits how well system worked
- Future work:
  - Make model more symmetric automatically
  - Potential within partial scan alignment