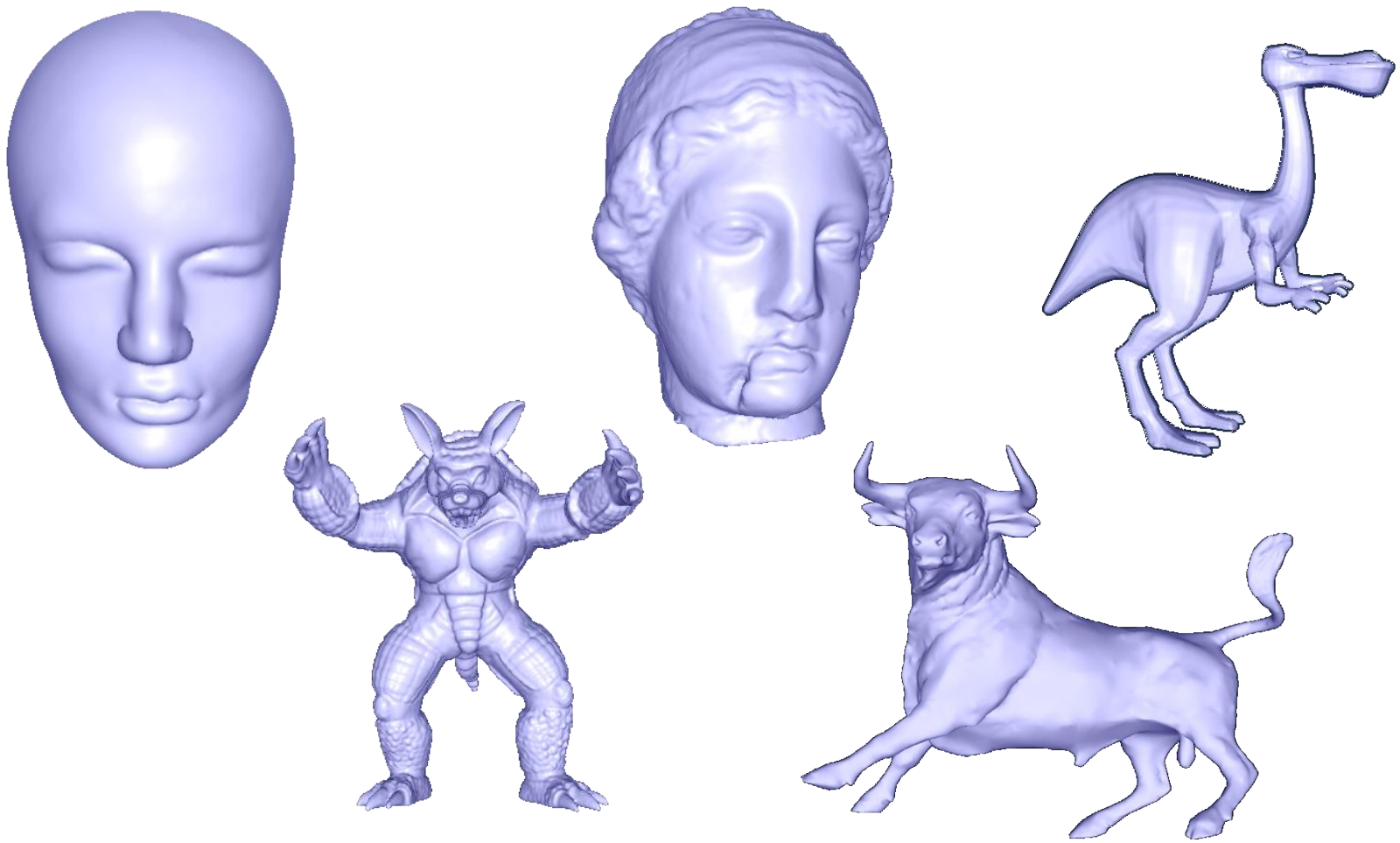
Symmetry Analysis and its Applications in Computer Graphics

Thomas Funkhouser
Alex Golovinskiy, Misha Kazhdan,
Vladimir Kim, Josh Podolak,
and Szymon Rusinkiewicz

Princeton University

Motivation

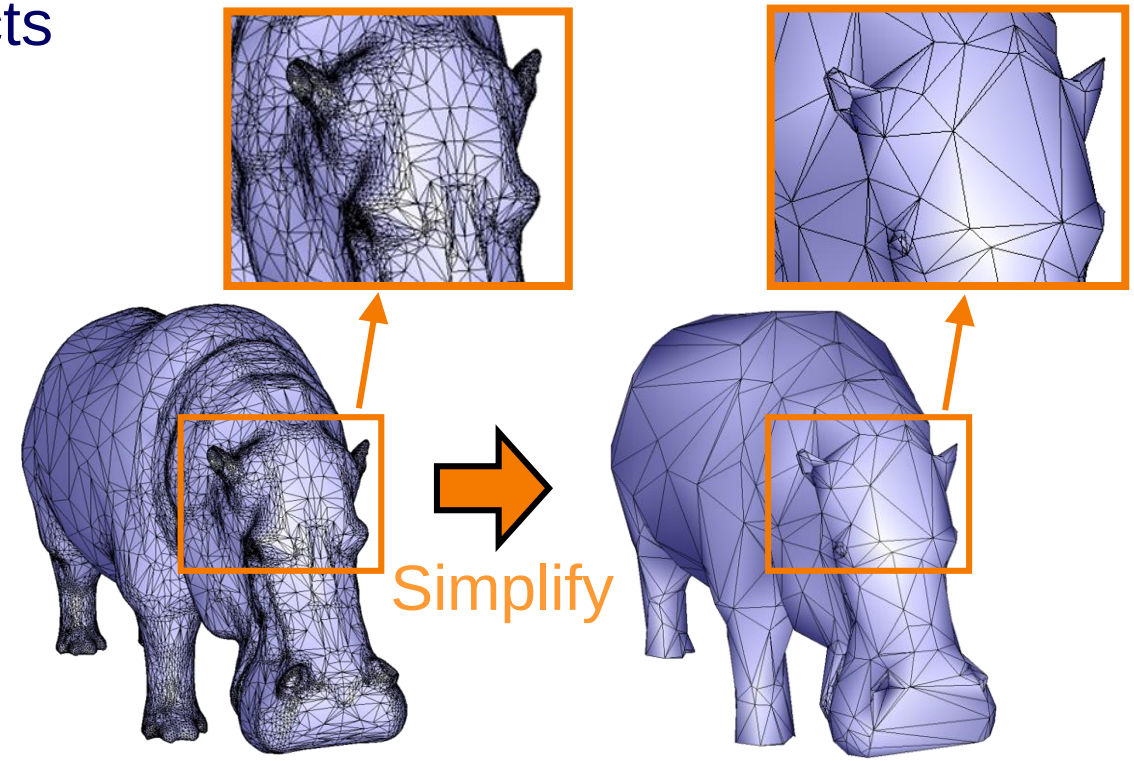
Symmetry is common in real-world objects



Problem

Most algorithms ignore (and sometimes destroy) symmetries when they process 3D models

- Rendering artifacts
- Simulation errors
- etc.



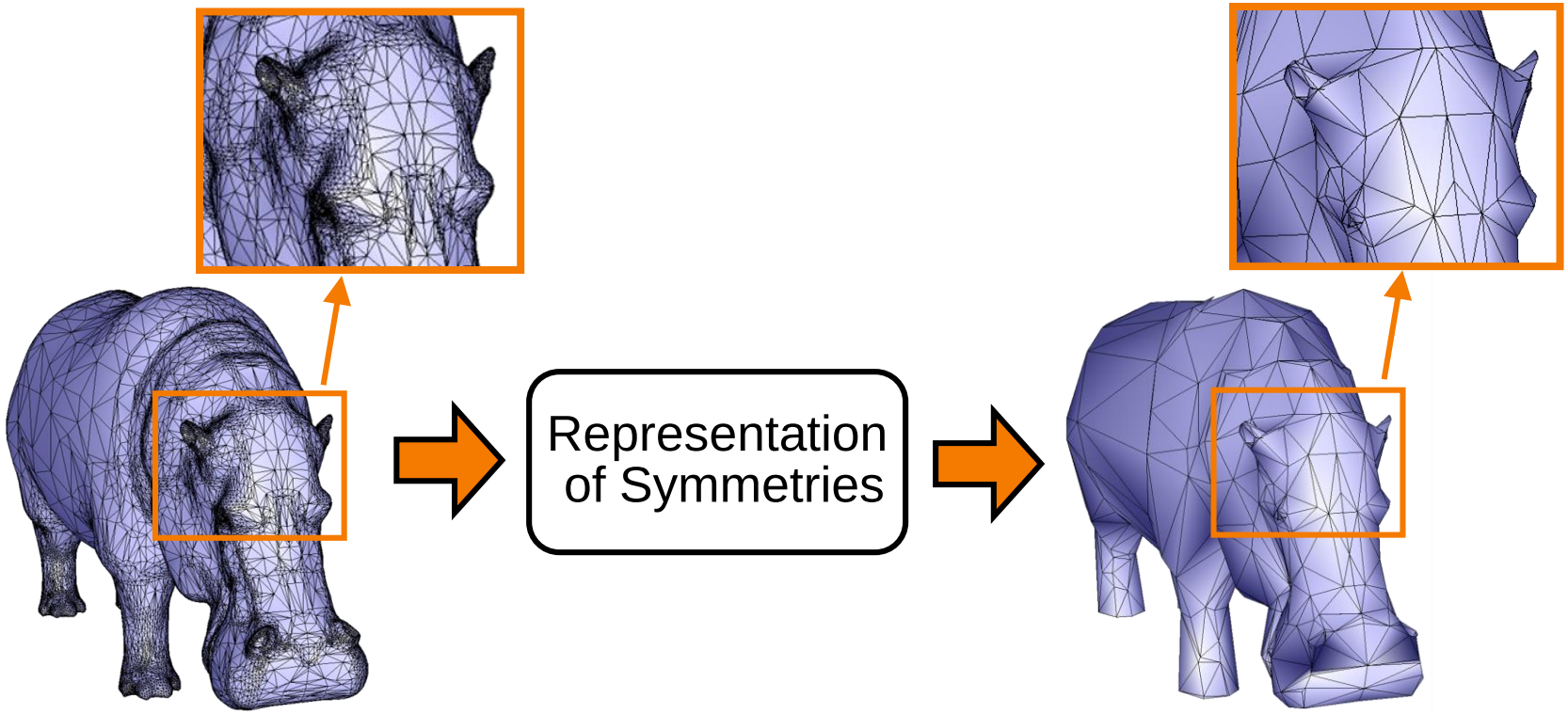
Symmetric Input Mesh

Asymmetric Output Mesh

[Cohen-Steiner04]

Goal

We need methods to detect, analyze, represent, and exploit symmetries in 3D models



Symmetric Input Mesh

Symmetric Output Mesh

Outline

Introduction

Background

Symmetry representations

- Symmetry descriptor
- Symmetry transform
- Principal symmetries

Applications

Conclusion

Outline

Introduction

Background ←

Symmetry representations

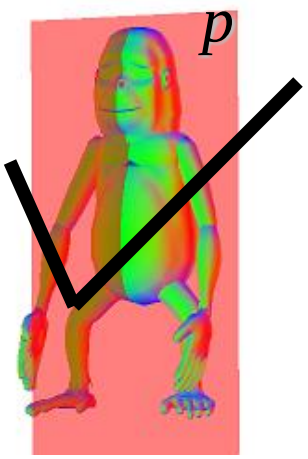
- Symmetry descriptor
- Symmetry transform
- Principal symmetries

Applications

Conclusion

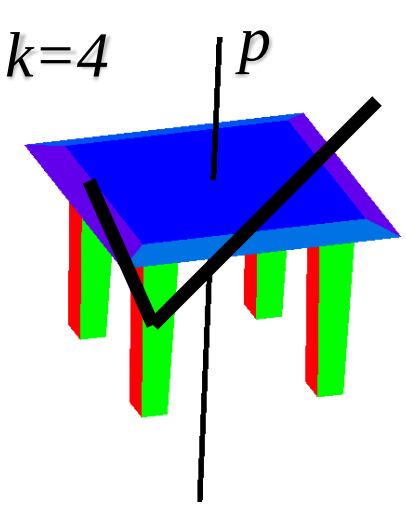
Symmetry

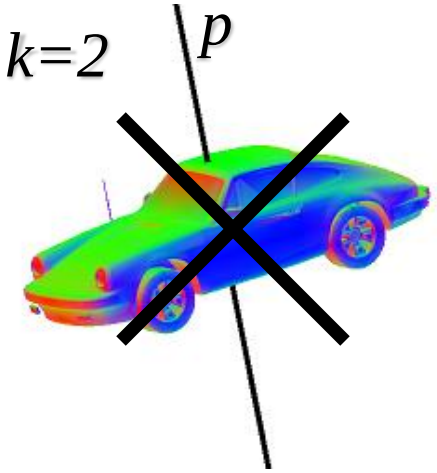
A shape has *reflective symmetry* w.r.t. some plane *p* if the reflection Ref_p through *p* fixes the collection



Symmetry

A shape has *rotational symmetry* of order k w.r.t. some axis p if the rotation Rot_p^k by an angle of $360^o/k$ about p fixes the collection.



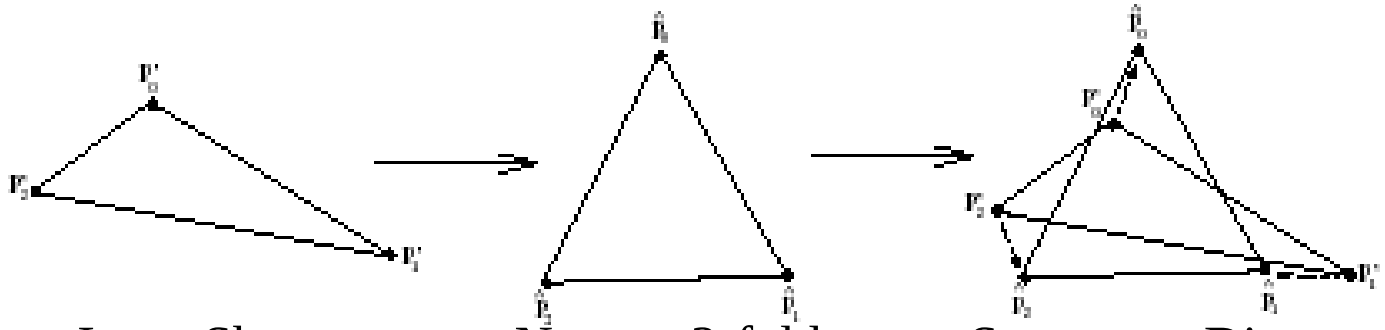


Approximate Symmetry

… but many shapes in computer graphics are not perfectly symmetric

Symmetry Distance

The *symmetry distance* of a boundary is the L^2 distance to the nearest symmetric boundary



Input Shape | Nearest 3-fold symmetric | Symmetry Distance

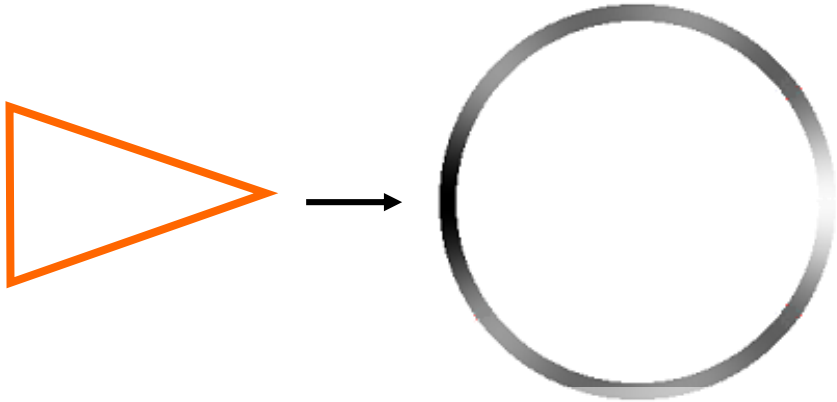
Requires point correspondences

IWVF, 1994. Zabrodsky et al.

IEEE, 1995. Zabrodsky et al.

Symmetry Distance

For circular functions, can replace distances between correspondences with correlation



2D Shape Function

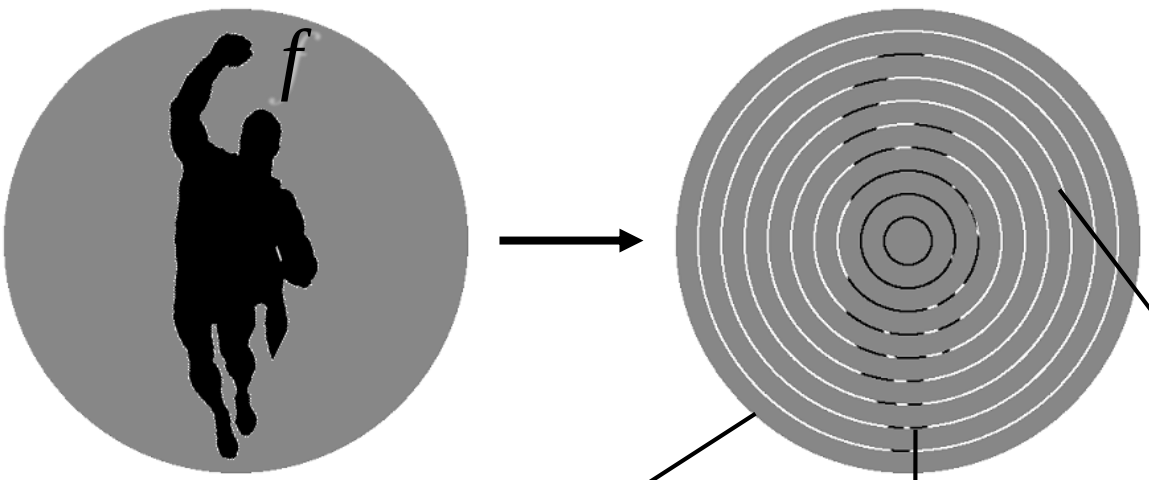
$$Sym(SD,\alpha) = \int_0^{2\pi} SD(t-\alpha)SD(t)dt$$

PRL, 1995. Sun

RTI, 1999. Sun et al.

Symmetry Distance

For images, can consider circular restrictions and compute correlations efficiently with Fourier transform



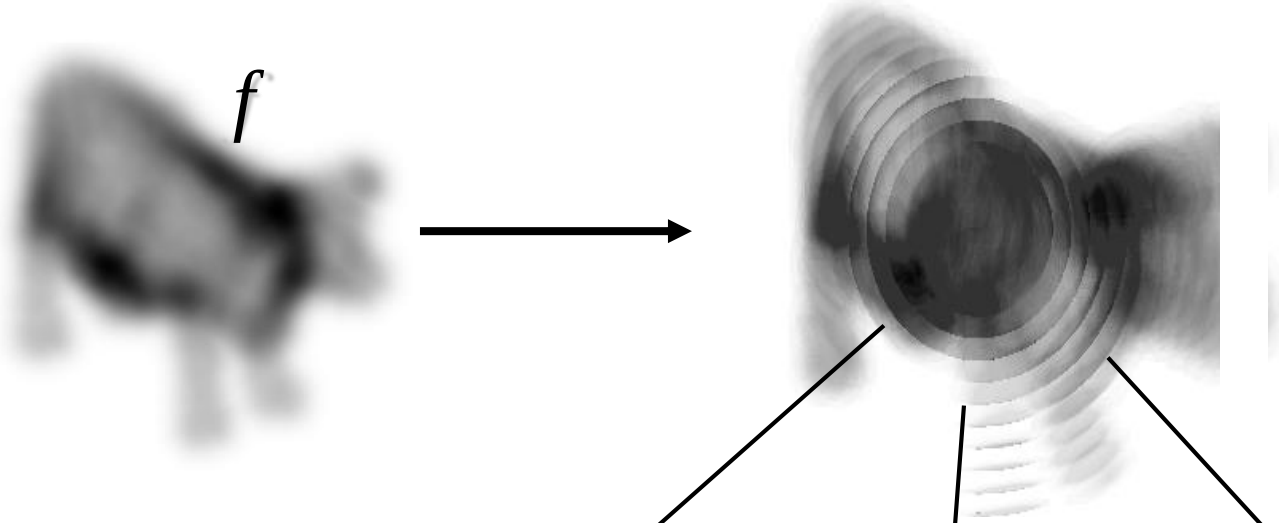
$$Sym(f,\gamma) = \overbrace{r_1}^{\text{scale factor}} \; Sym(\;,\gamma) + \overbrace{r_2}^{\text{scale factor}} \; Sym(\;,\gamma) + \overbrace{r_3}^{\text{scale factor}} \; Sym(\;,\gamma) + \ldots$$

PRL, 1995. Sun

RTI, 1999. Sun et al.

Symmetry Distance

For 3D functions, can consider spherical restrictions and compute correlation efficiently with spherical harmonics and Wigner-D^{-1} transform

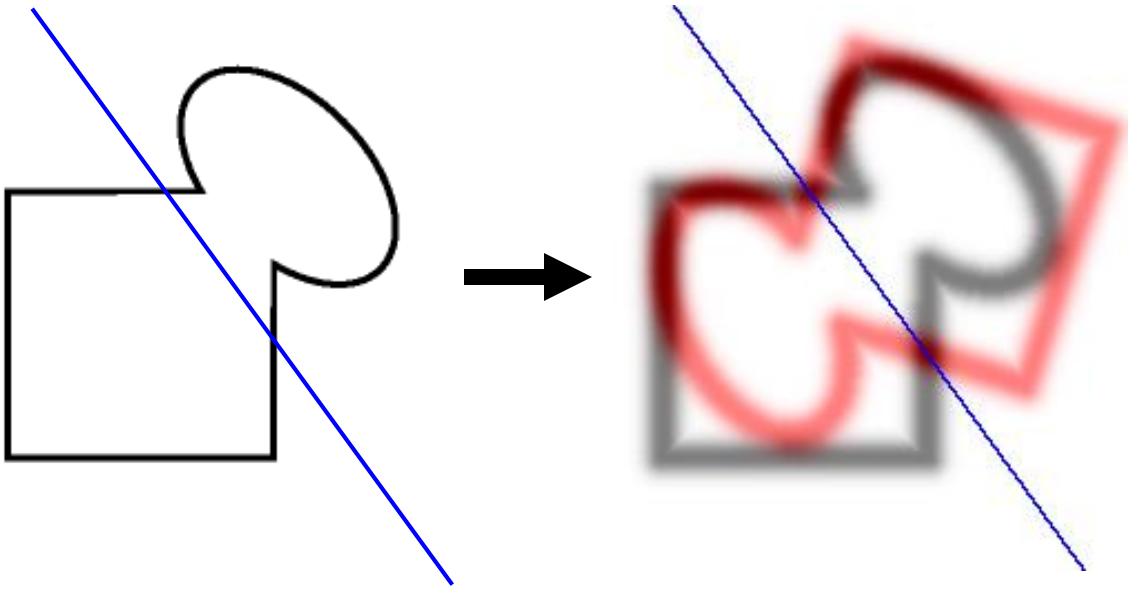


$$Sym(f,R) = \overbrace{r_1}^{\text{scale factor}} \; Sym(\;,R) + \overbrace{r_2}^{\text{scale factor}} \; Sym(\;,R) + \overbrace{r_3}^{\text{scale factor}} \; Sym(\;,R) + \ldots$$

ECCV, 2002. Kazhdan et al.

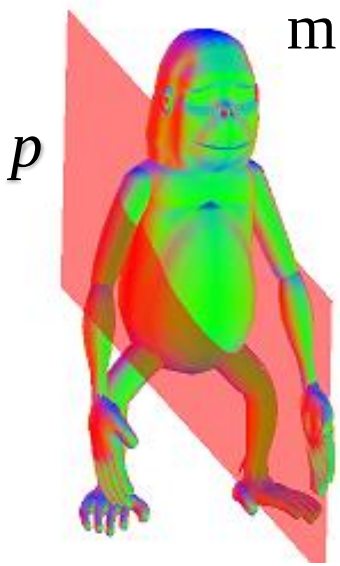
Symmetry Distance

For boundaries, can approximate symmetry distance by converting to a function and then computing correlation



Symmetry Distance

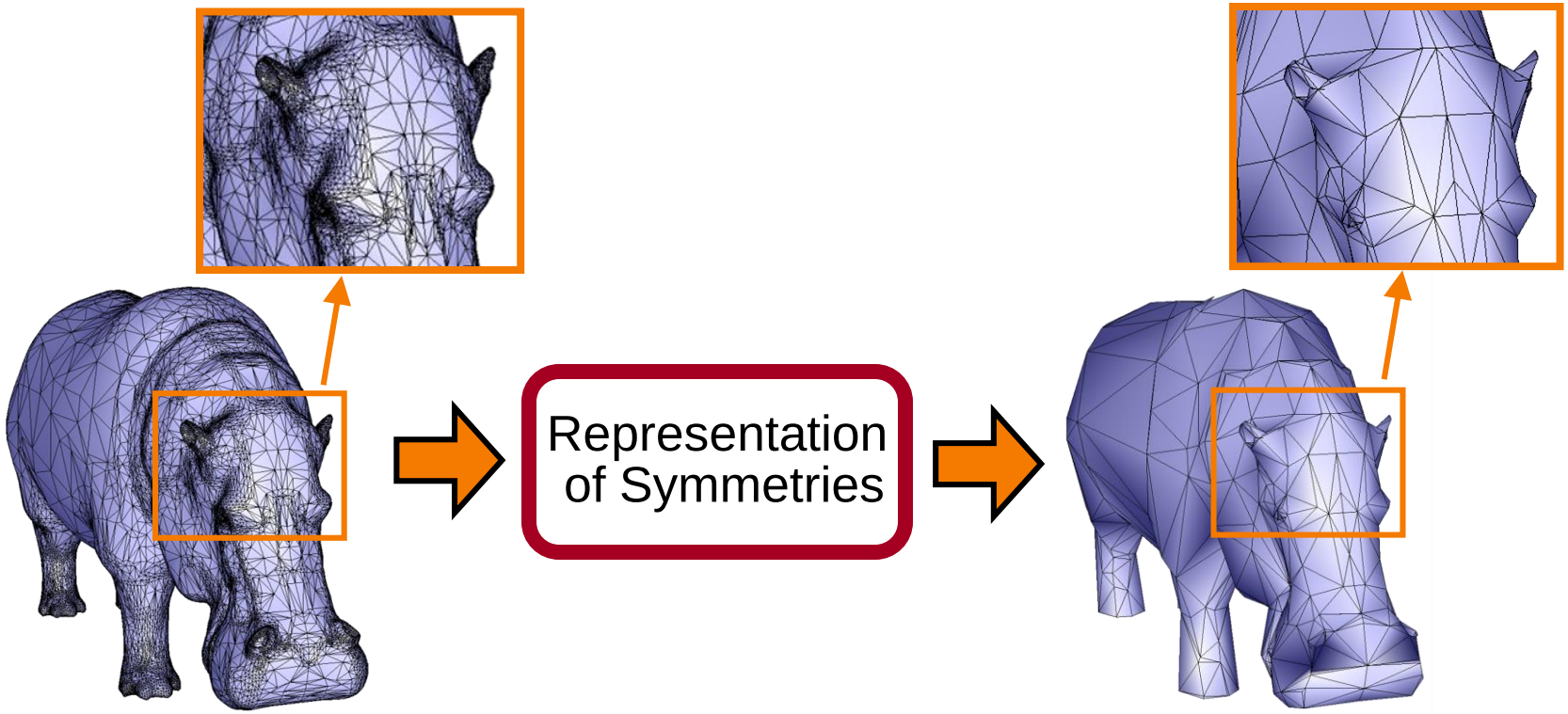
Can use this method to compute symmetry distance for any 3D mesh (without finding correspondences)



Sym(m,p) = 0.3

Back to the Goal …

We need methods to detect, analyze, **represent**, and exploit symmetries in 3D models



Symmetric Input Mesh

Symmetric Output Mesh

Outline

Introduction

Background

Symmetry representations

- Symmetry descriptor
- Symmetry transform
- Principal symmetries

Applications

Conclusion

Outline

Introduction

Background

Symmetry representations

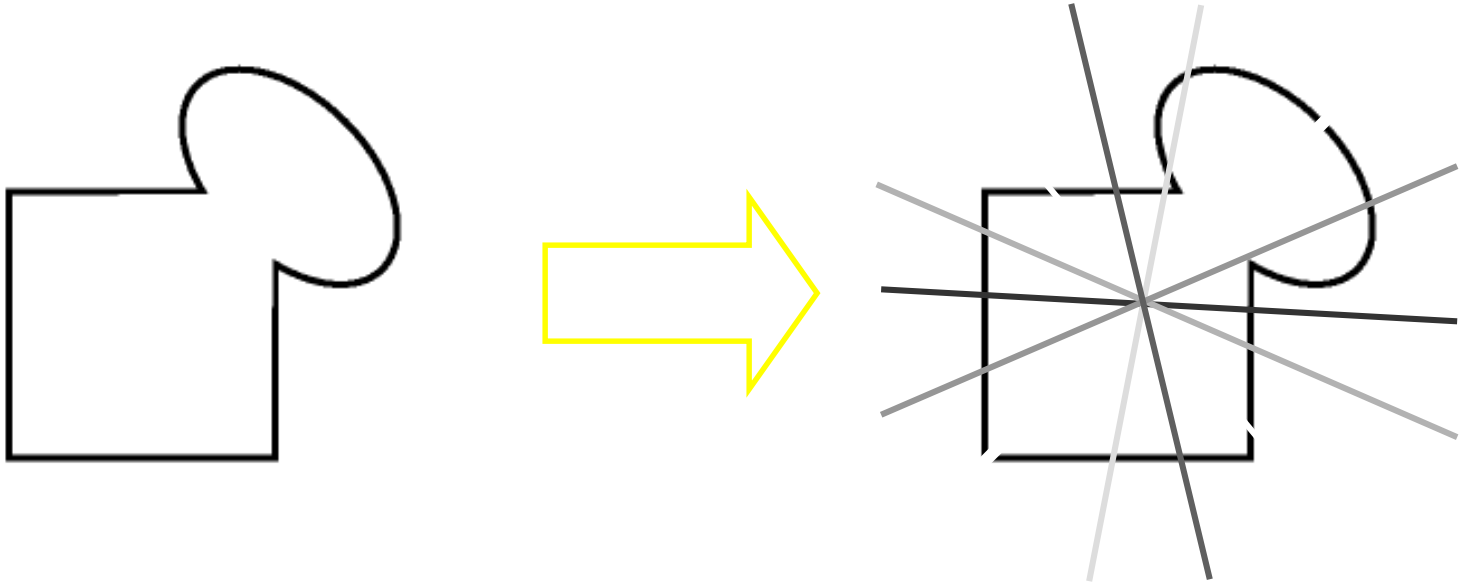
- Symmetry descriptor
- Symmetry transform
- Principal symmetries

Applications

Conclusion

Symmetry Descriptor

Measure the symmetry of an object with respect to every transformation through its center



Input Shape

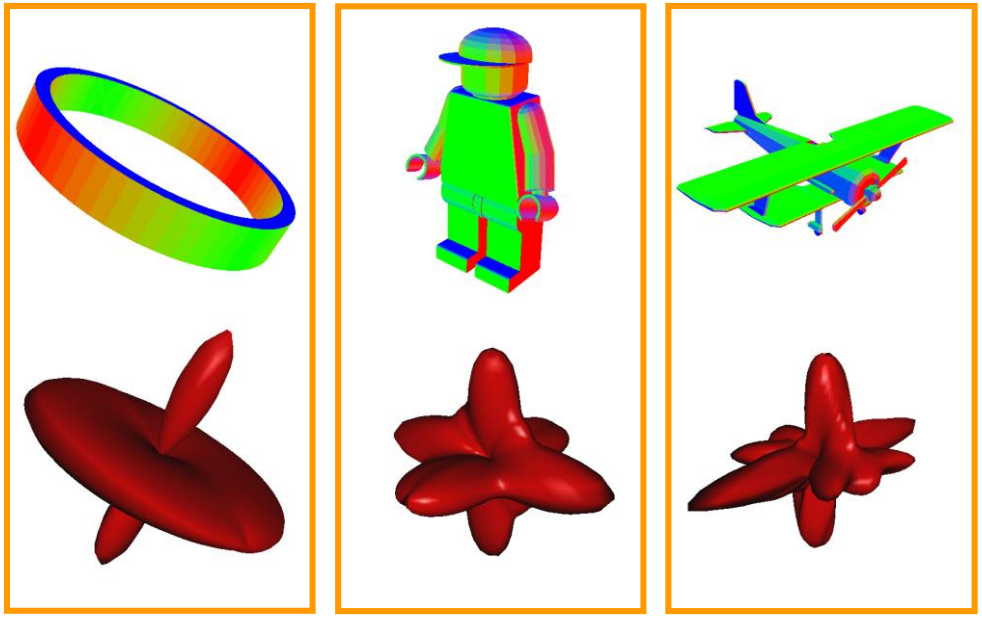
Planar Reflective Symmetry Descriptor

Symmetry Descriptor

Planar reflective symmetry descriptors:

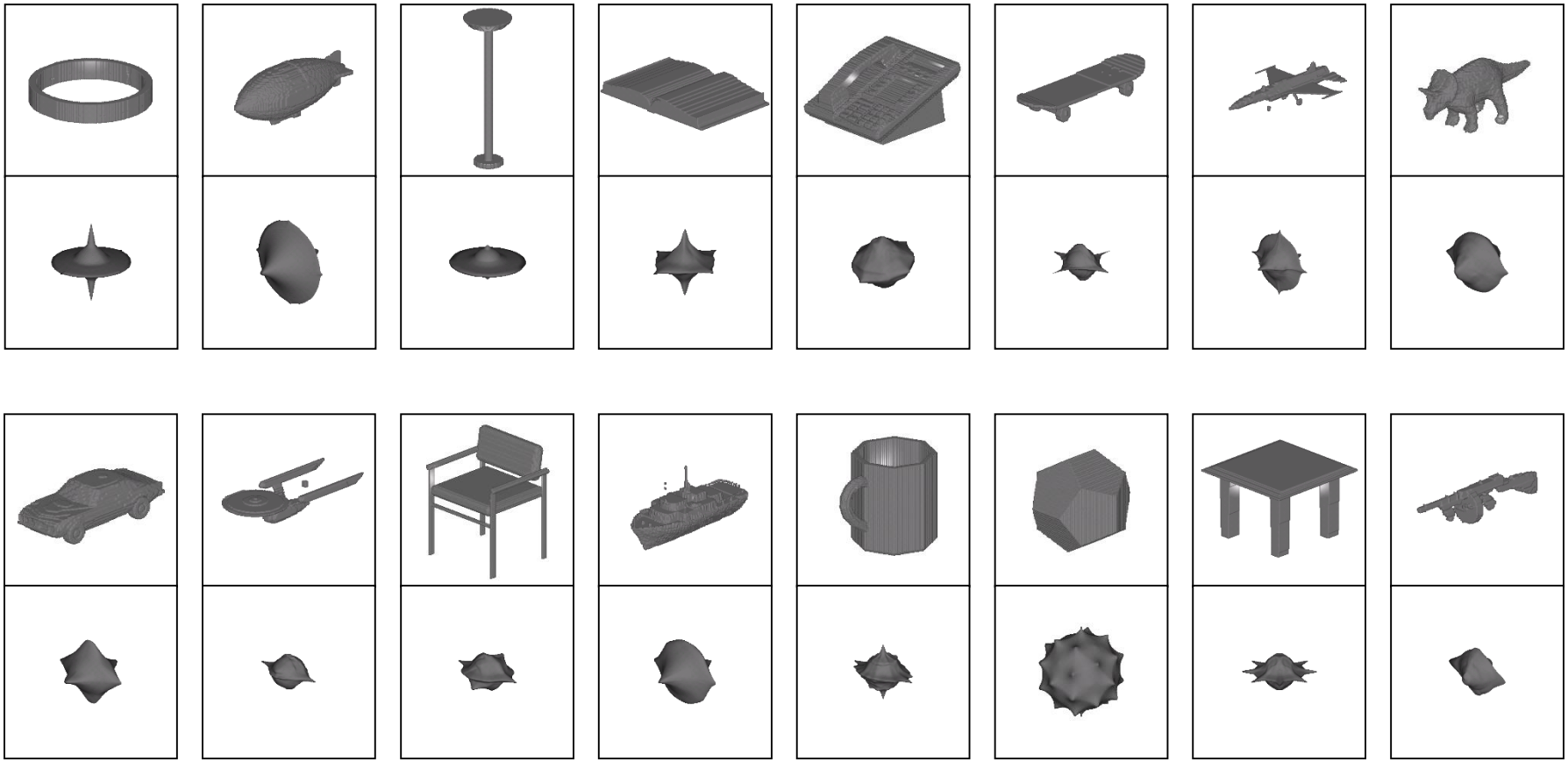
Input Mesh

Planar Reflective Symmetry Descriptor



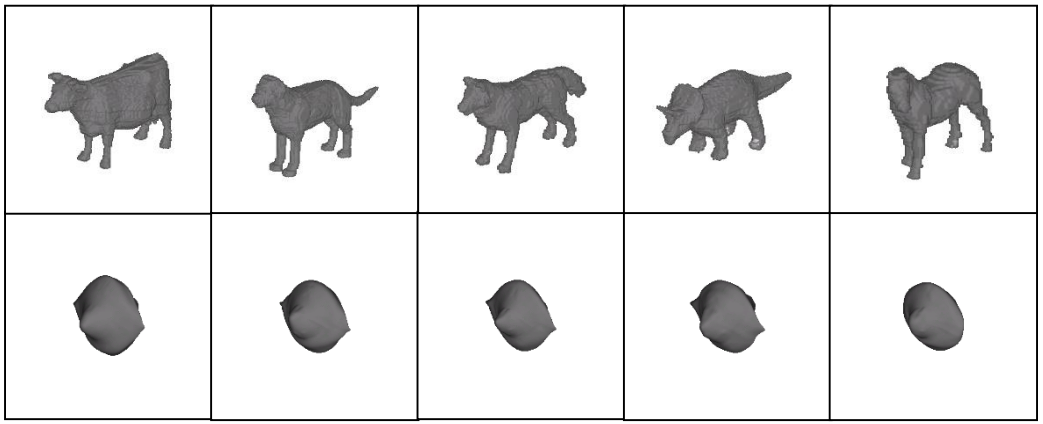
Symmetry Descriptor

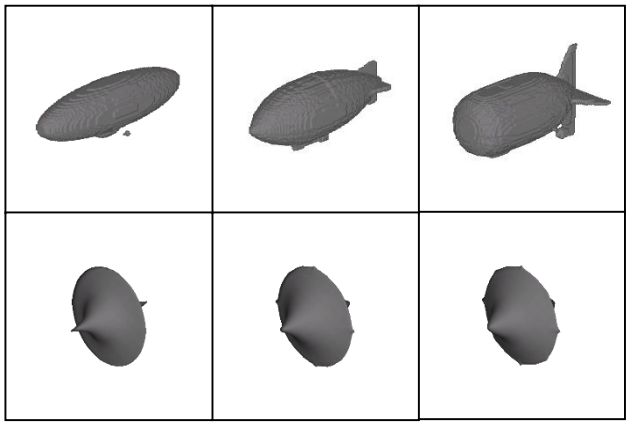
Planar reflective symmetry descriptors:

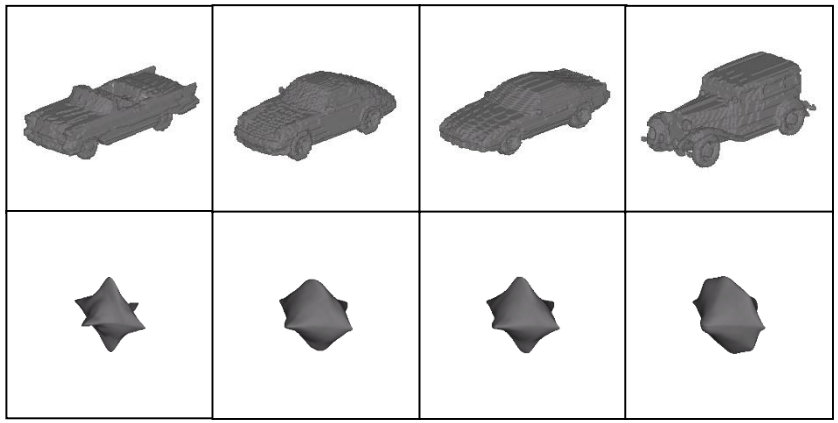


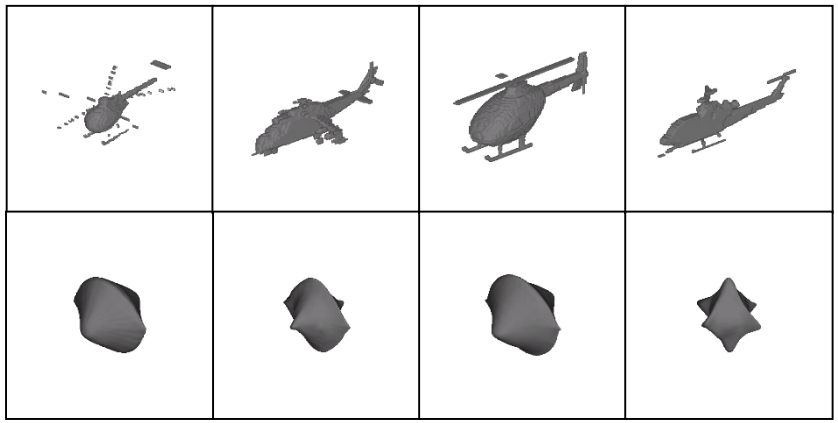
Symmetry Descriptor

Planar reflective symmetry descriptors:



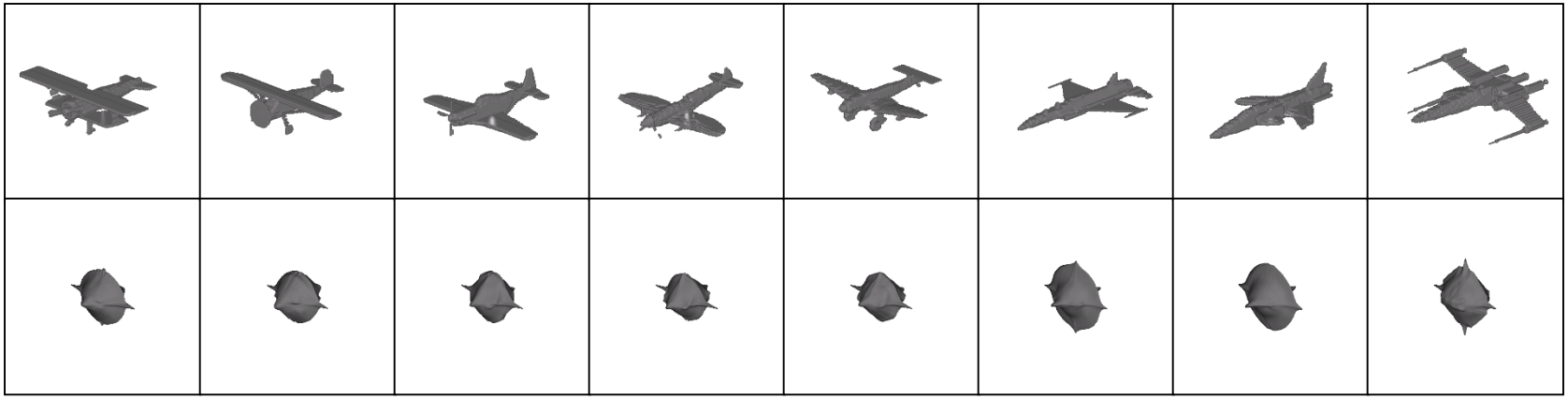


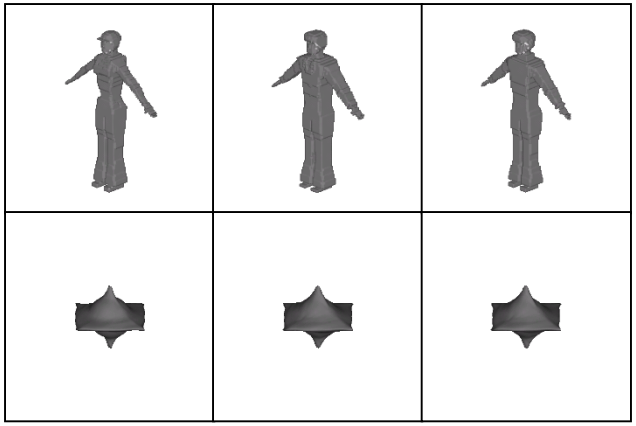


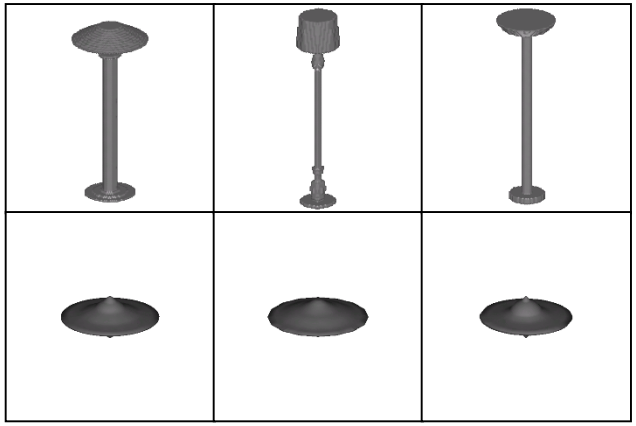


Symmetry Descriptor

Planar reflective symmetry descriptors:

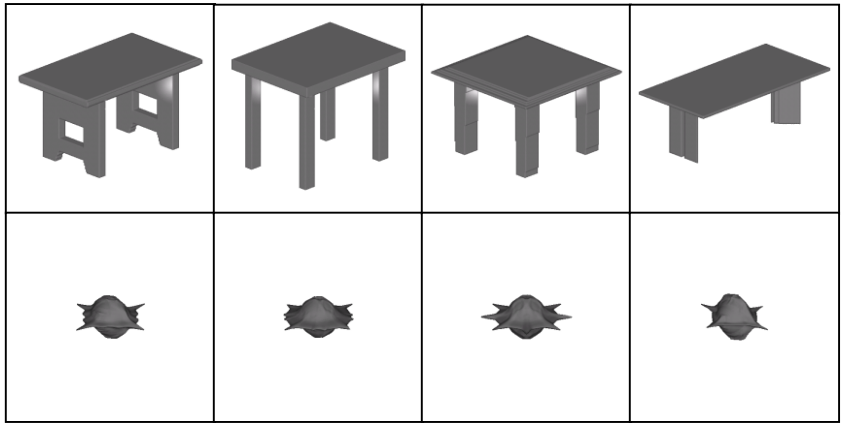


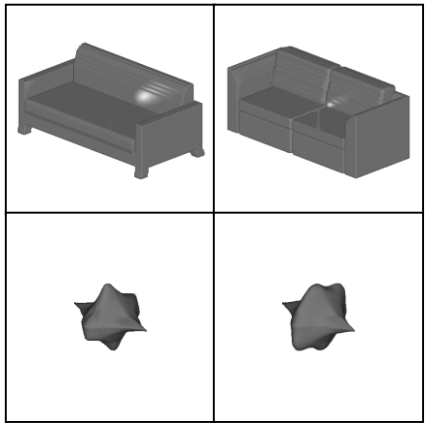




Symmetry Descriptor

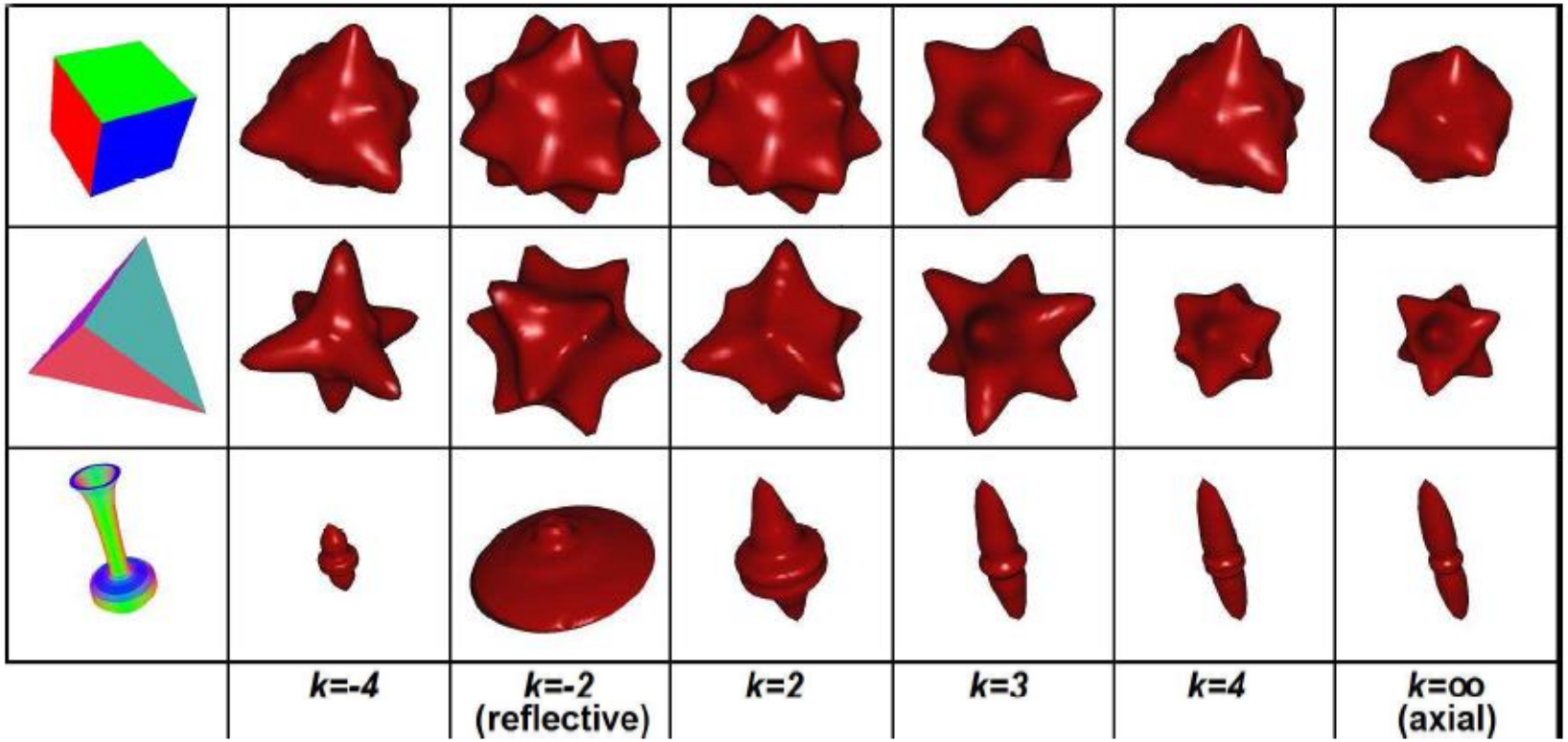
Planar reflective symmetry descriptors:





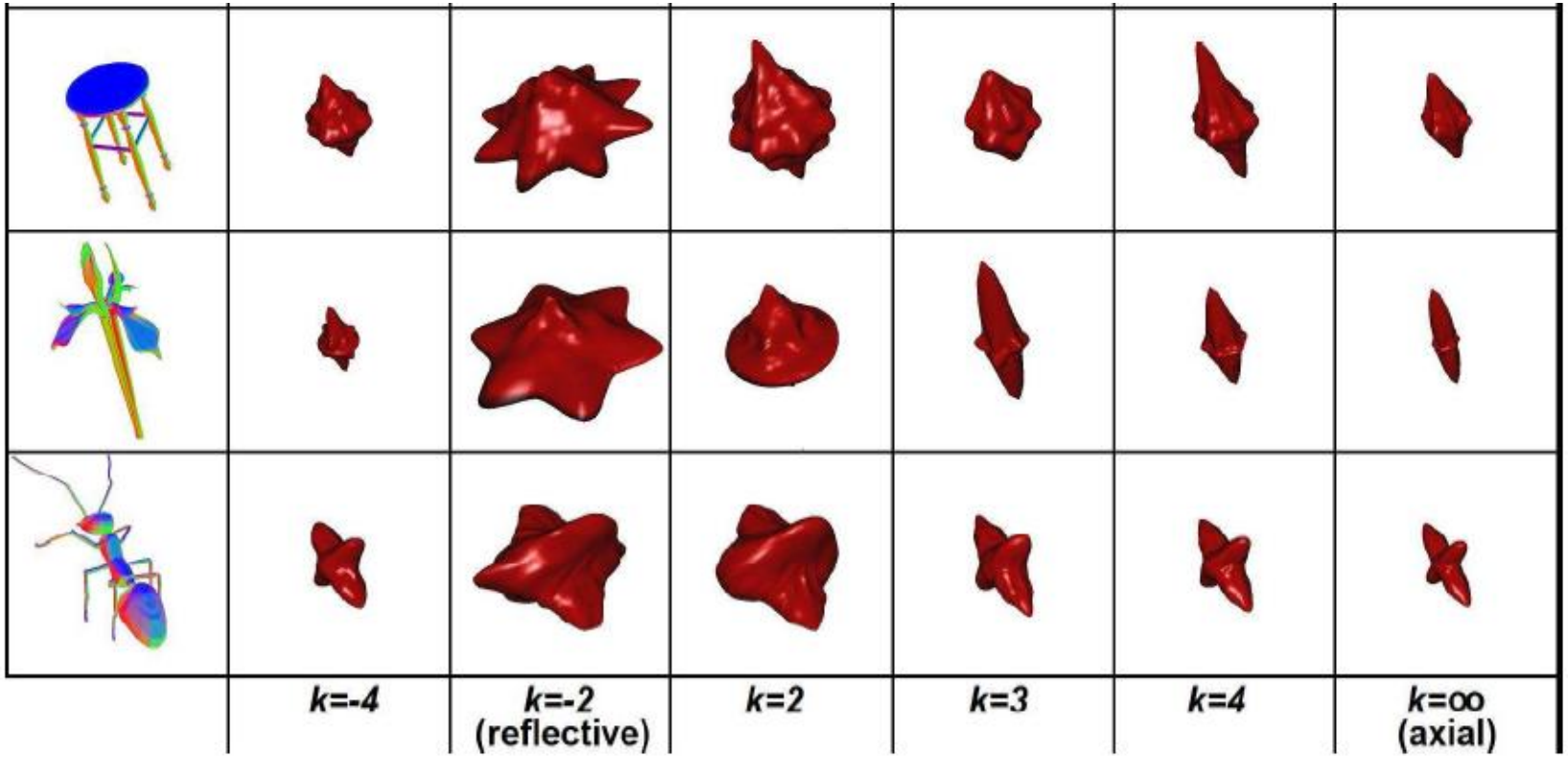
Symmetry Descriptor

Rotational symmetry descriptors:



Symmetry Descriptor

Rotational symmetry descriptors:



Symmetry Descriptor

Properties:

- Canonical parameterization
  - Parameterized over the (projective) sphere
- Insensitive to noise:
  - Integration scales down high frequency Fourier coefficients
- Global:
  - For functions *f* and *g*, and any reflection *r:*

$$\left| Sym(f,r) - Sym(g,r) \right| \leq \left\| f - g \right\|_2$$

Outline

Introduction

Background

Symmetry representations

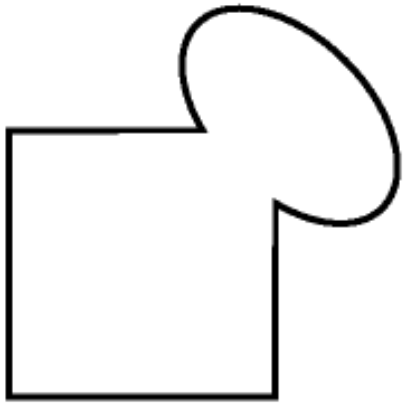
- Symmetry descriptor
- Symmetry transform ←
- Principal symmetries

Applications

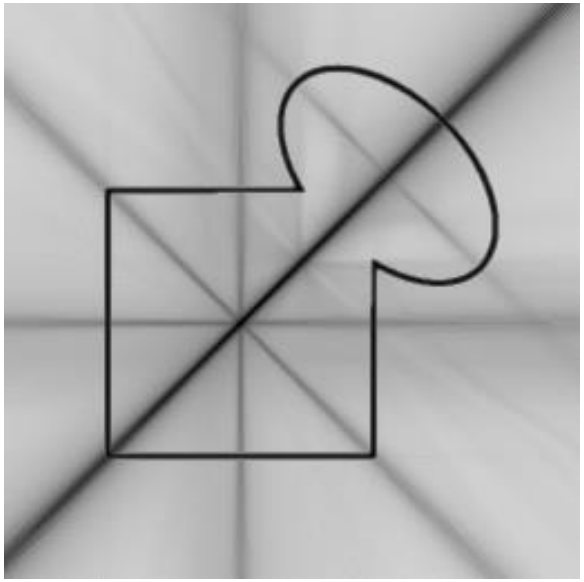
Conclusion

Symmetry Transform

Measure the symmetry of an object with respect to all transformations



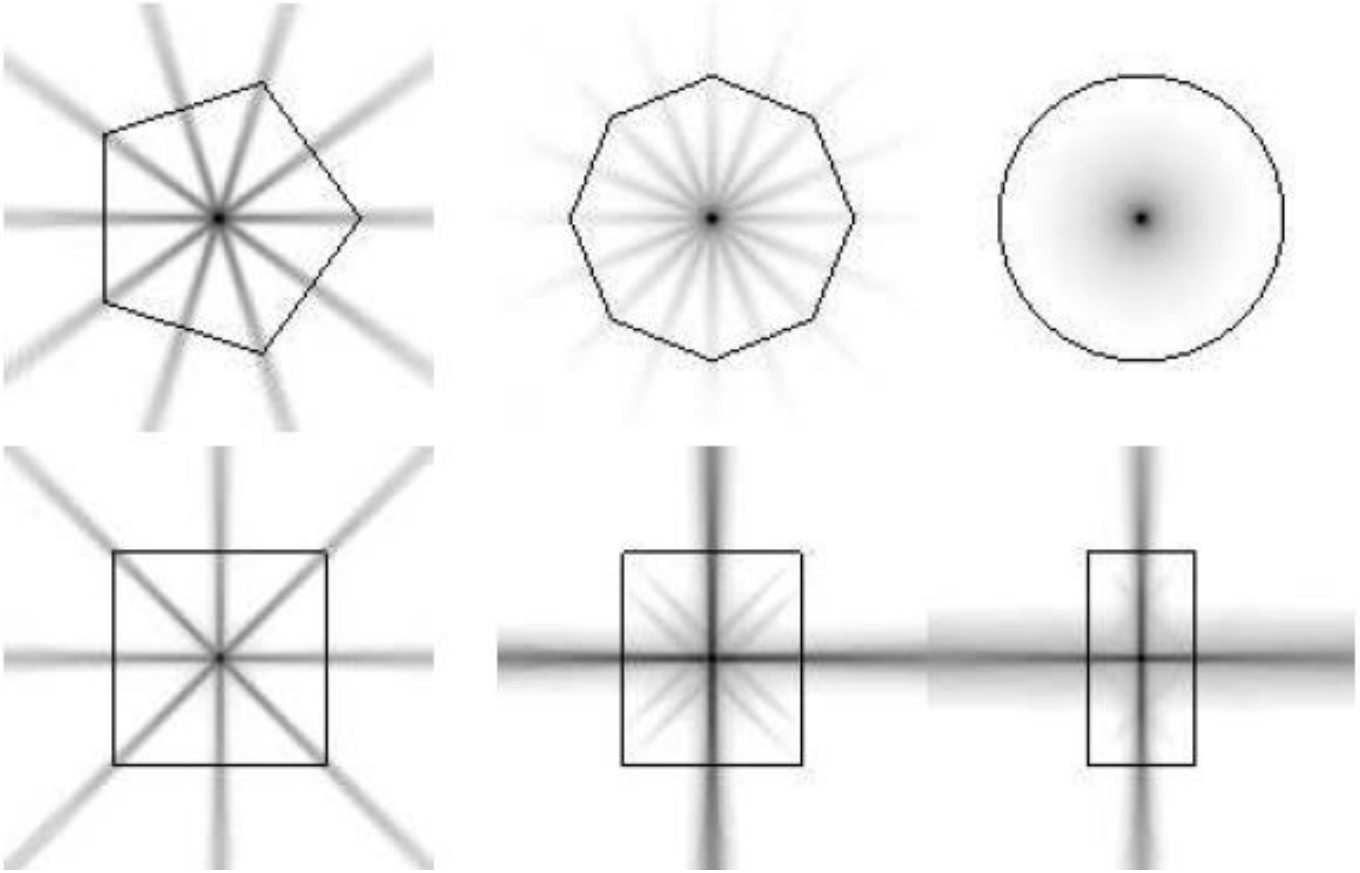
Input Shape



Planar Reflective Symmetry Transform

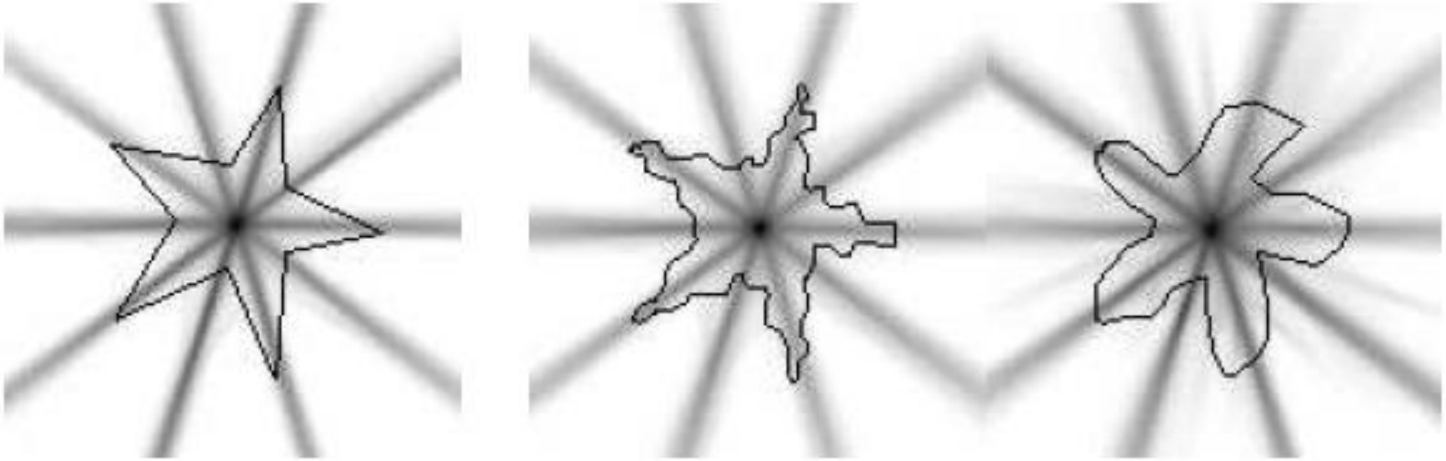
Symmetry Transform

Planar reflective symmetry transform:



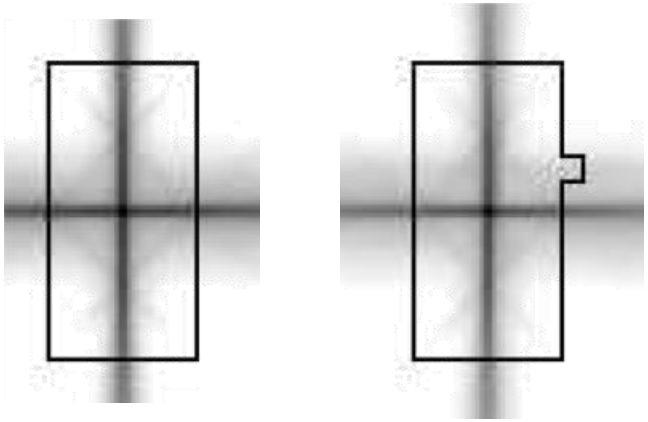
Symmetry Transform

Stability with noise:

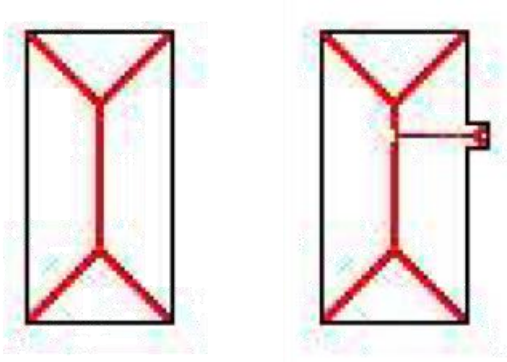


Symmetry Transform

Stability with small extra features:



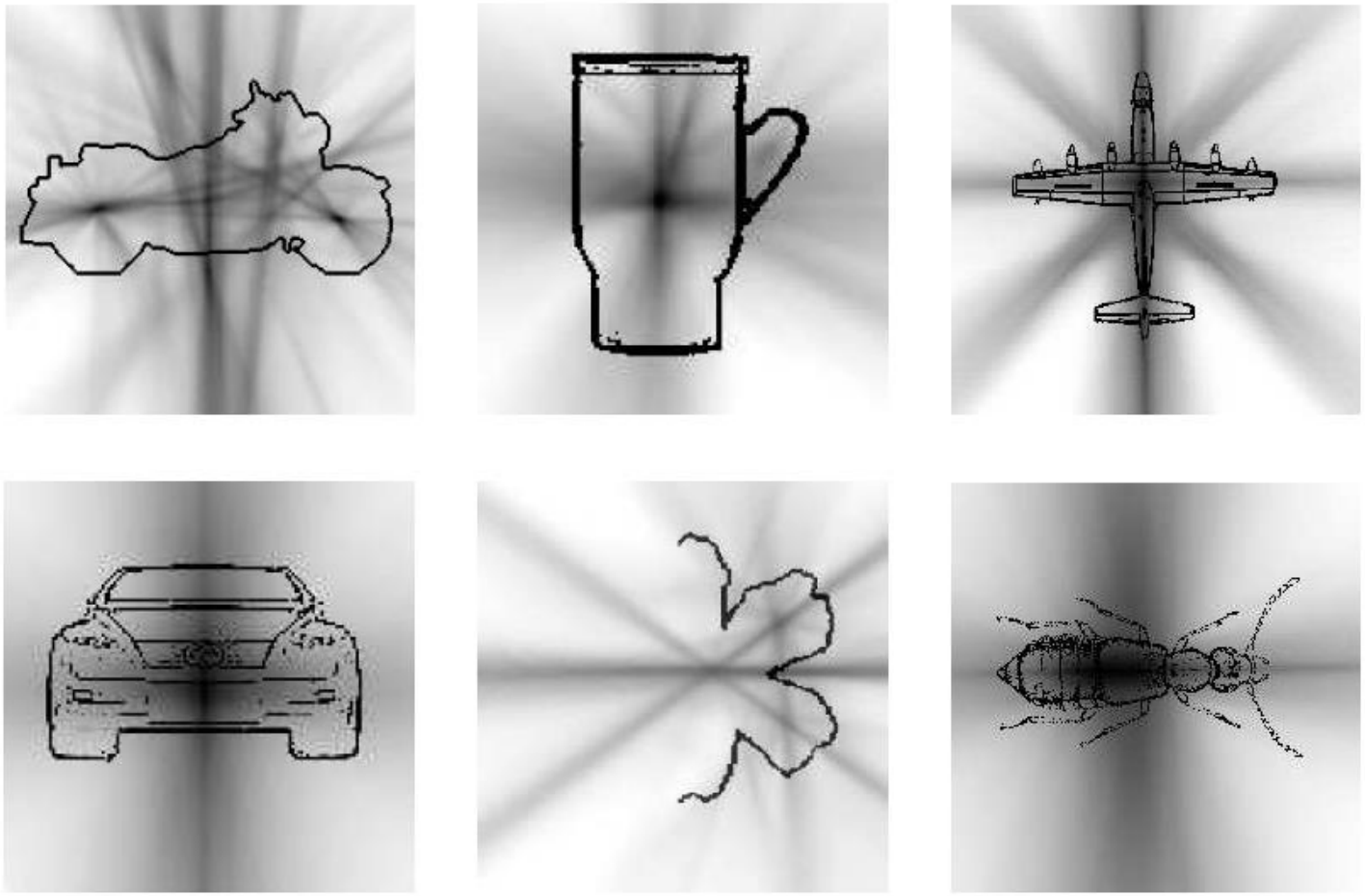
Planar Reflective Symmetry Transform



Medial Axis Transform

Symmetry Transform

Highlights large symmetric features of shape:



Symmetry Transform

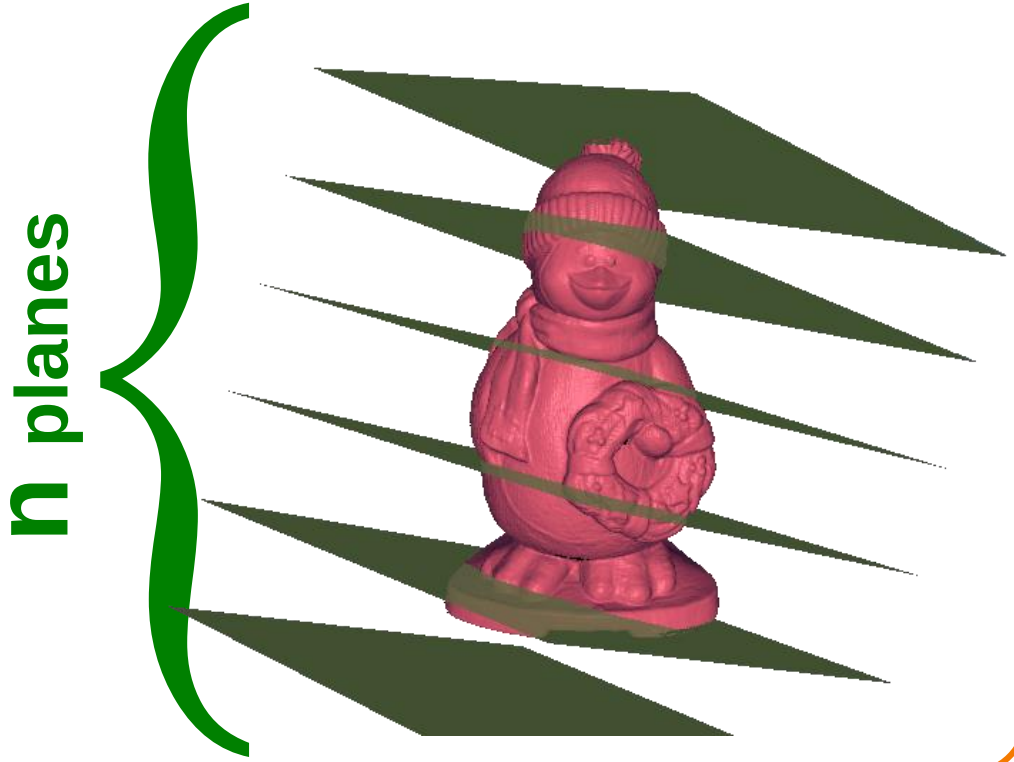
Computation:

- Brute Force: $O(n^6)$

$O(n^3)$ planes

X = $O(n^6)$

$O(n^3)$ correlation



Symmetry Transform

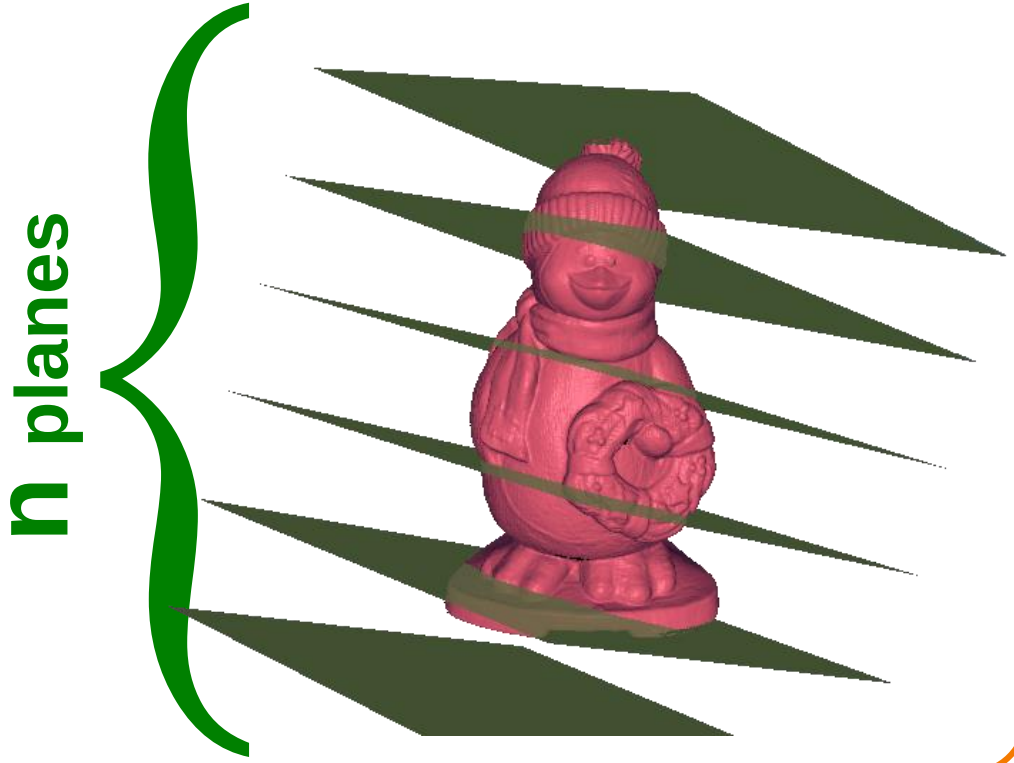
Computation:

- Brute Force: $O(n^6)$
- ➢ Convolution: $O(n^5 \log n)$

$O(n^2)$ normal directions

X = $O(n^5 \log n)$

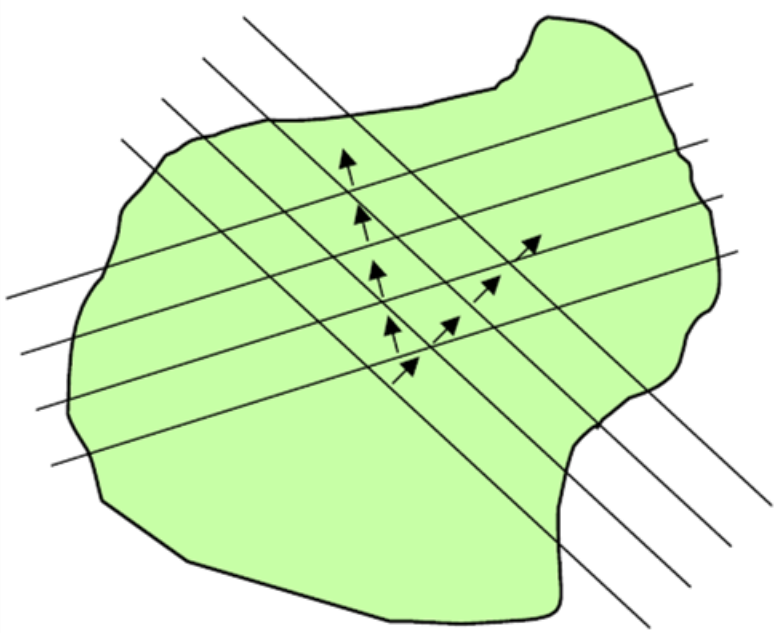
$O(n^3 \log n)$ per direction



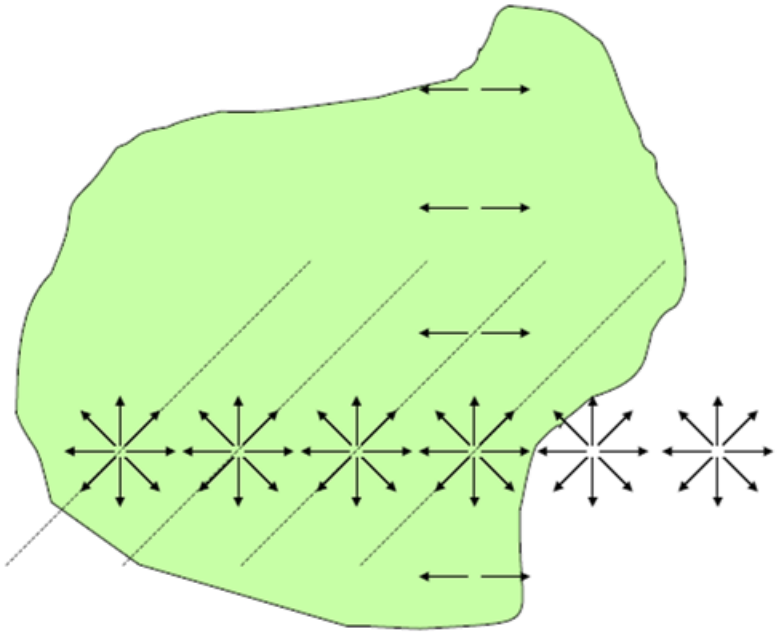
Symmetry Transform

Computation:

- Brute Force: $O(n^6)$
- ➢ Convolution: $O(n^5 \log n)$



Translate planes



Rotate planes

Symmetry Transform

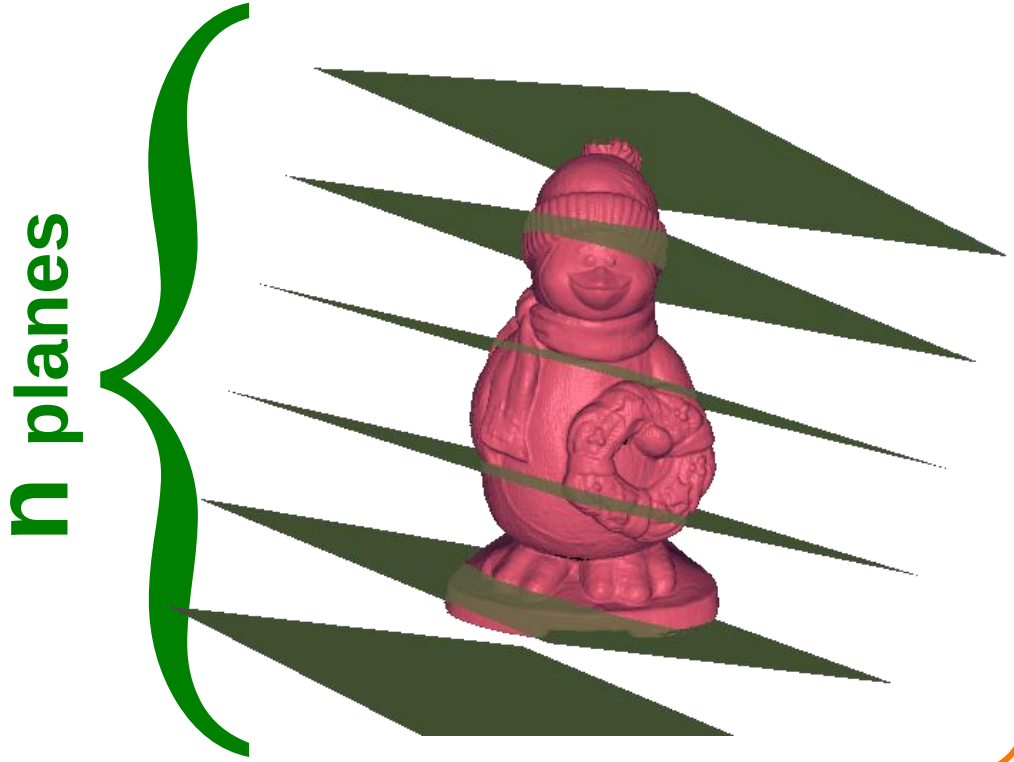
Computation:

- Brute Force: $O(n^6)$
- Convolution: $O(n^5 \log n)$
- ➢ Monte Carlo: $O(n^4)$ for surfaces

$O(n^2)$ surface points

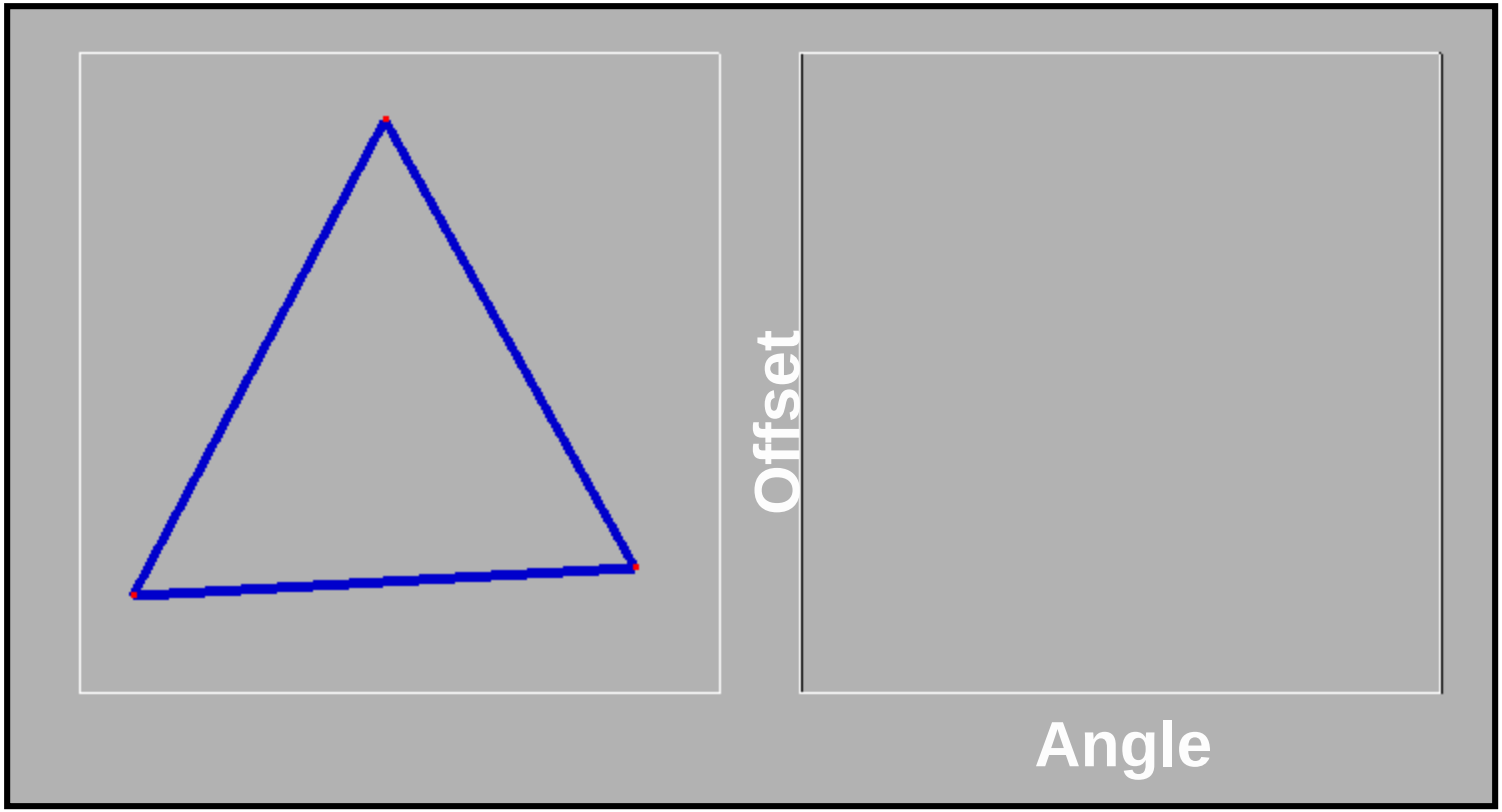
X = $O(n^4)$

$O(n^2)$ pairs



Symmetry Transform

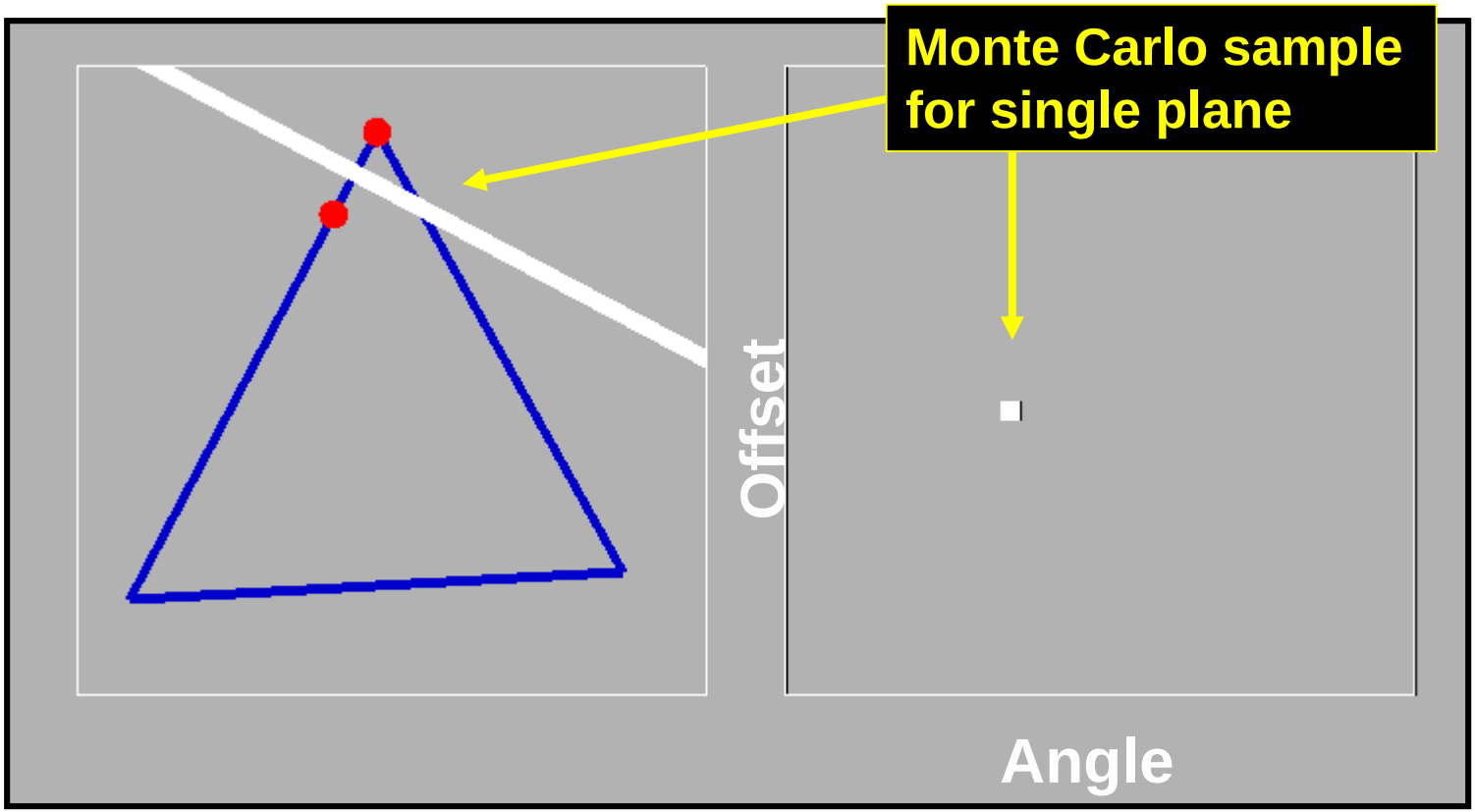
Monte Carlo algorithm:



Input Model **Symmetry Transform**

Symmetry Transform

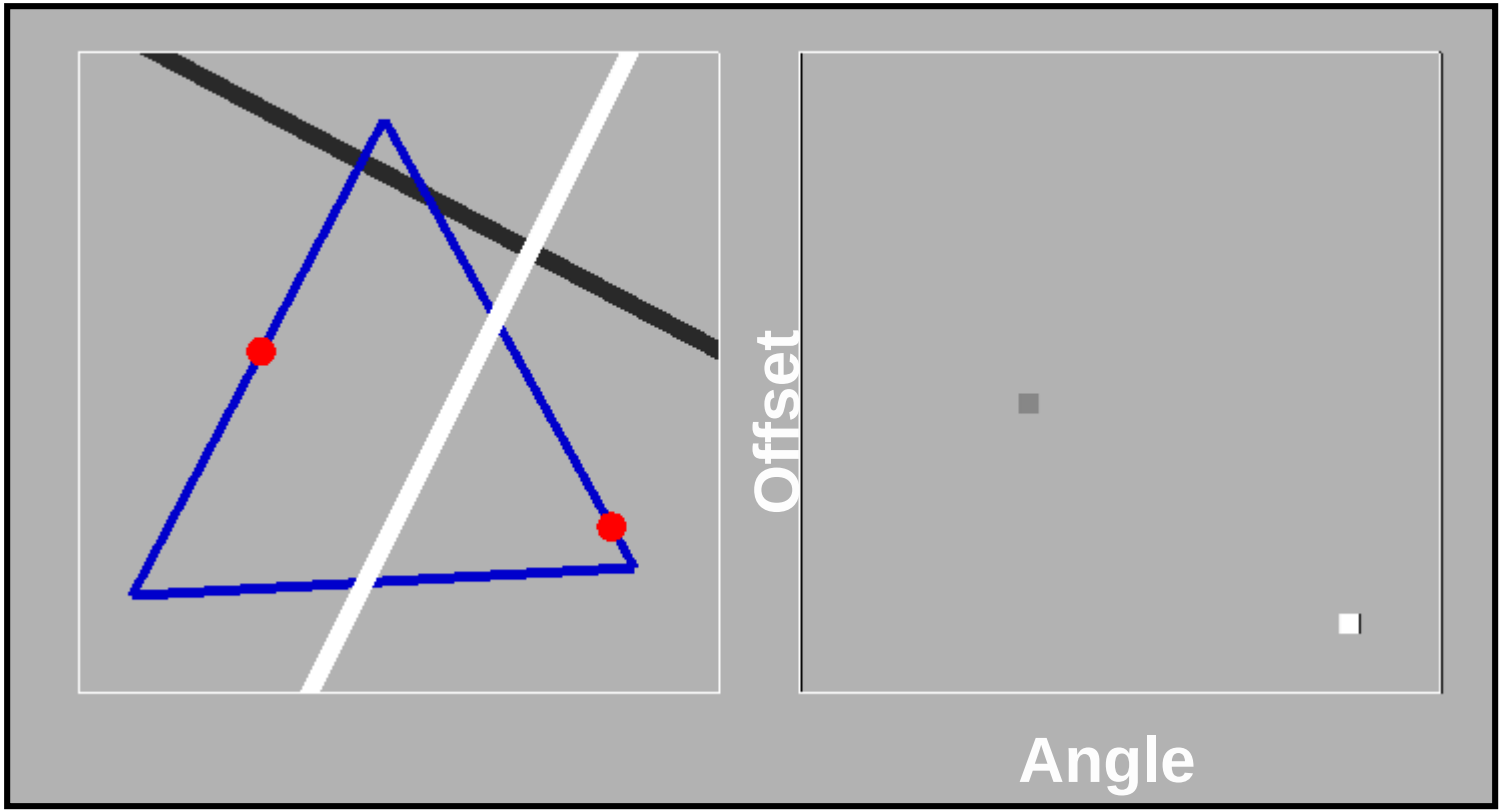
Monte Carlo algorithm:



Input Model **Symmetry Transform**

Symmetry Transform

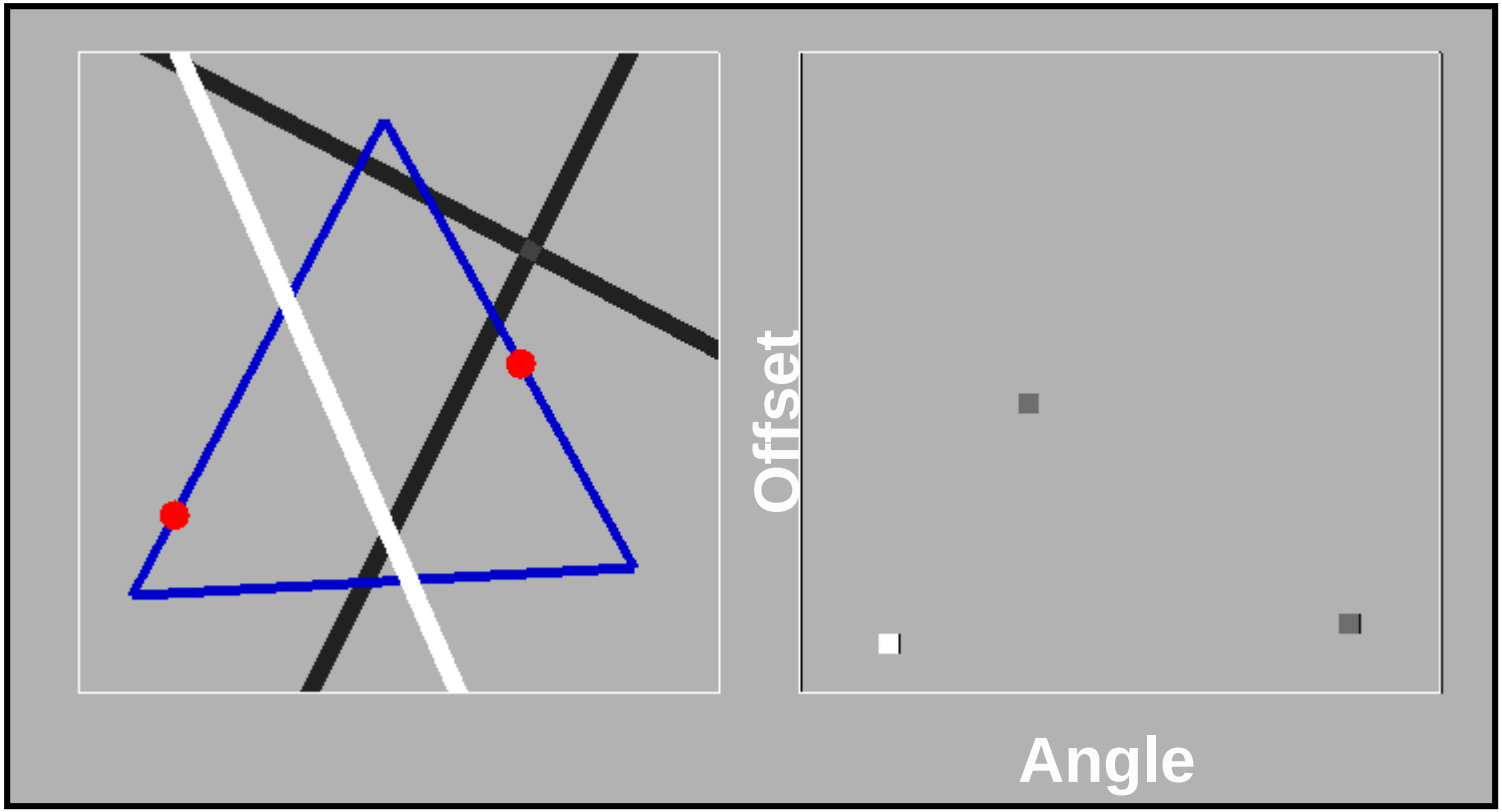
Monte Carlo algorithm:



Input Model **Symmetry Transform**

Symmetry Transform

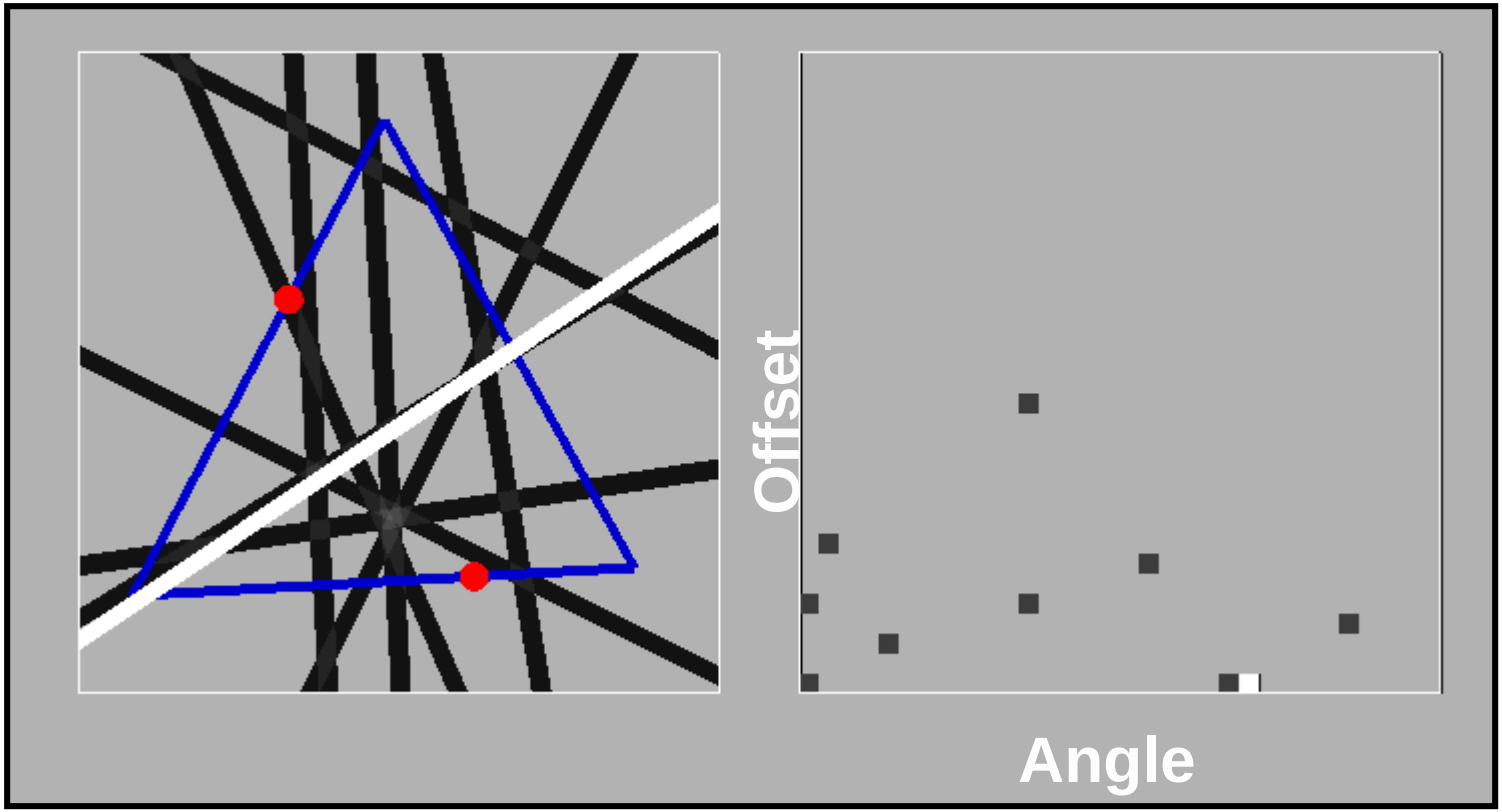
Monte Carlo algorithm:



Input Model **Symmetry Transform**

Symmetry Transform

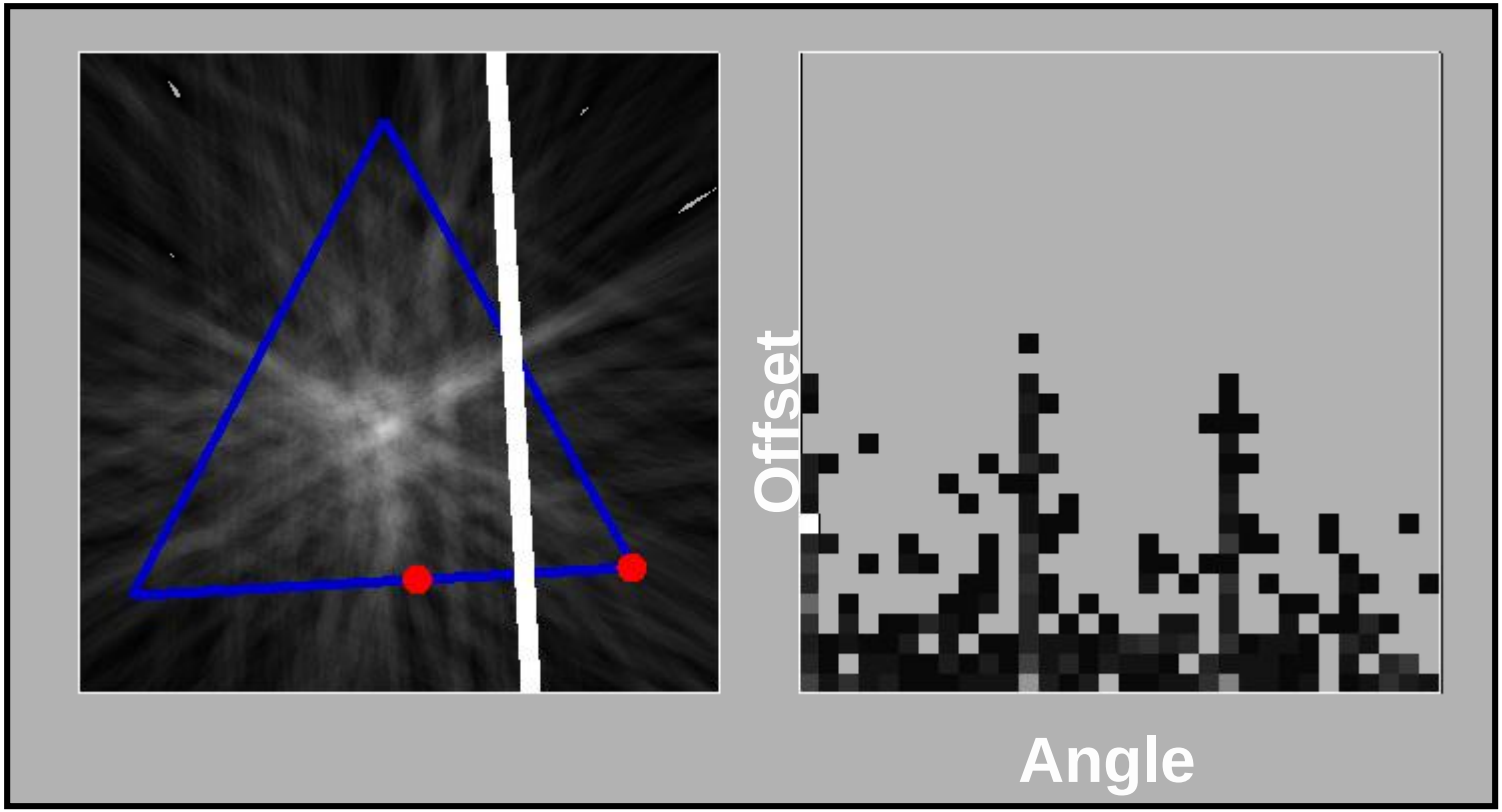
Monte Carlo algorithm:



Input Model **Symmetry Transform**

Symmetry Transform

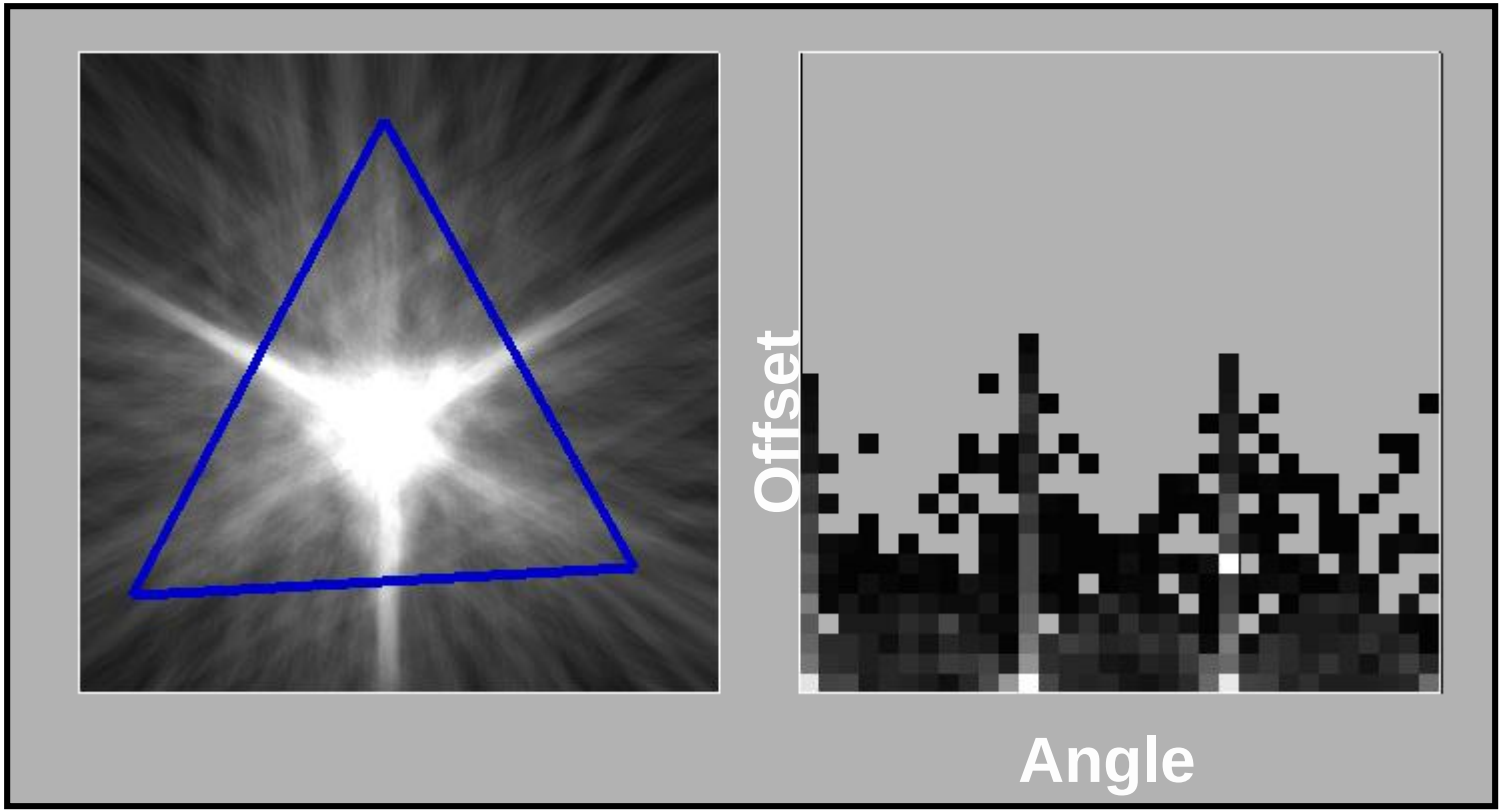
Monte Carlo algorithm:



Input Model **Symmetry Transform**

Symmetry Transform

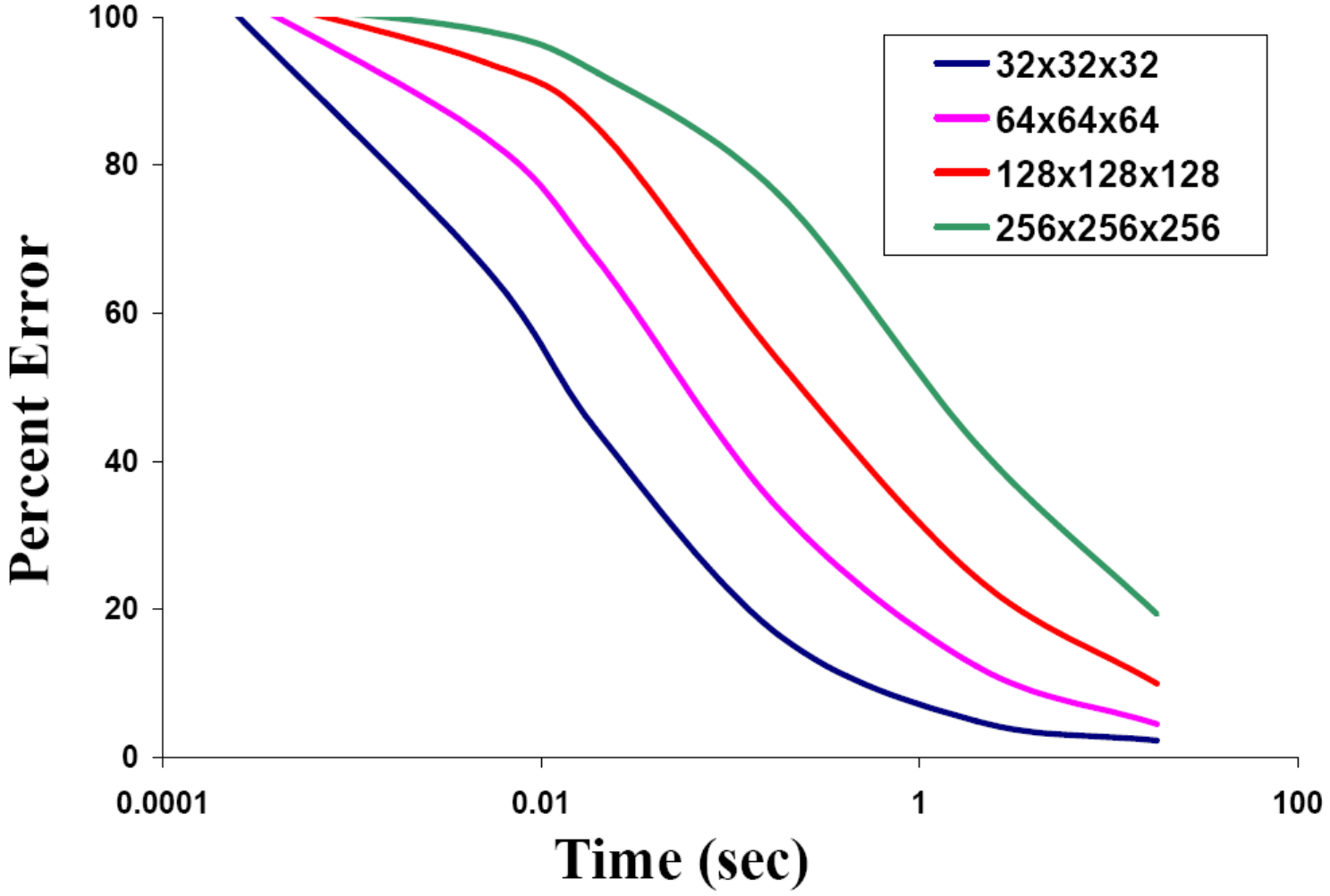
Monte Carlo algorithm:



Input Model **Symmetry Transform**

Symmetry Transform

Compute time:



Outline

Introduction

Background

Symmetry representations

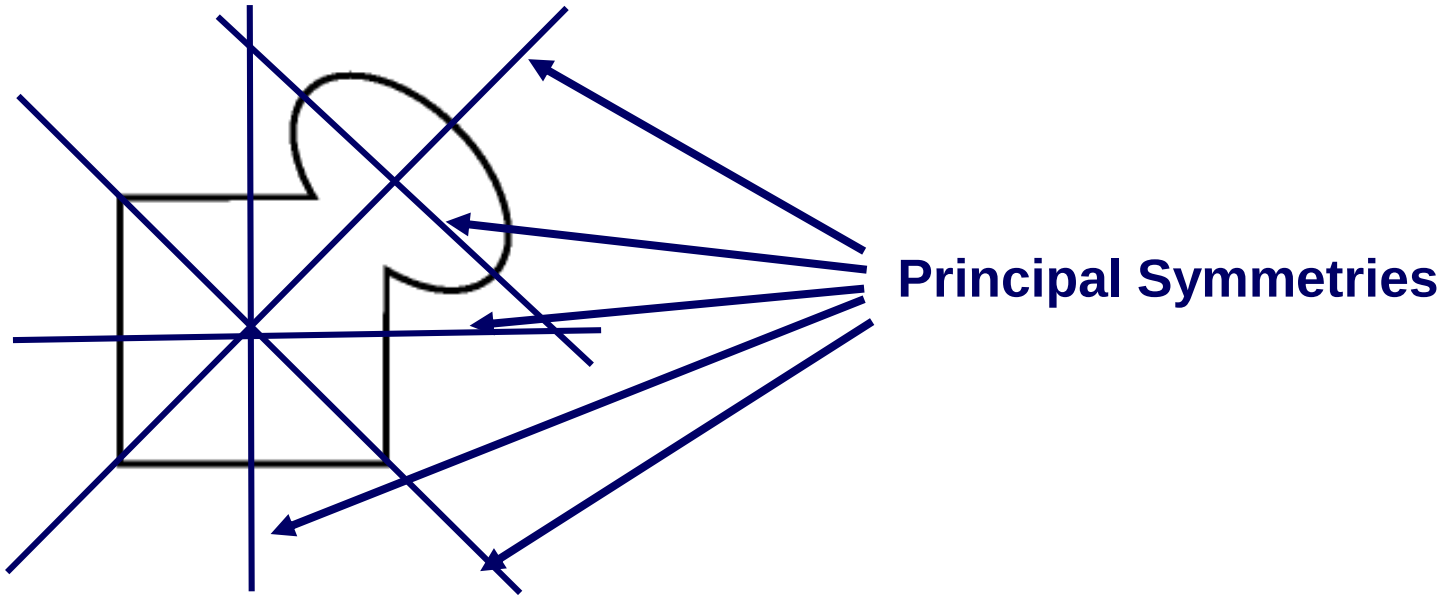
- Symmetry descriptor
- Symmetry transform
- Principal symmetries

Applications

Conclusion

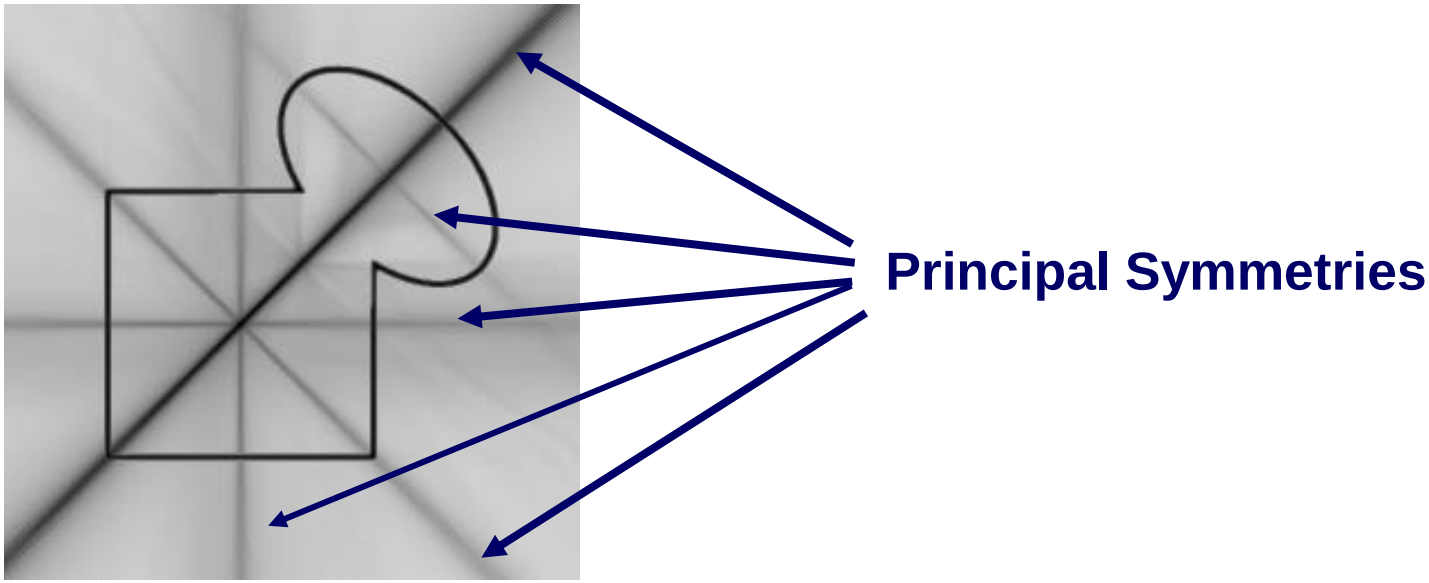
Principal Symmetries

Motivation: finding and representing significant symmetries is sufficient for many applications



Principal Symmetries

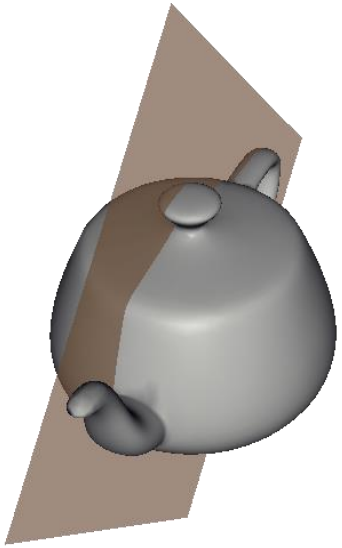
Observation: significant symmetries are usually local maxima of symmetry transformation



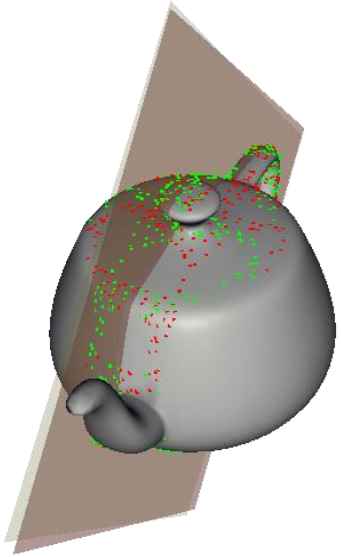
Principal Symmetries

Computing local maxima precisely:

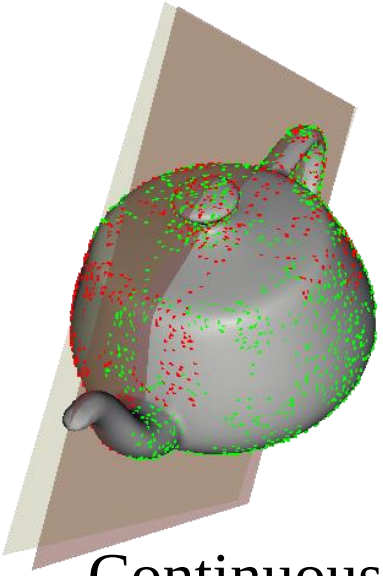
- Start from local maxima of discrete transform
- Establish closest point correspondences
- Refine transformation
- Iterate until find local maxima



Discrete
Local Maximum



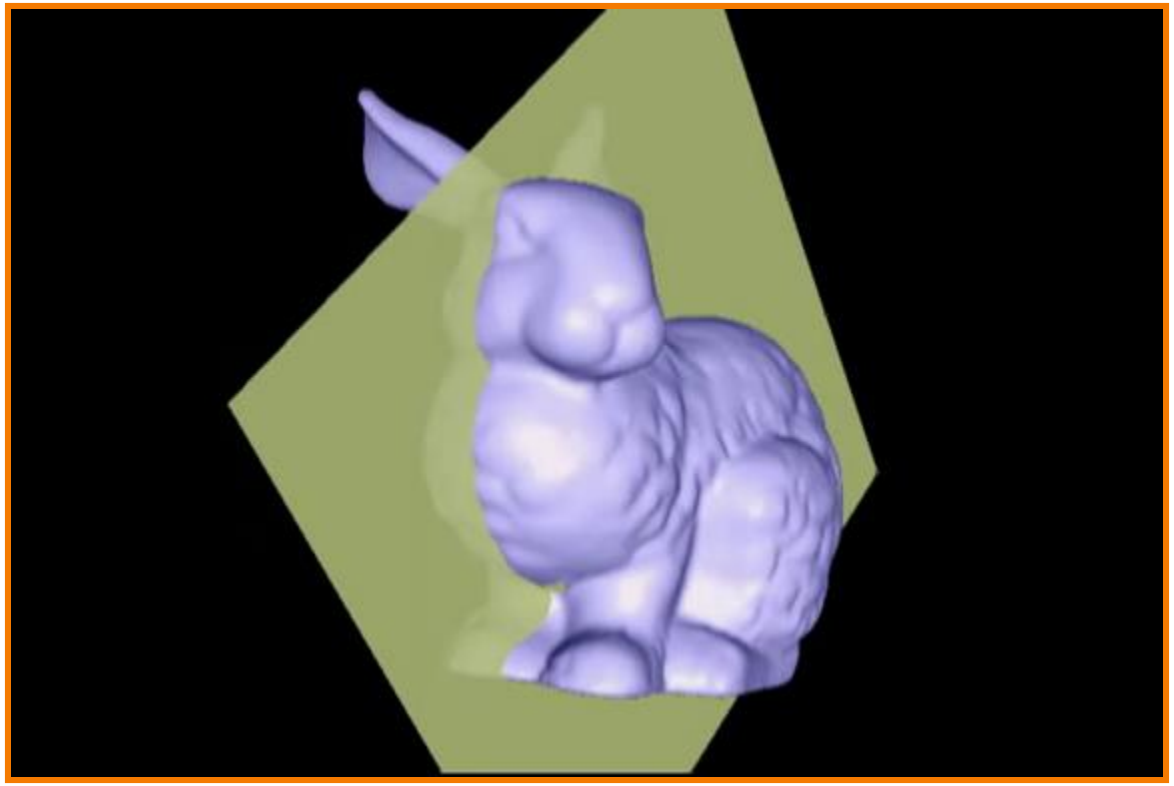
Closest Point
Correspondences



Continuous
Local Maximum

Principal Symmetries

Computing local maxima precisely:



Outline

Introduction

Background

Symmetry representations

- Symmetry descriptor
- Symmetry transform
- Principal symmetries

Applications ←

Conclusion

Related Work

Exploiting symmetry in geometric processing

- Completion – Zabrodsky93, Thrun et al. 2005
- Alignment – Zabrodsky et al. 1995, Kazhdan et al. 2002
- Symmetrization - Zabrodsky et al. 1997, Mitra et al. 2007
- Feature detection – Reisfeld et al. 1995
- Reverse engineering - Mills et al. 2001
- Instancing - Martinet et al. 2005
- Matching – Kazhdan et al. 2002, Gal et al. 2005
- Compression - Simari et al. 2006
- Segmentation - Mitra et al. 2006, Podolak et al. 2006
- Viewpoint selection – Podolak et al. 2006
- Editing – Mitra et al. 2006
- Simplification – Podolak et al. 2007

Applications

Alignment

Matching

Segmentation

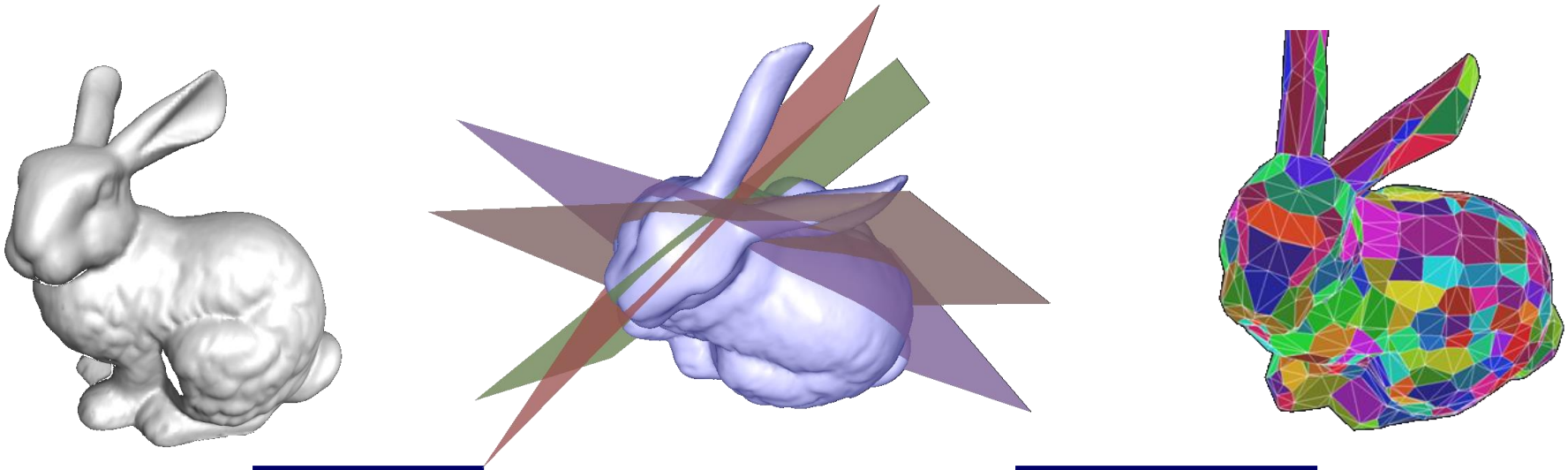
Viewpoint selection

Simplification

Beautification

Texture synthesis

Symmetry-Aware Processing



Input Mesh → Symmetry Analysis → *Principal Symmetry Transformations + Regions of Support* → Symmetry Aware Processing → Output Mesh

Applications

Alignment ←

Matching

Segmentation

Viewpoint selection

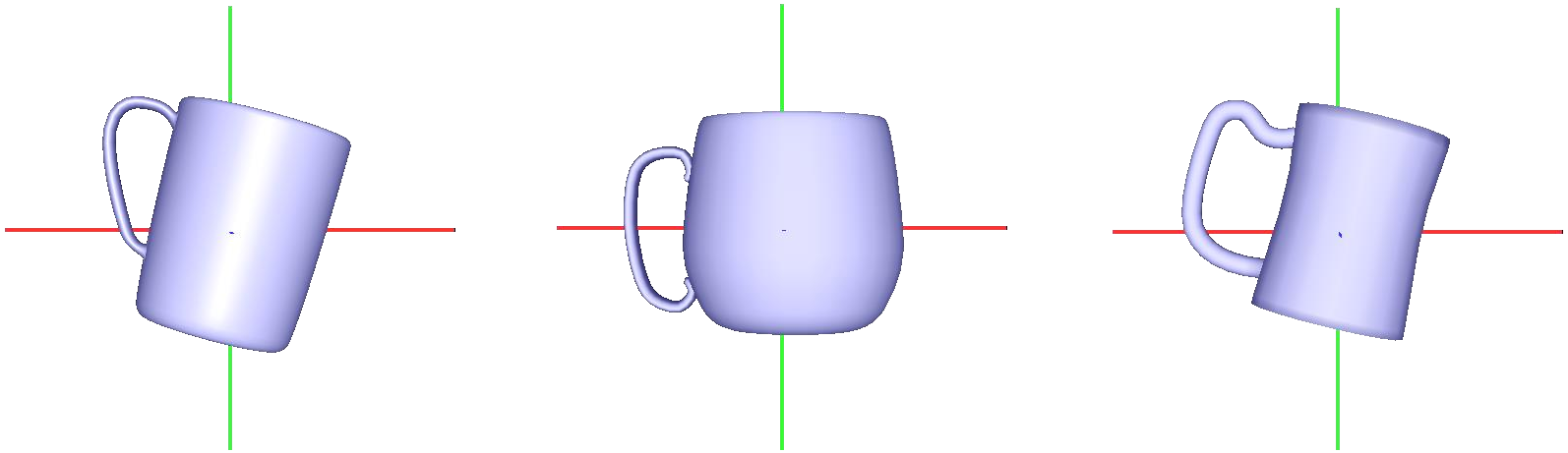
Simplification

Beautification

Texture synthesis

Application: Alignment

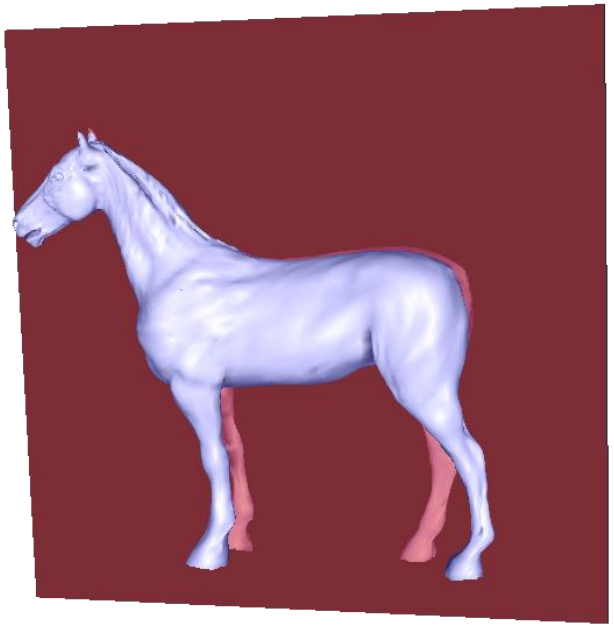
Motivation: registration, modeling, etc.



PCA Alignment

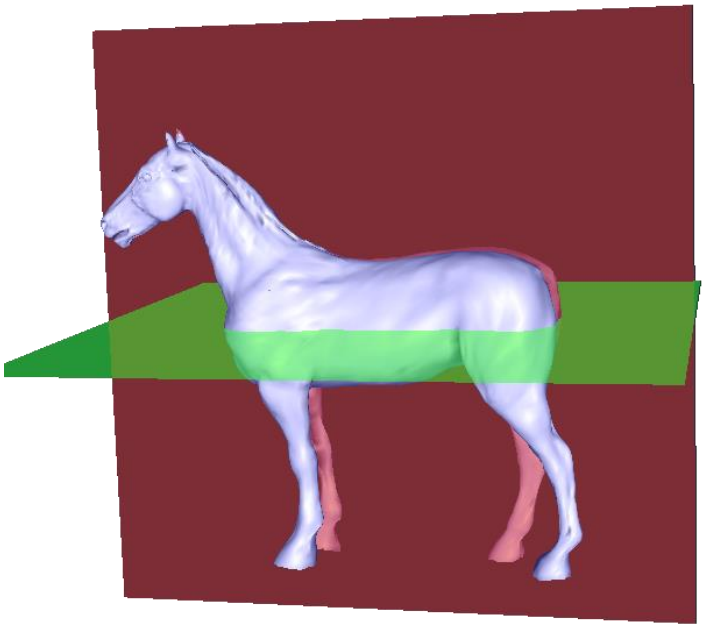
Application: Alignment

Approach: align planes with highest symmetries



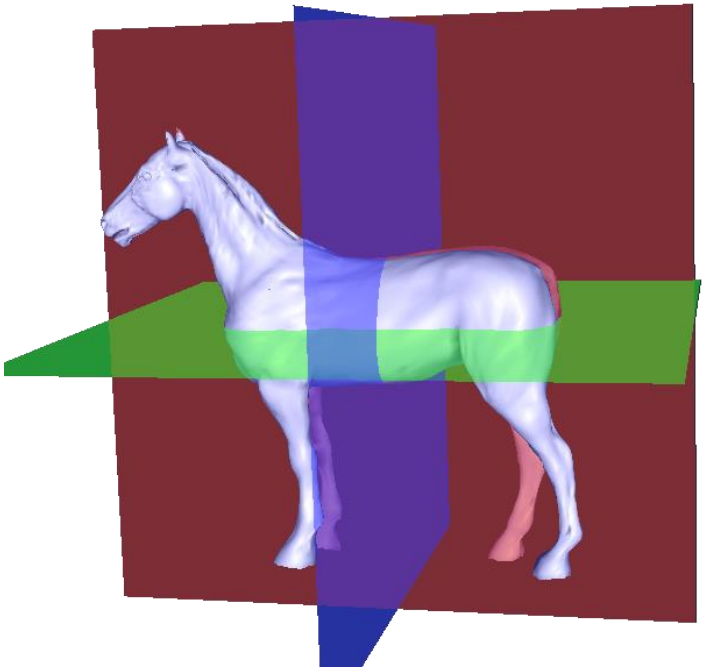
Application: Alignment

Approach: align planes with highest symmetries



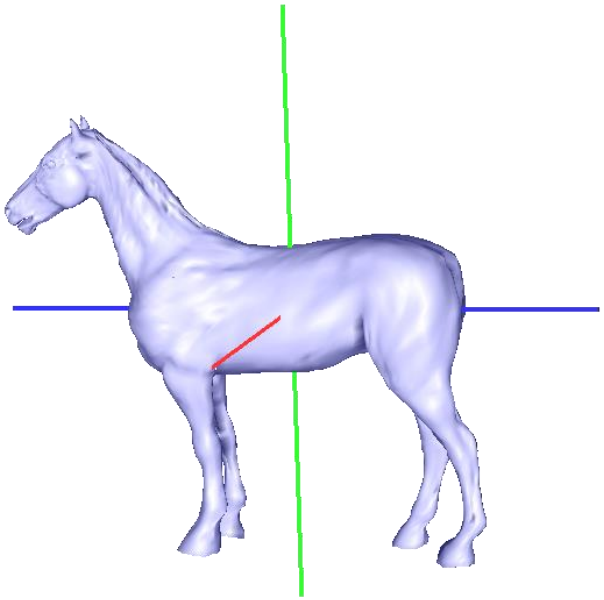
Application: Alignment

Approach: align planes with highest symmetries



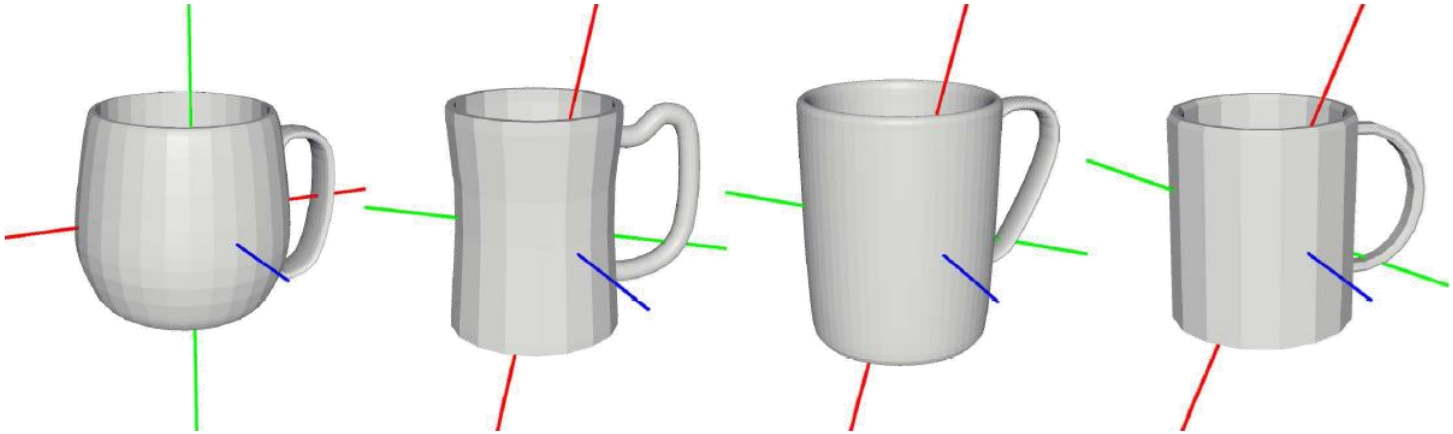
Application: Alignment

Approach: align planes with highest symmetries

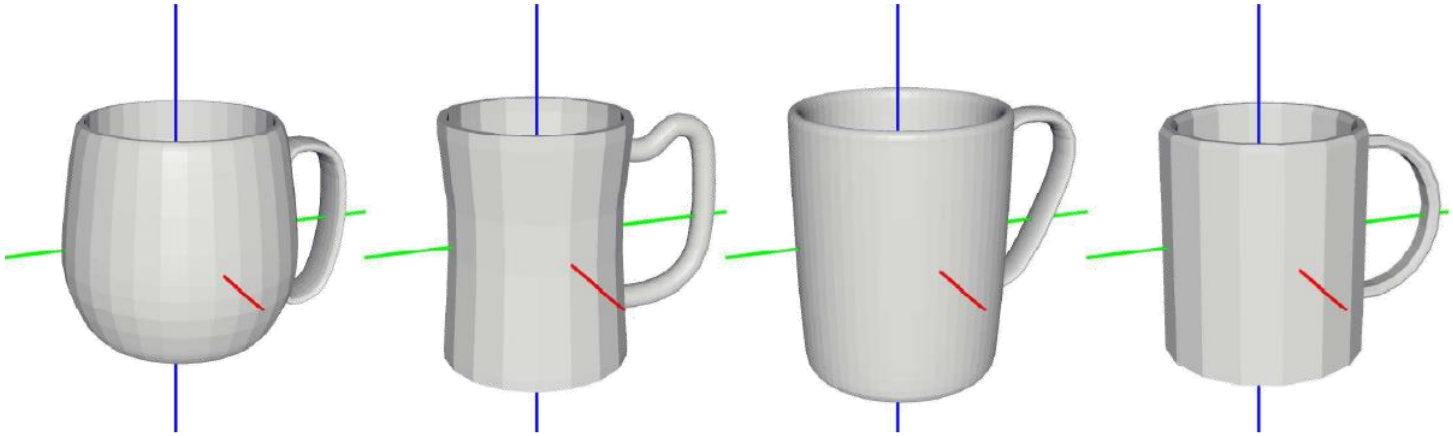


Application: Alignment

Results:



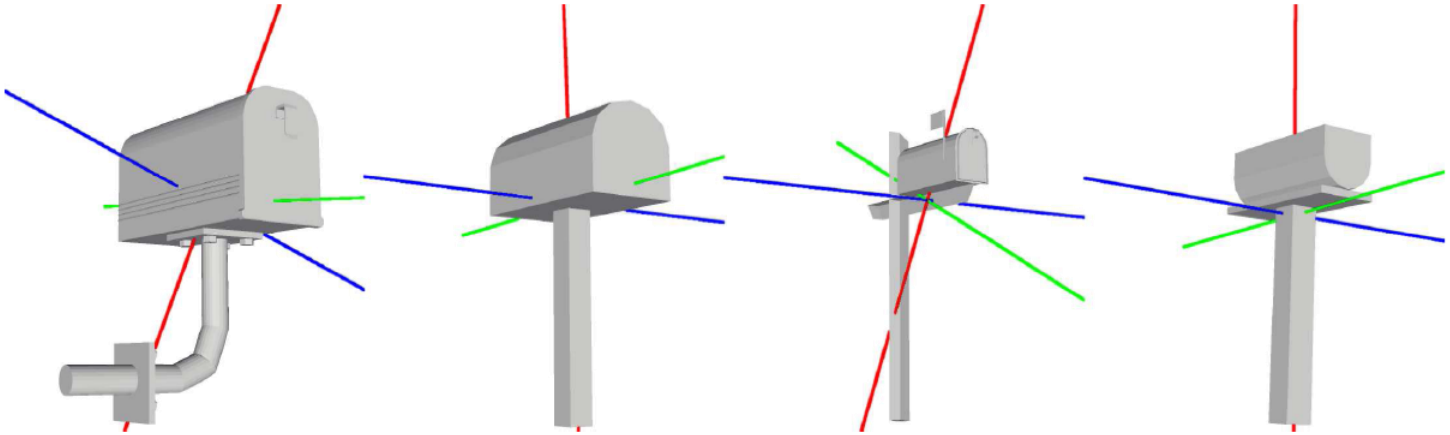
Center of mass and PCA



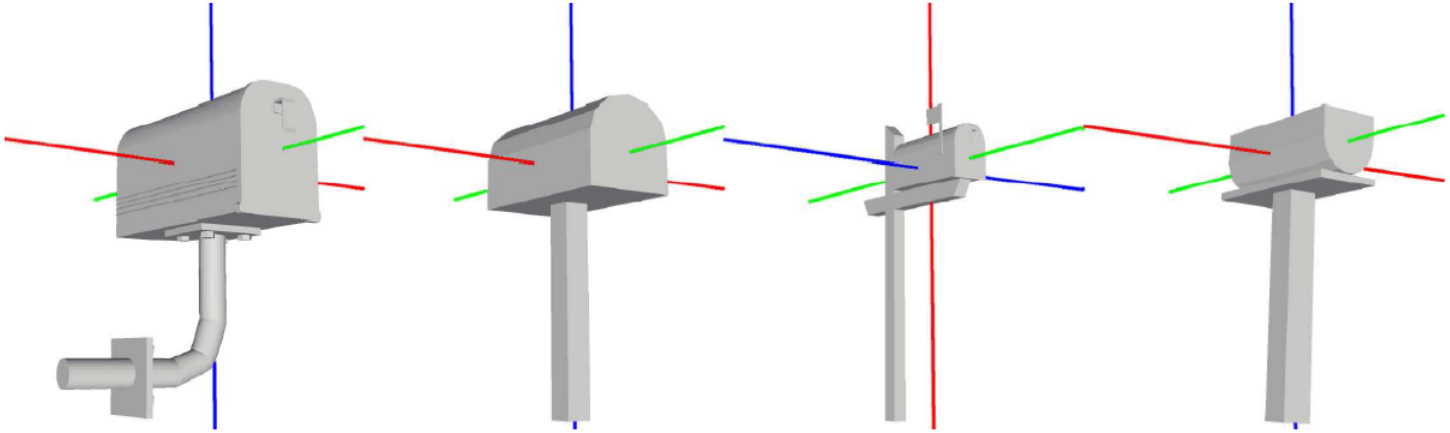
Center of symmetry and principal symmetry axes

Application: Alignment

Results:



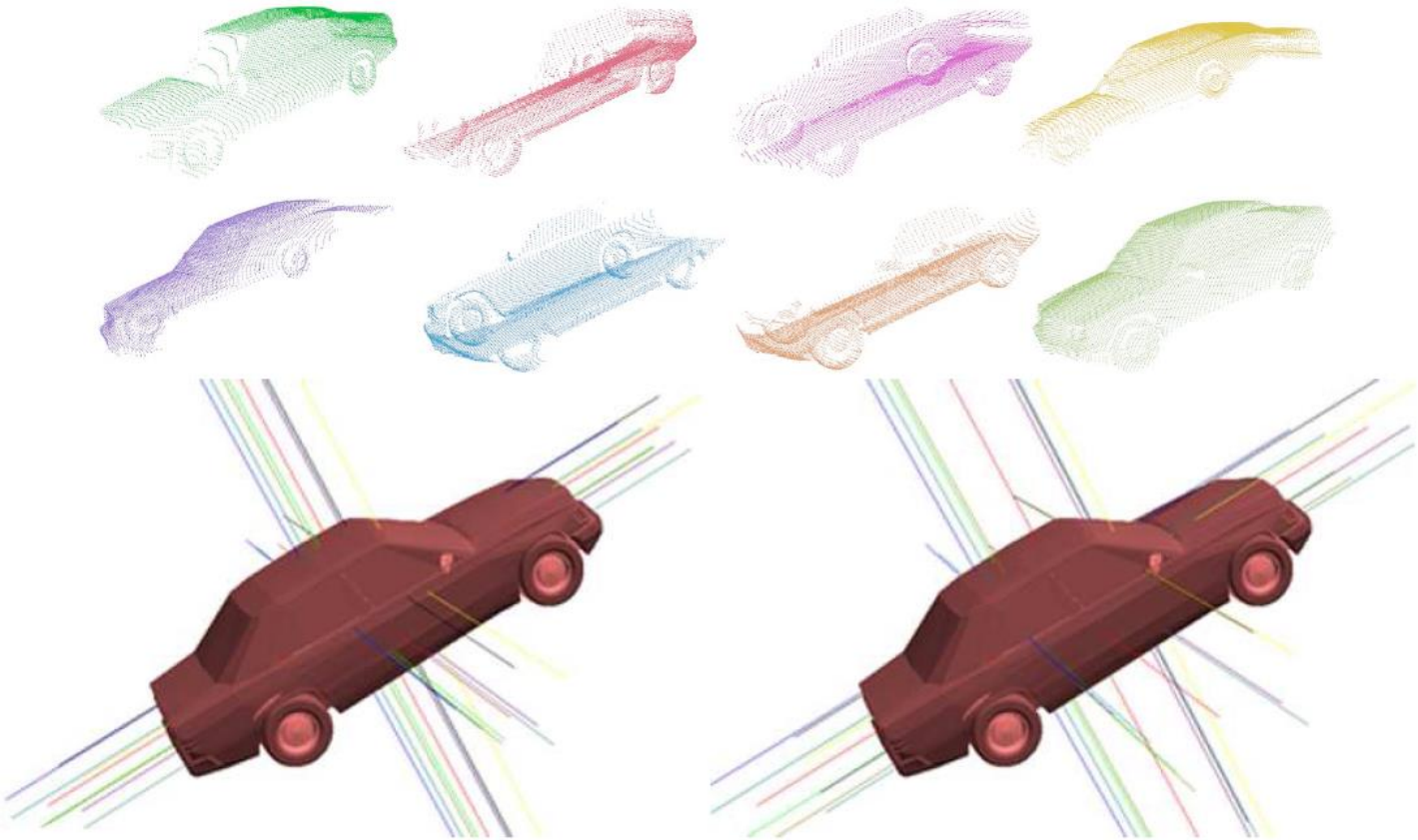
Center of mass and PCA



Center of symmetry and principal symmetry axes

Application: Alignment

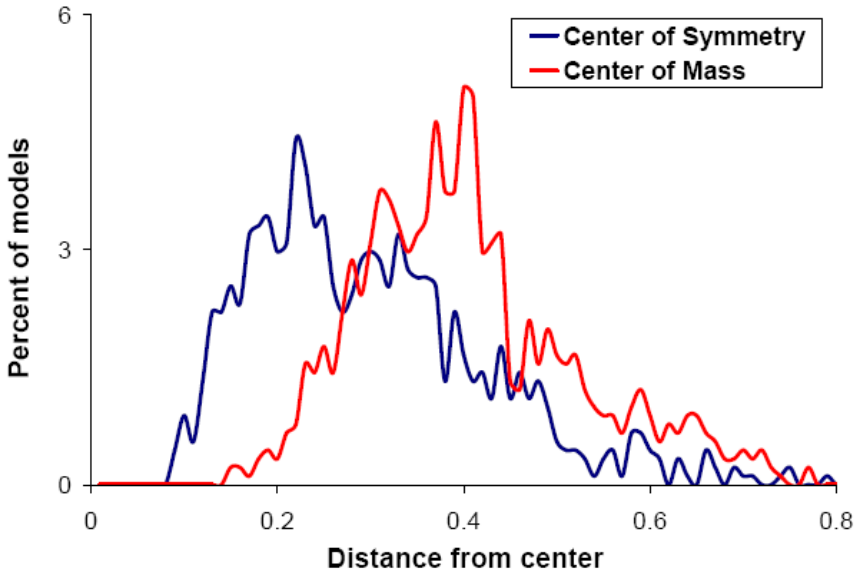
Results:

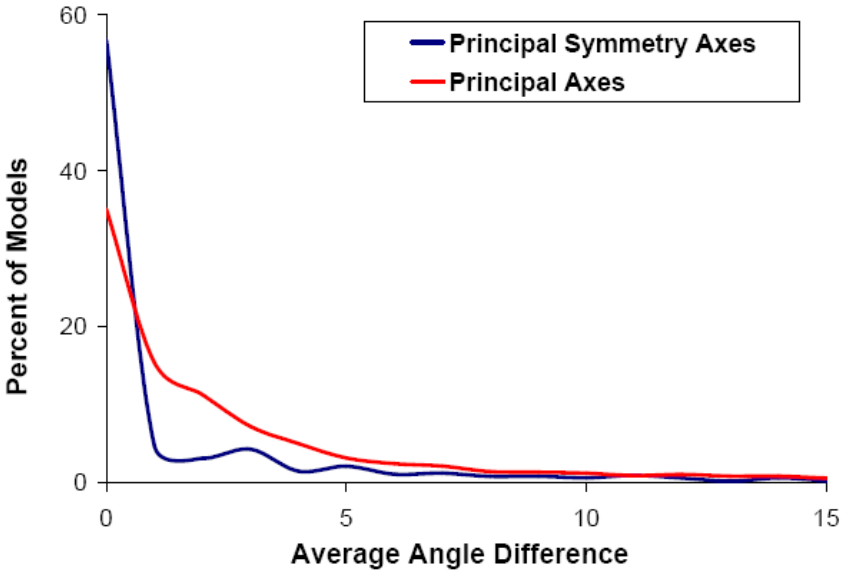


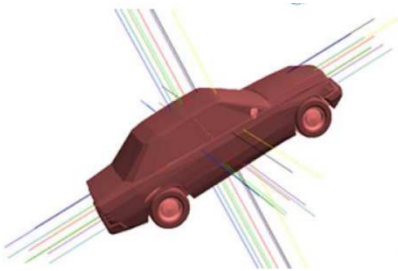
(Center of Symmetry) (Center of Mass)

Application: Alignment

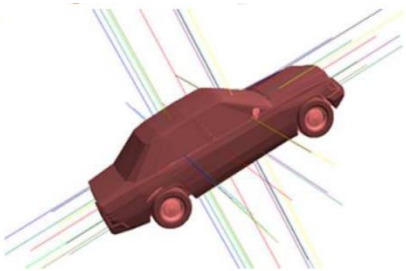
Results:







(Center of Symmetry)



(Center of Mass)

Applications

Alignment

Matching ←

Segmentation

Viewpoint selection

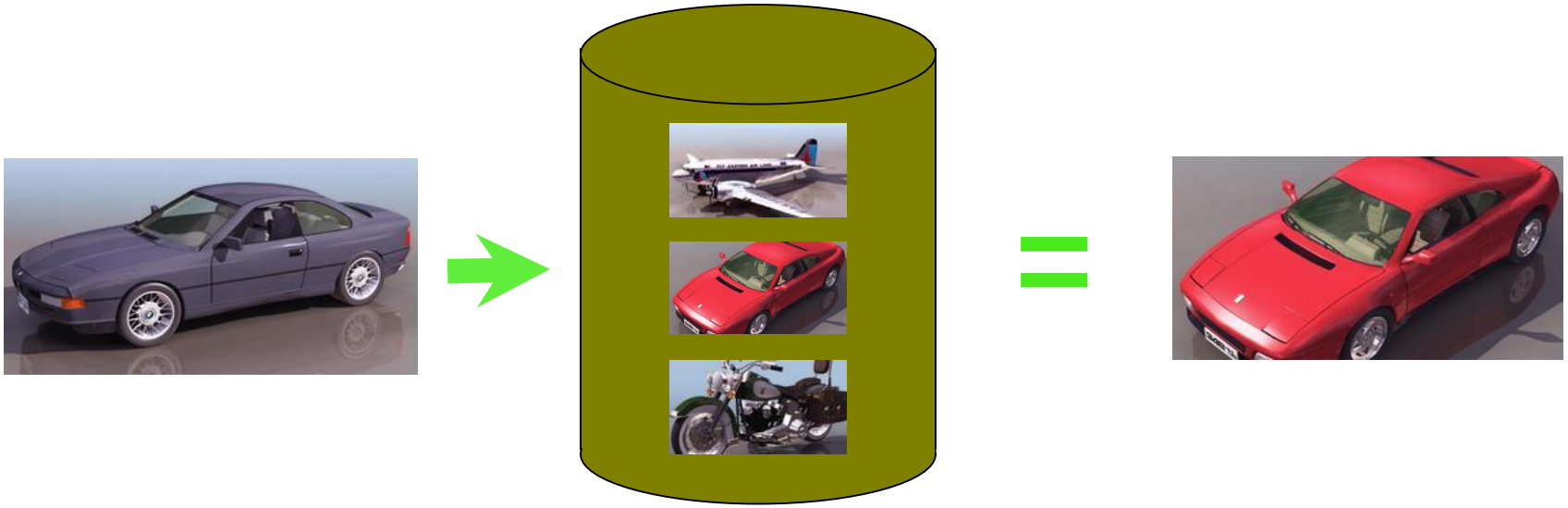
Simplification

Beautification

Texture synthesis

Application: Matching

Motivation: similarity search of database



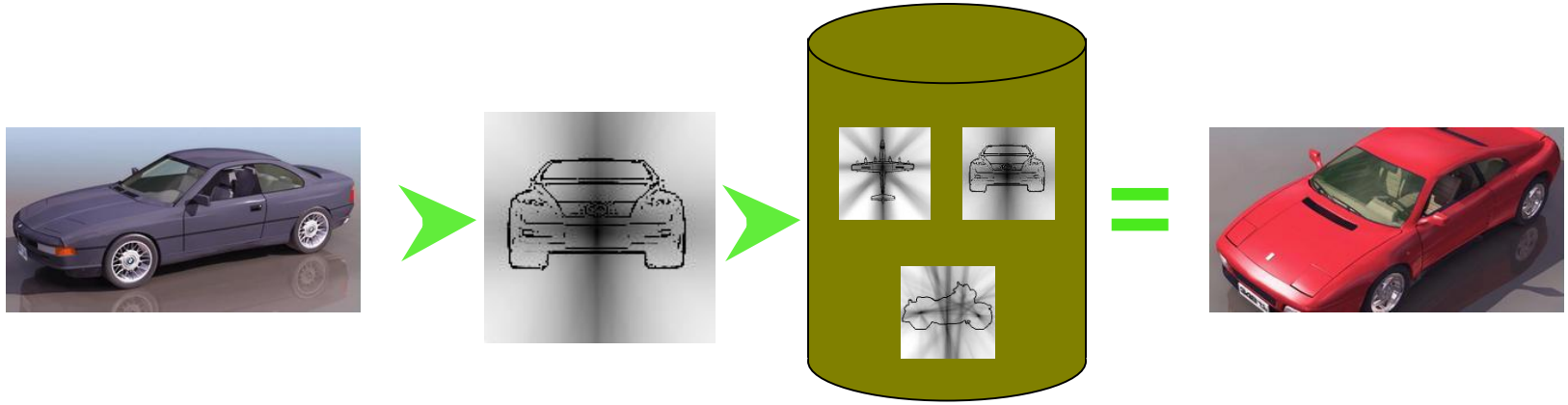
Query Database Best Match

Application: Matching

Observation: symmetry is more consistent than shape for some object classes

Application: Matching

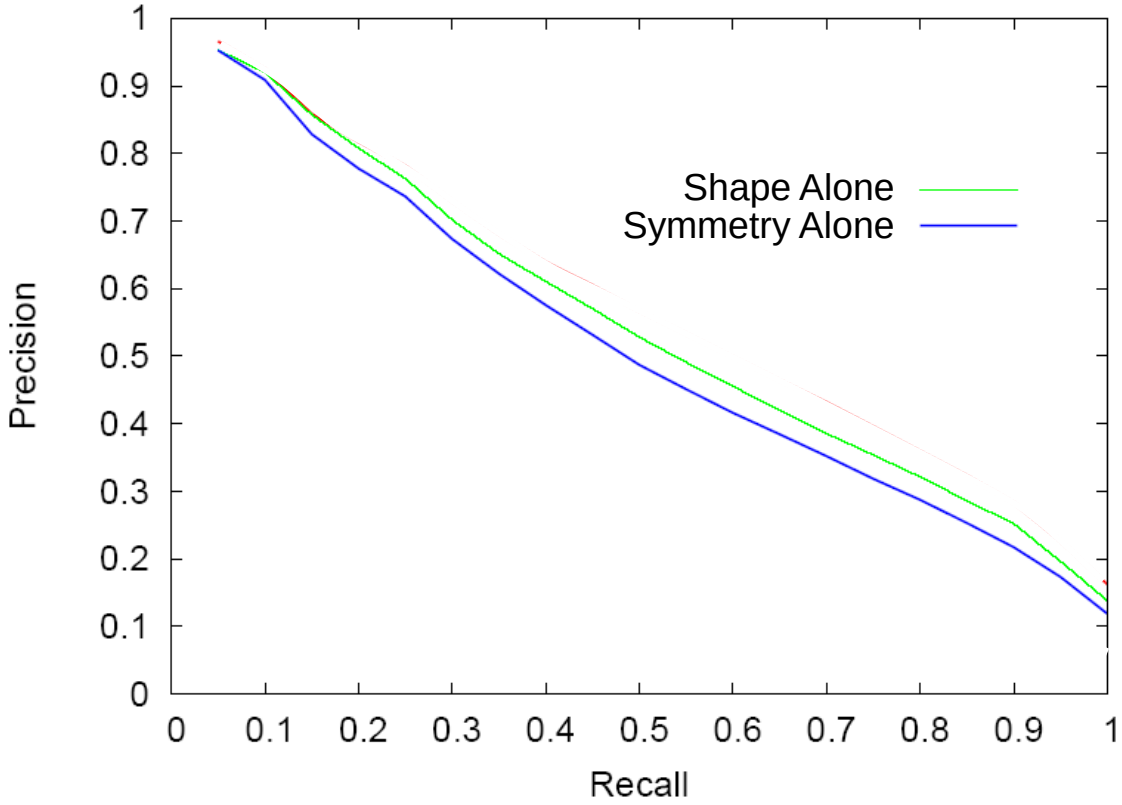
Approach: use symmetry transform (or descriptor) as shape descriptor



Query | Symmetry Transform | Database | Best Match

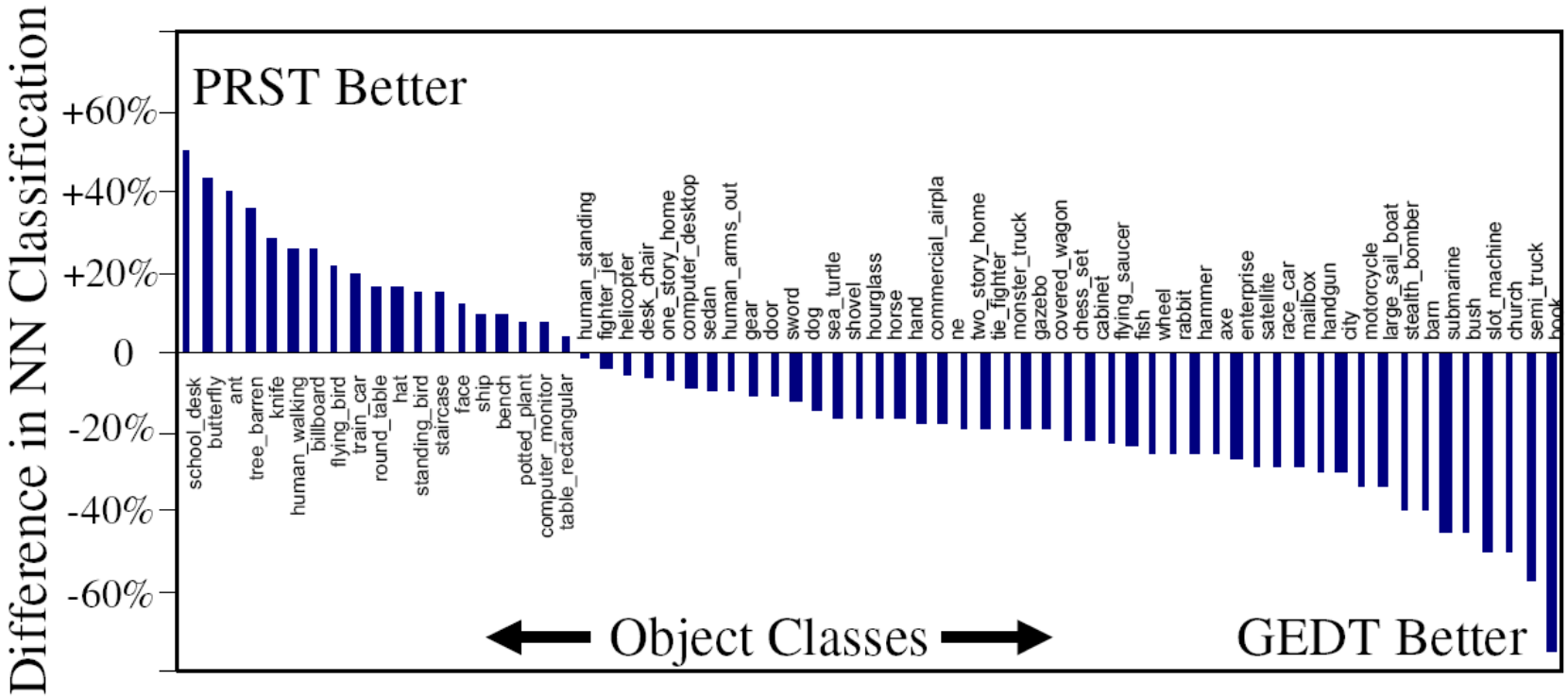
Application: Matching

Results: symmetry is not as discriminating as shape



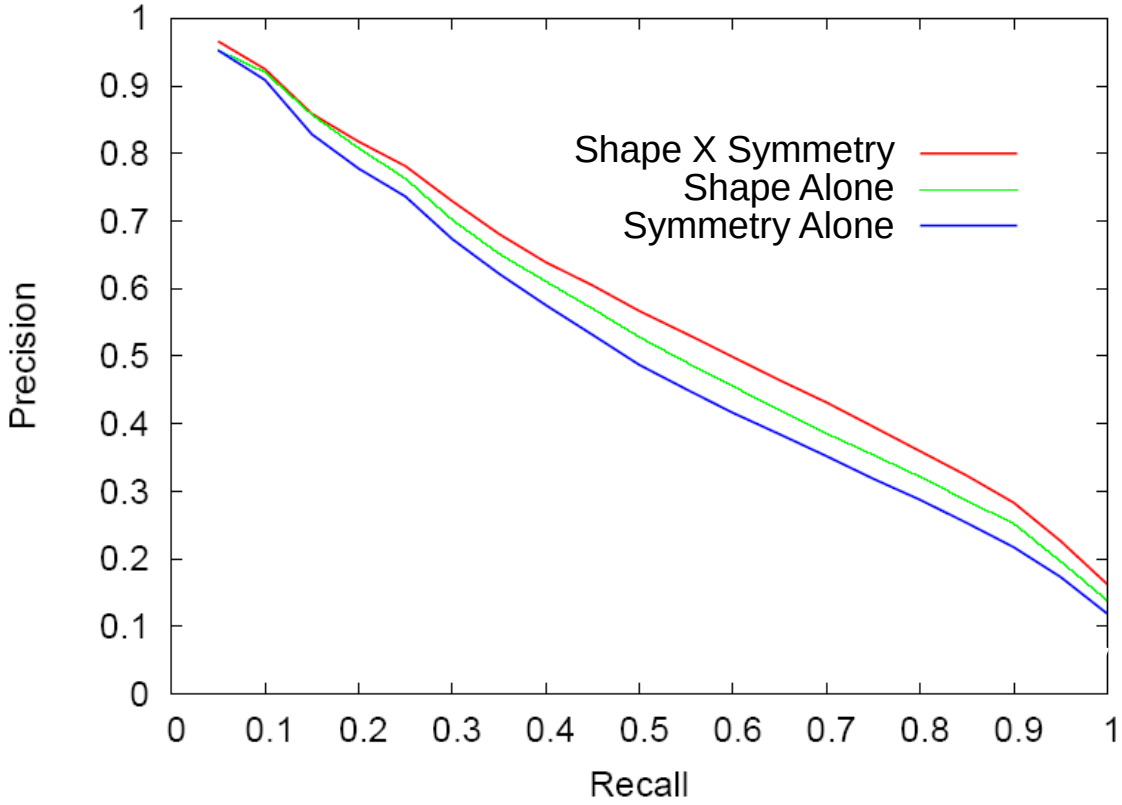
Application: Matching

Results: symmetry is not as discriminating as shape, but it is better for some classes



Application: Matching

Results: symmetry is not as discriminating as shape, but it is better for some classes, and so the two together are better than either alone



Applications

Alignment

Matching

Segmentation ←

Viewpoint selection

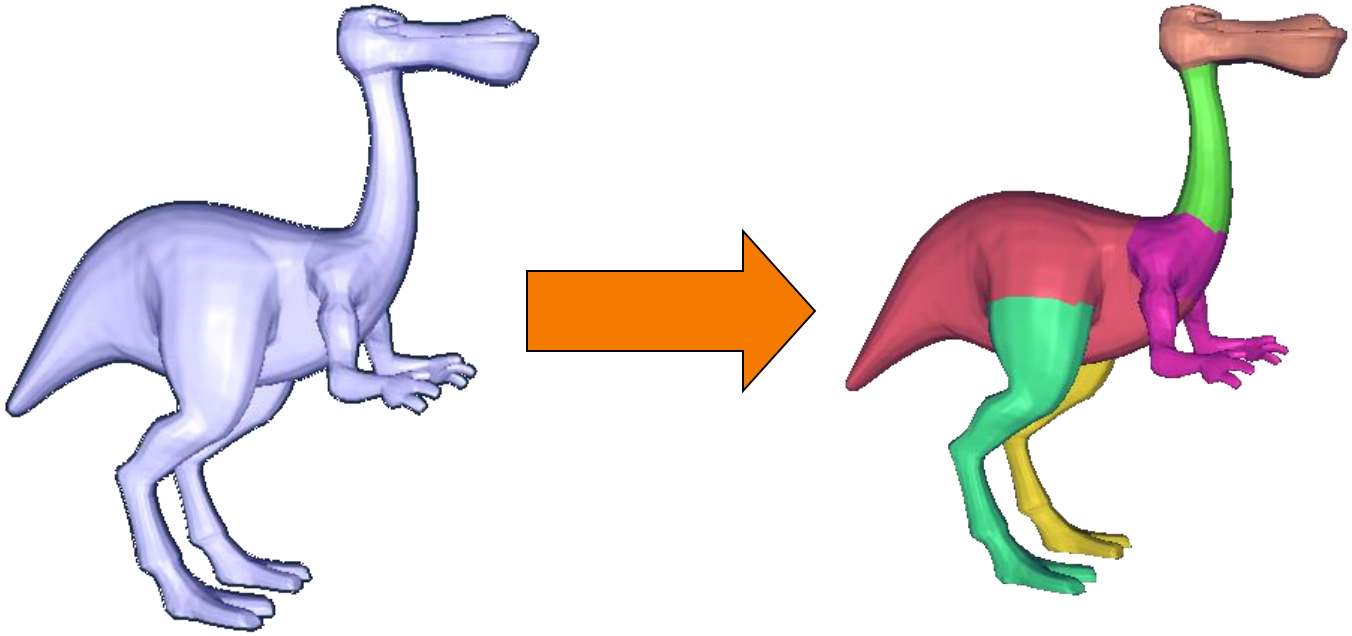
Simplification

Beautification

Texture synthesis

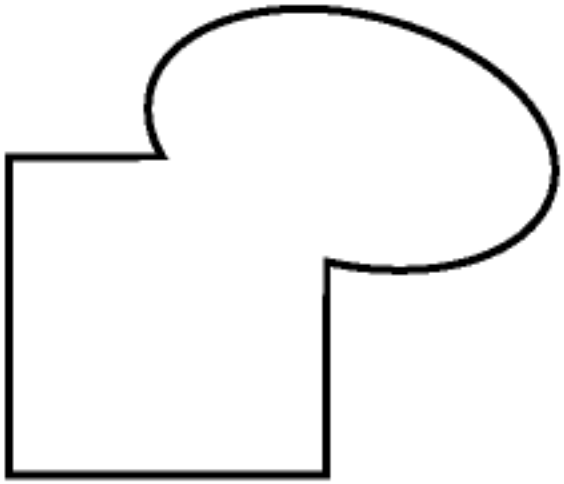
Application: Segmentation

Motivation: animation, modeling by parts, etc.



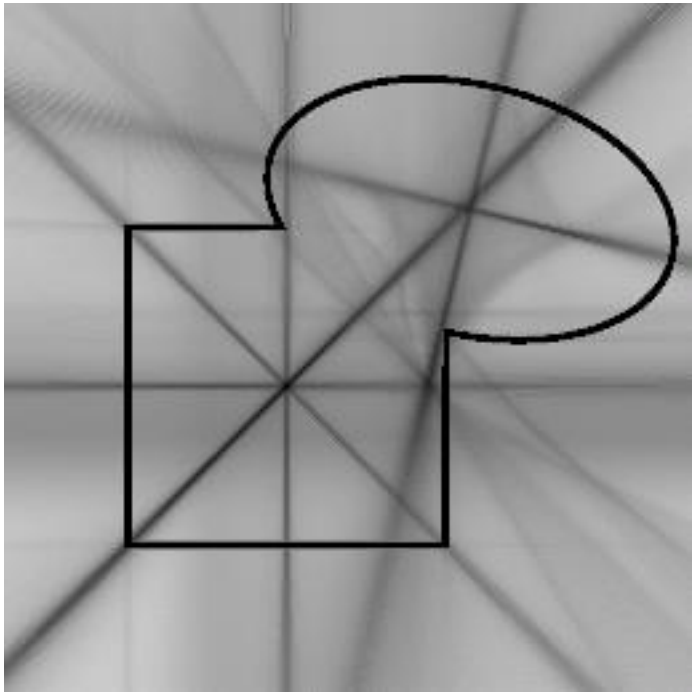
Application: Segmentation

Observation: distinct parts have strong local symmetries not shared by other parts



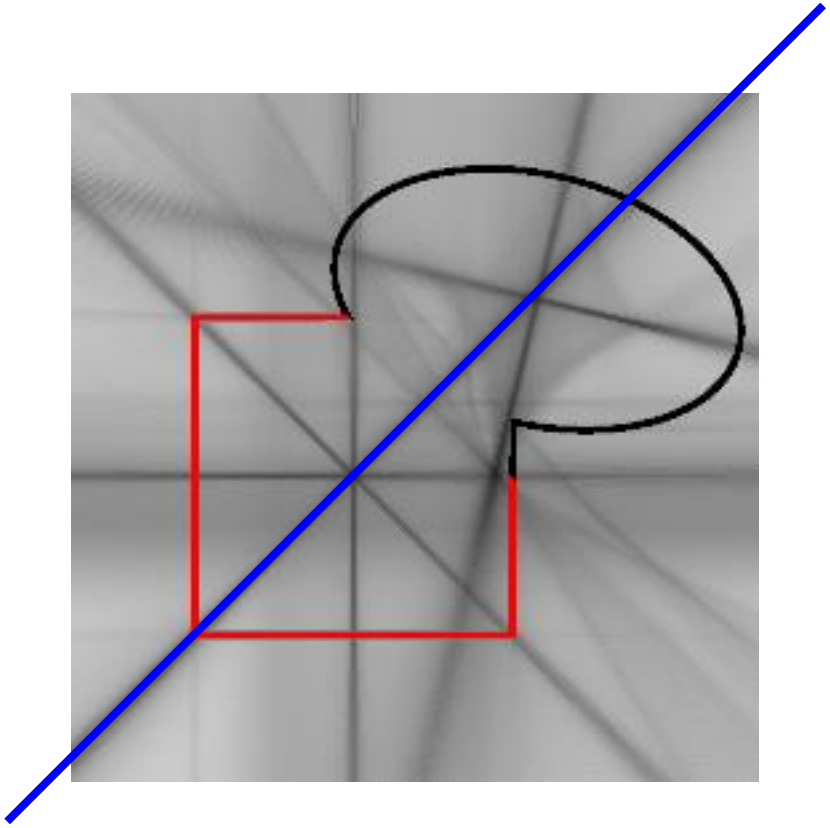
Application: Segmentation

Observation: distinct parts have strong local symmetries not shared by other parts



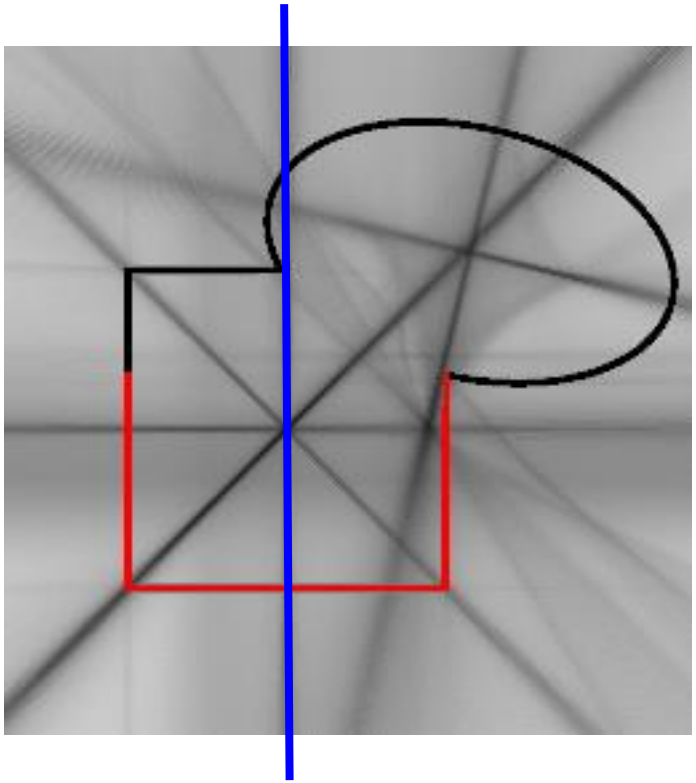
Application: Segmentation

Observation: distinct parts have strong local symmetries not shared by other parts



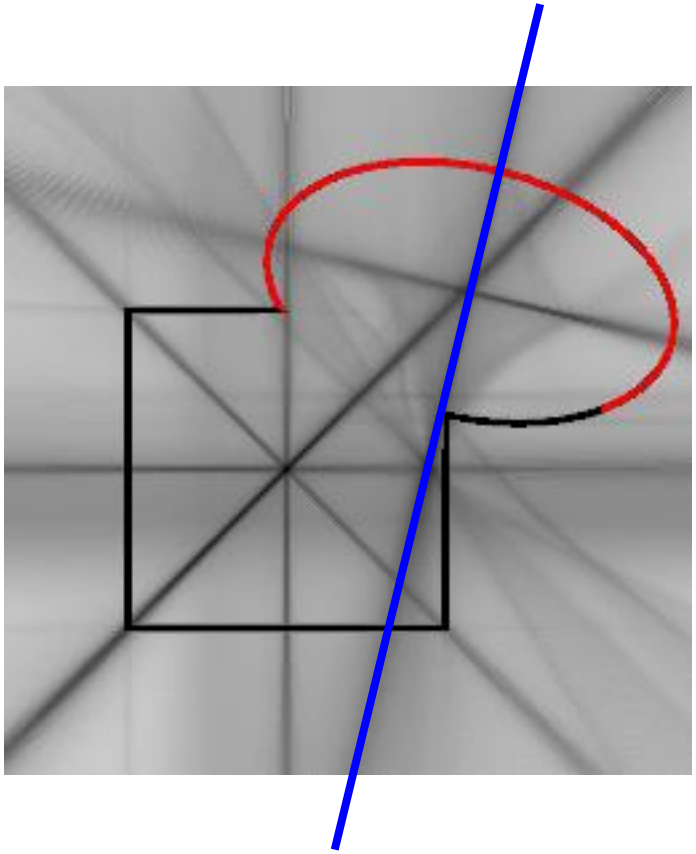
Application: Segmentation

Observation: distinct parts have strong local symmetries not shared by other parts



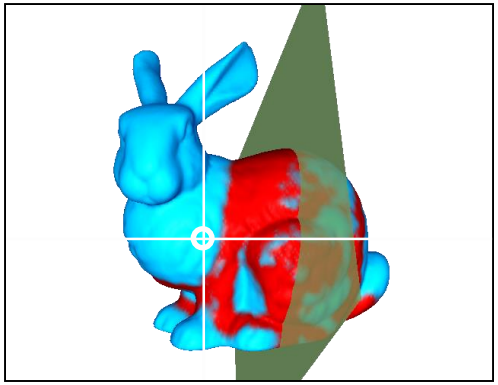
Application: Segmentation

Observation: distinct parts have strong local symmetries not shared by other parts

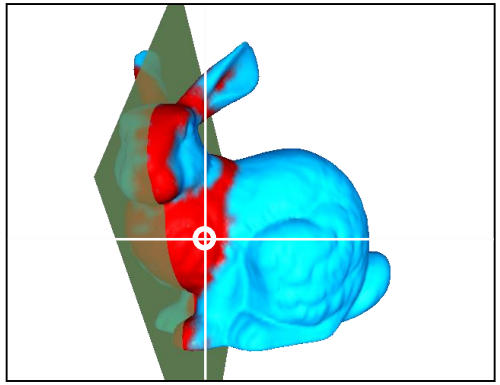


Application: Segmentation

Approach: cluster points on the surface by how much they support different symmetries

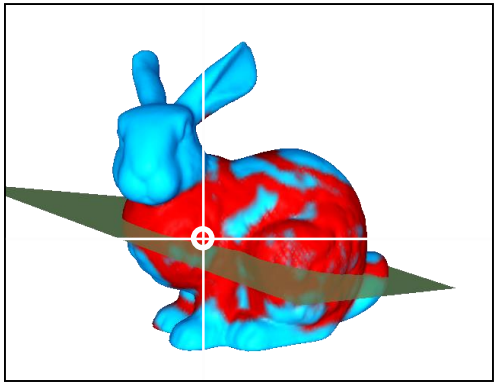


Support = 0.1



Support = 0.5

.....

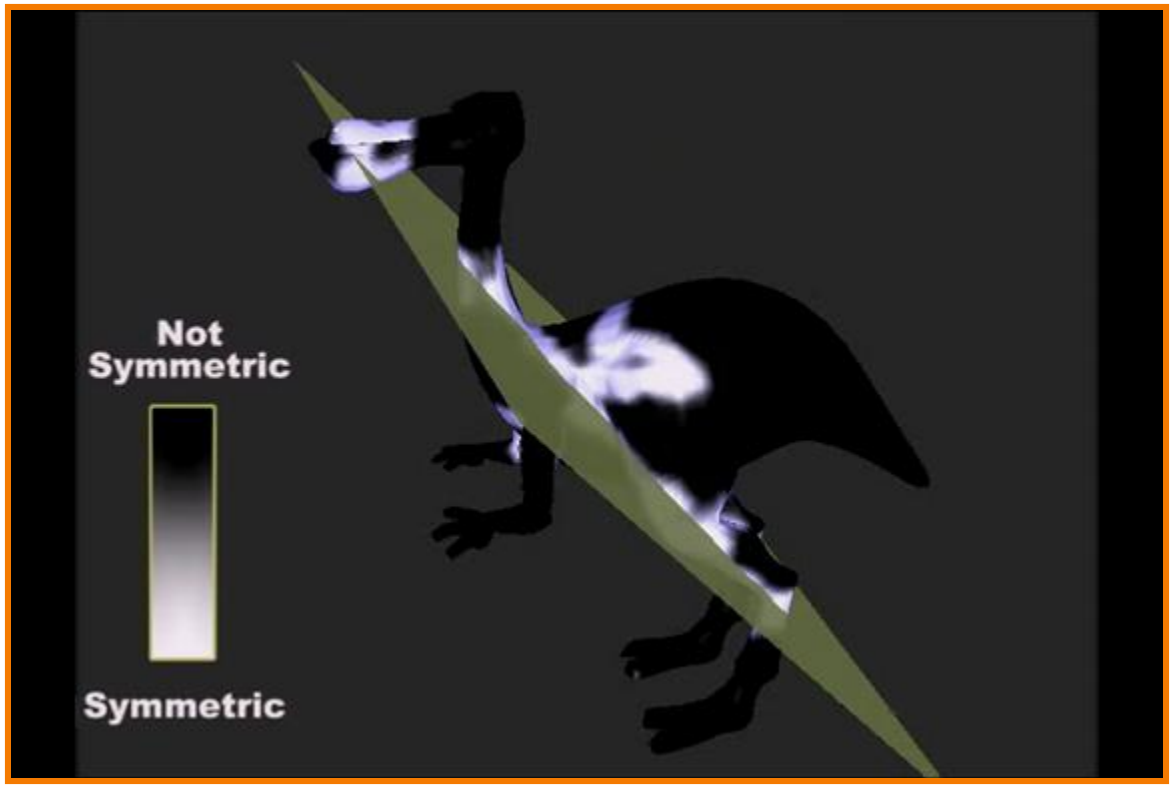


Support = 0.9

Symmetry Vector = { 0.1 , 0.5 , …. , 0.9 }

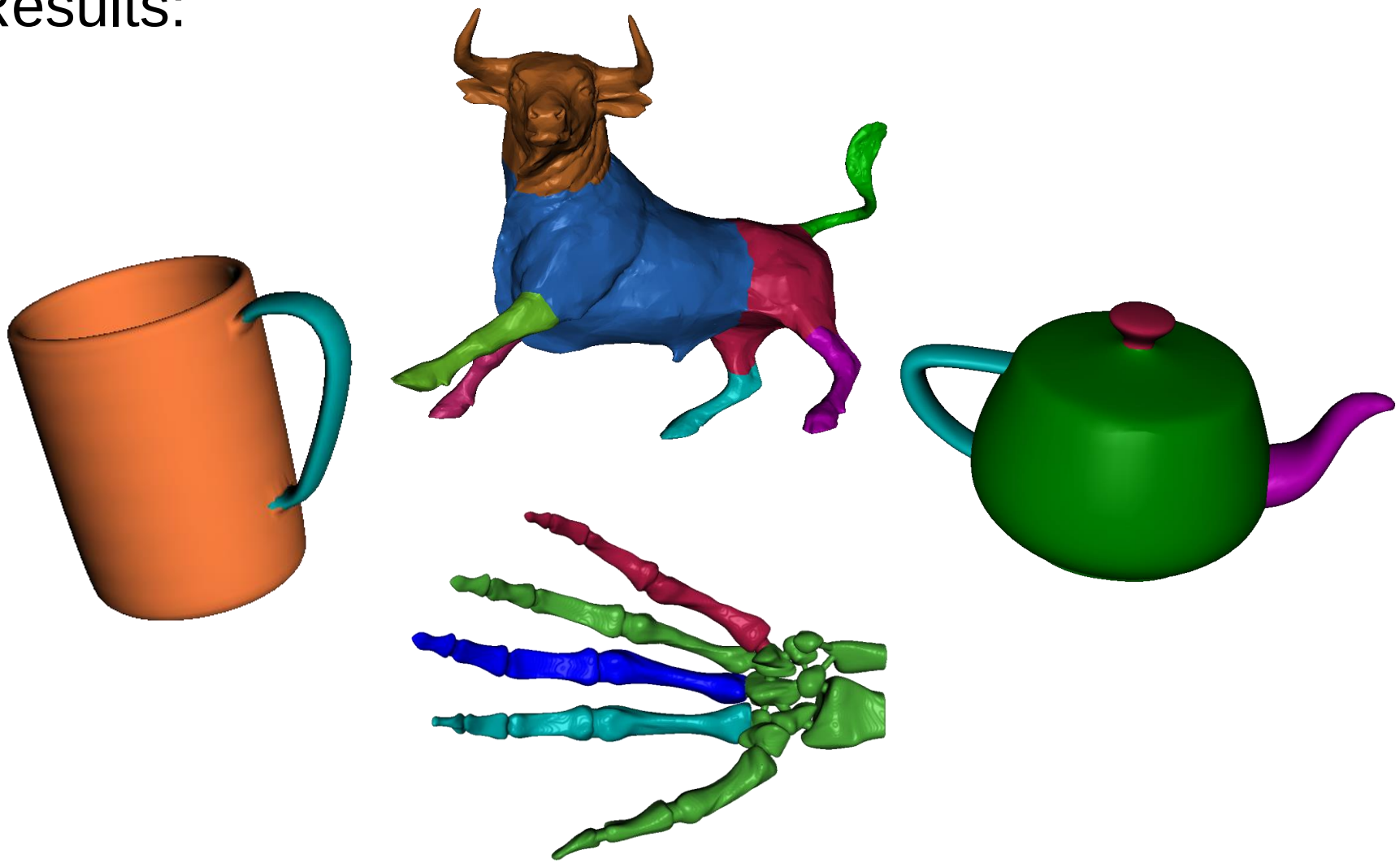
Application: Segmentation

Approach: cluster points on the surface by how much they support different symmetries



Application: Segmentation

Results:



Applications

Alignment

Matching

Segmentation

Viewpoint selection ←

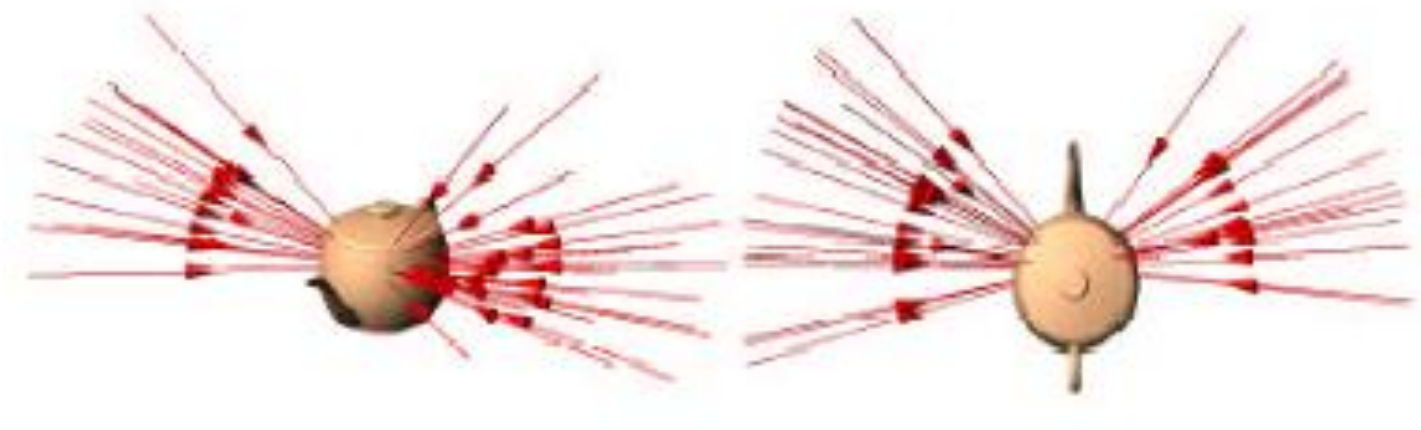
Simplification

Beautification

Texture synthesis

Application: Viewpoint Selection

Motivation: visualization, icon generation, etc.



Picture from Blanz et al. ‘99

Application: Viewpoint Selection

Observation: symmetry represents redundancy

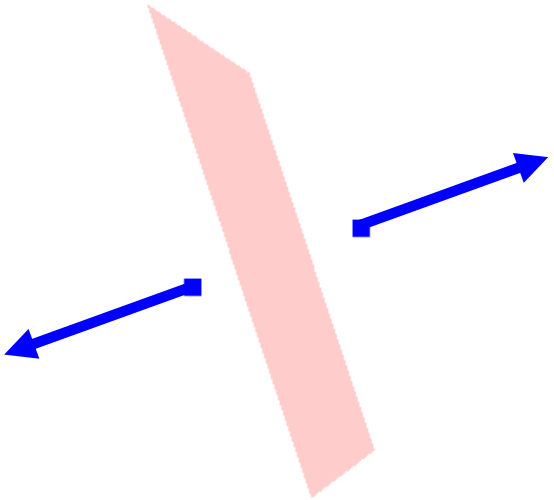
Bad Viewpoint

Application: Viewpoint Selection

Approach: Minimize visible symmetry

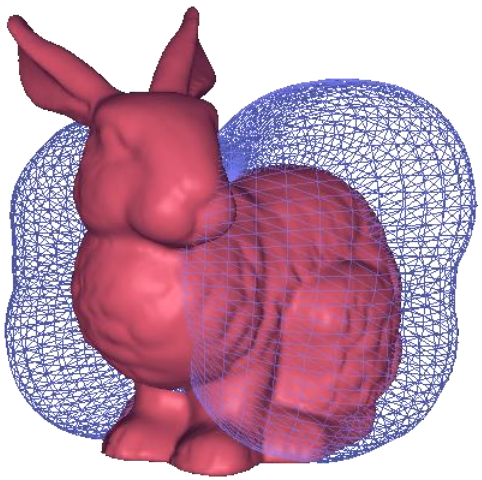
- Every plane of symmetry votes for a viewing direction perpendicular to it

Best Viewing Directions

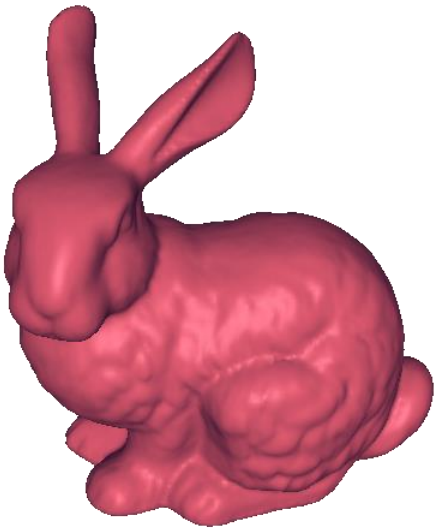


Application: Viewpoint Selection

Results:



Viewpoint Function

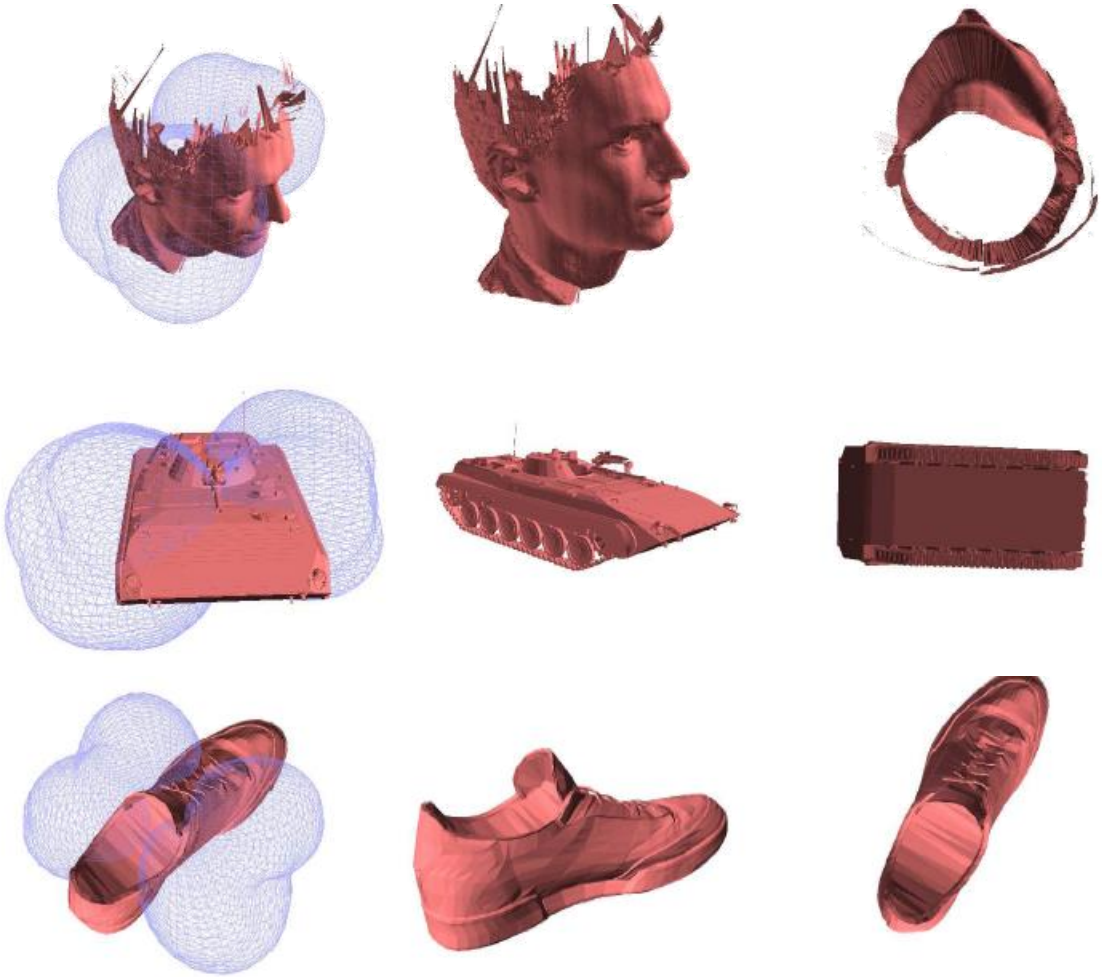


Best Viewpoint

Worst Viewpoint

Application: Viewpoint Selection

Results:



Viewpoint Function | Best Viewpoint | Worst Viewpoint

Applications

Alignment

Matching

Segmentation

Viewpoint selection

Simplification ←

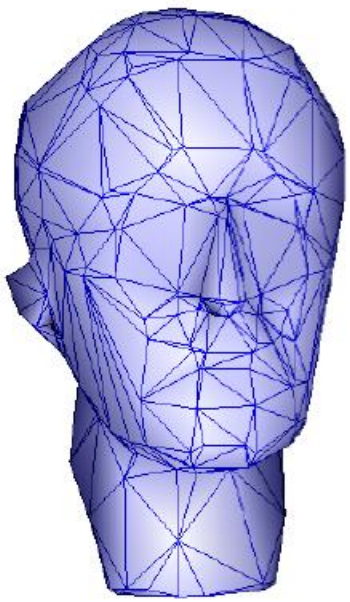
Beautification

Texture synthesis

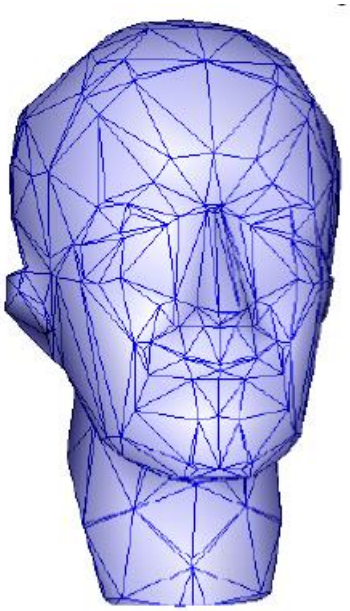
Application: Simplification

Motivation: preserve symmetry when simplify mesh

Input
Mesh



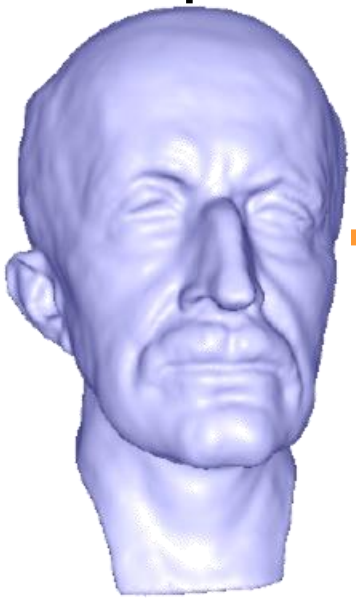
Standard
Simplification



Symmetric
Simplification

Application: Simplification

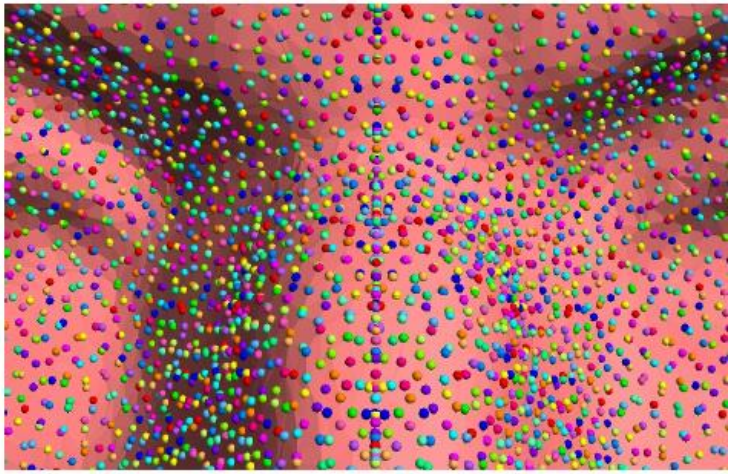
Approach 1: detect (approximate) symmetries and then preserve them as decimate



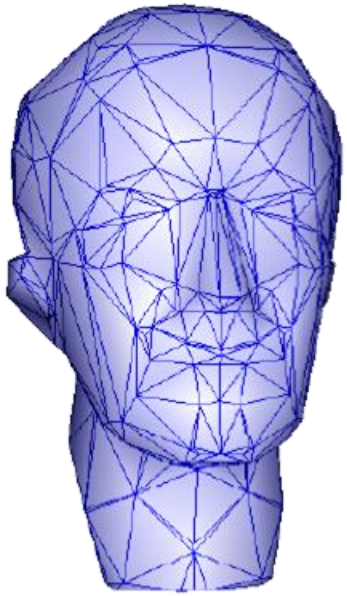
Input Mesh

Symmetry Analysis

Symmetry Aware Decimation



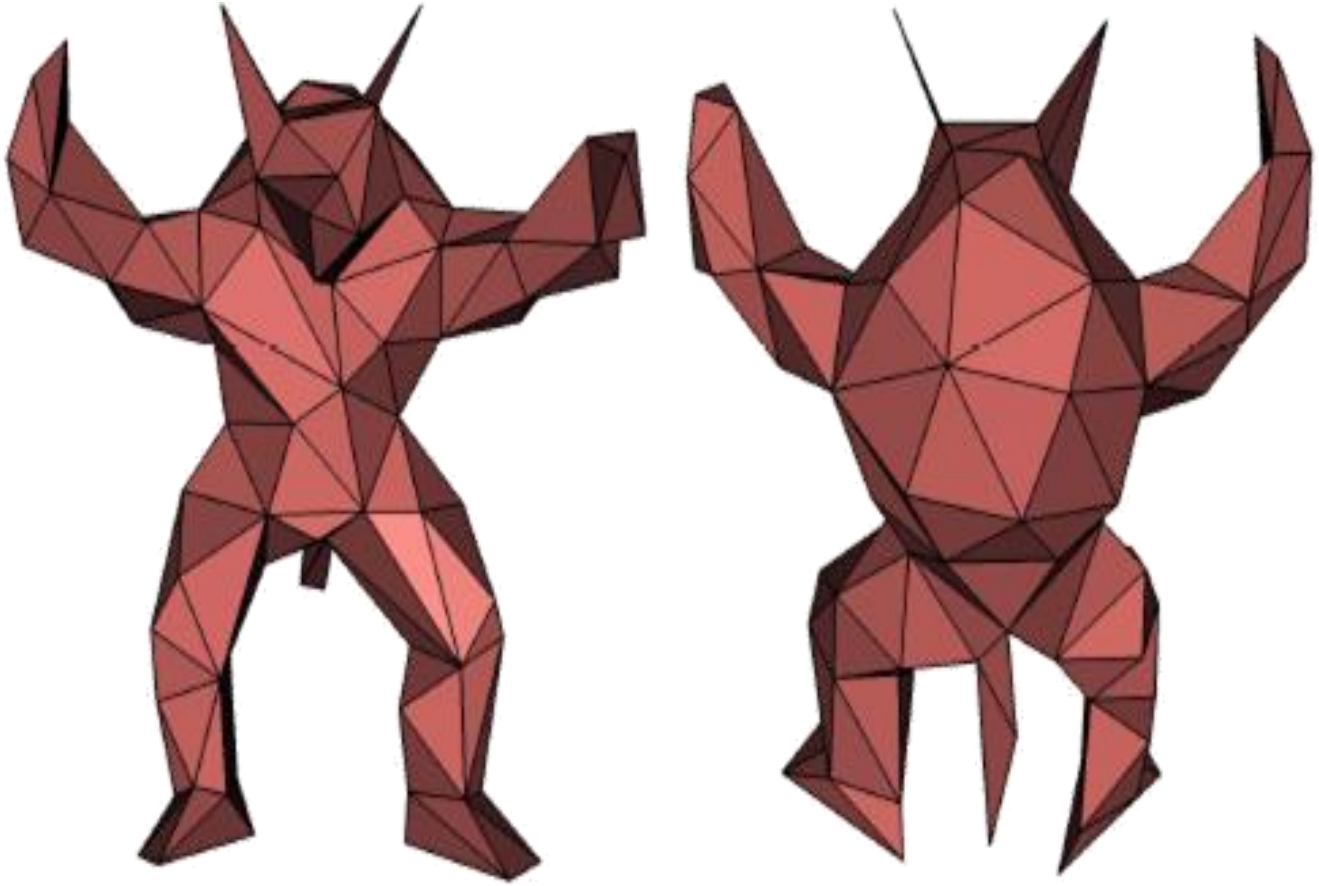
Symmetric Correspondences



Symmetric Simplification

Application: Simplification

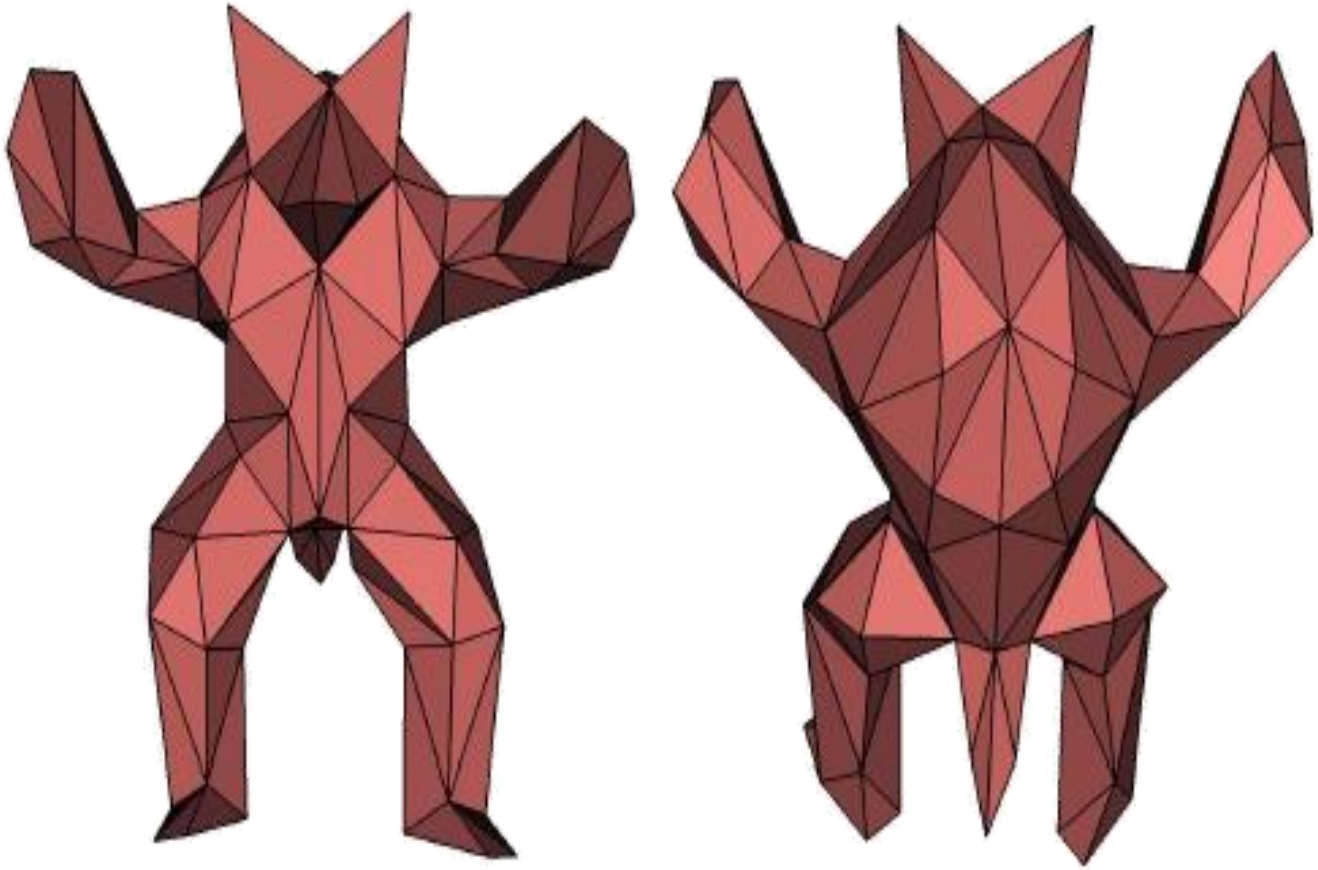
Results:



Original Qslim

Application: Simplification

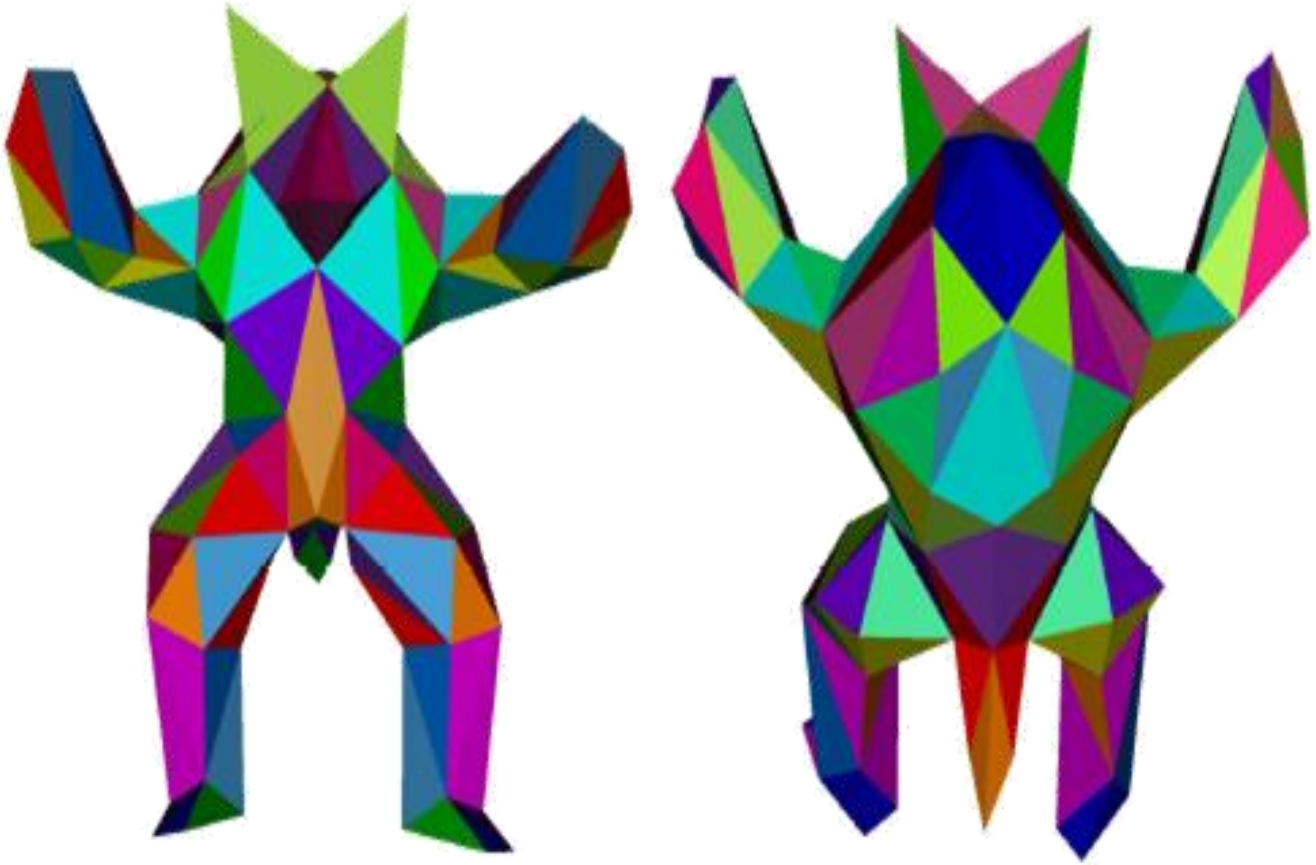
Results:



Symmetry-Preserving Qslim

Application: Simplification

Results:

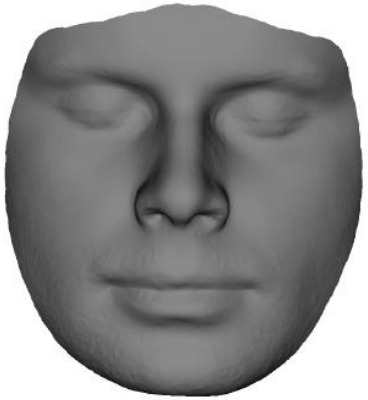


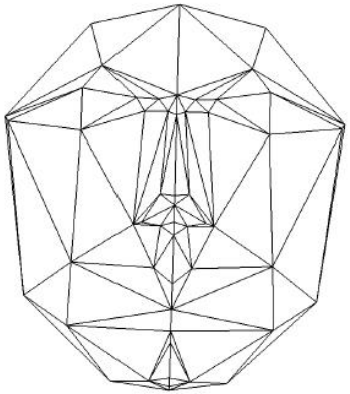
Symmetry-Preserving Qslim

Application: Parameterization

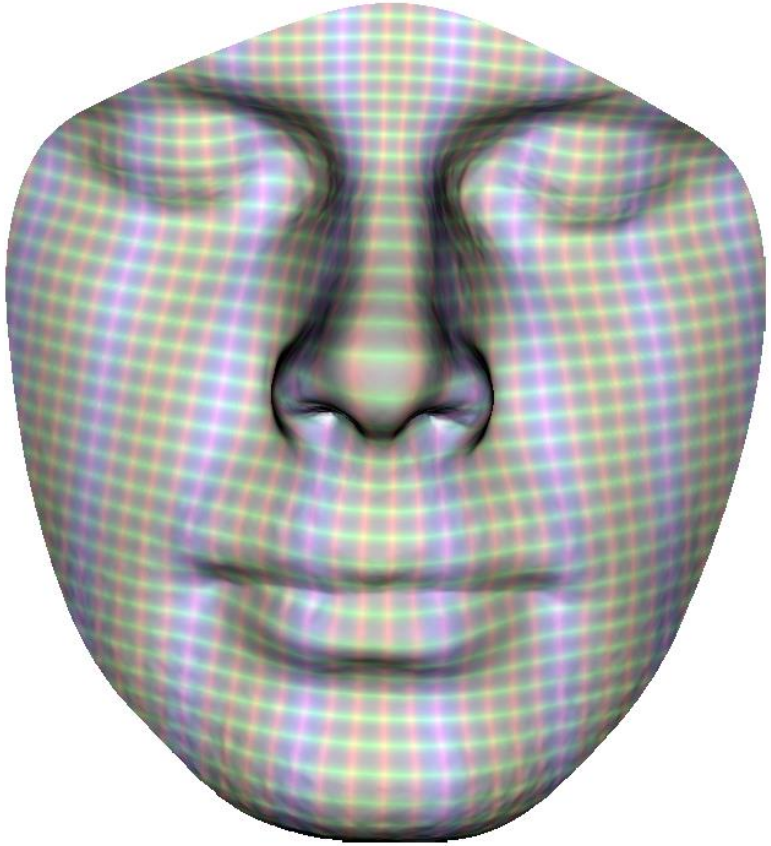
Preserve symmetries in base domain

Input Mesh





Base Domain



Parameterization

Applications

Alignment

Matching

Segmentation

Viewpoint selection

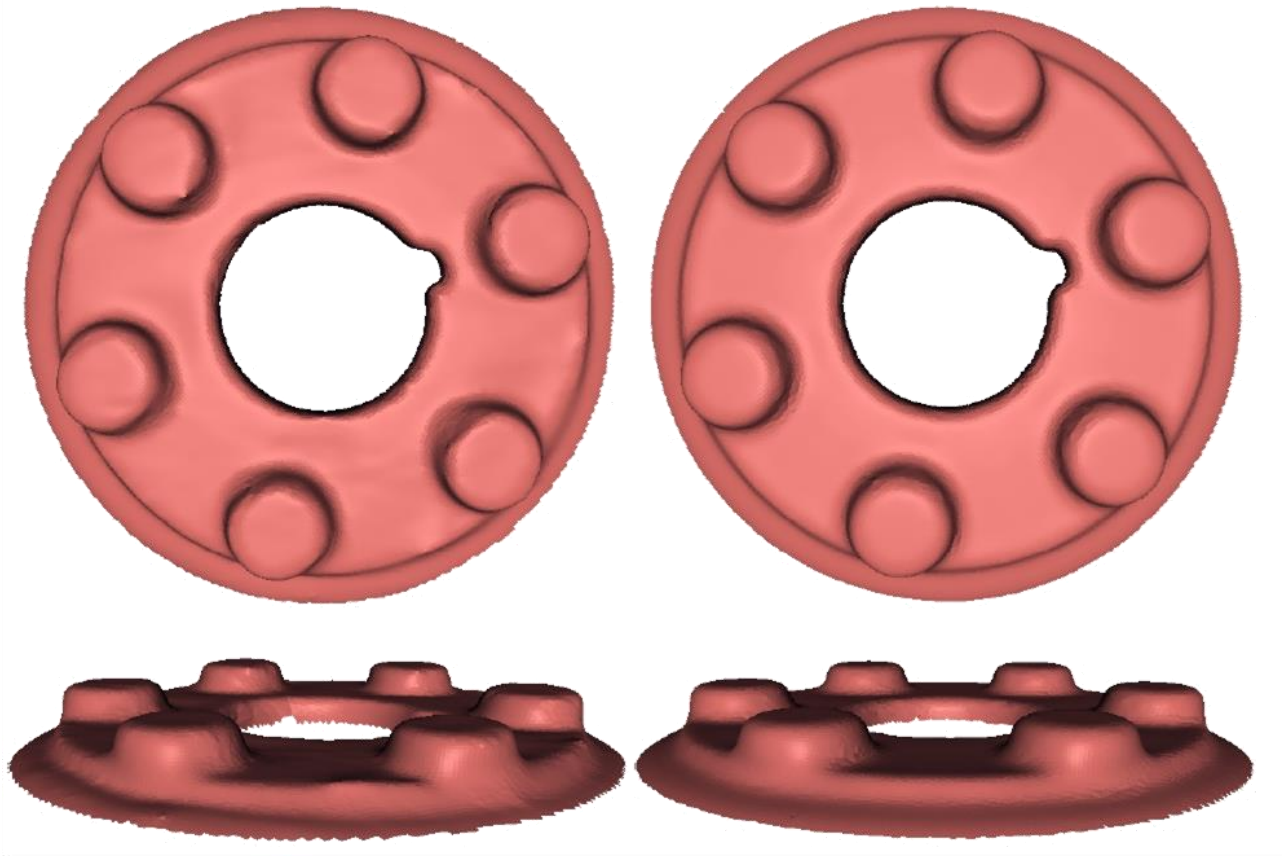
Simplification

Beautification ←

Texture synthesis

Application: Beautification

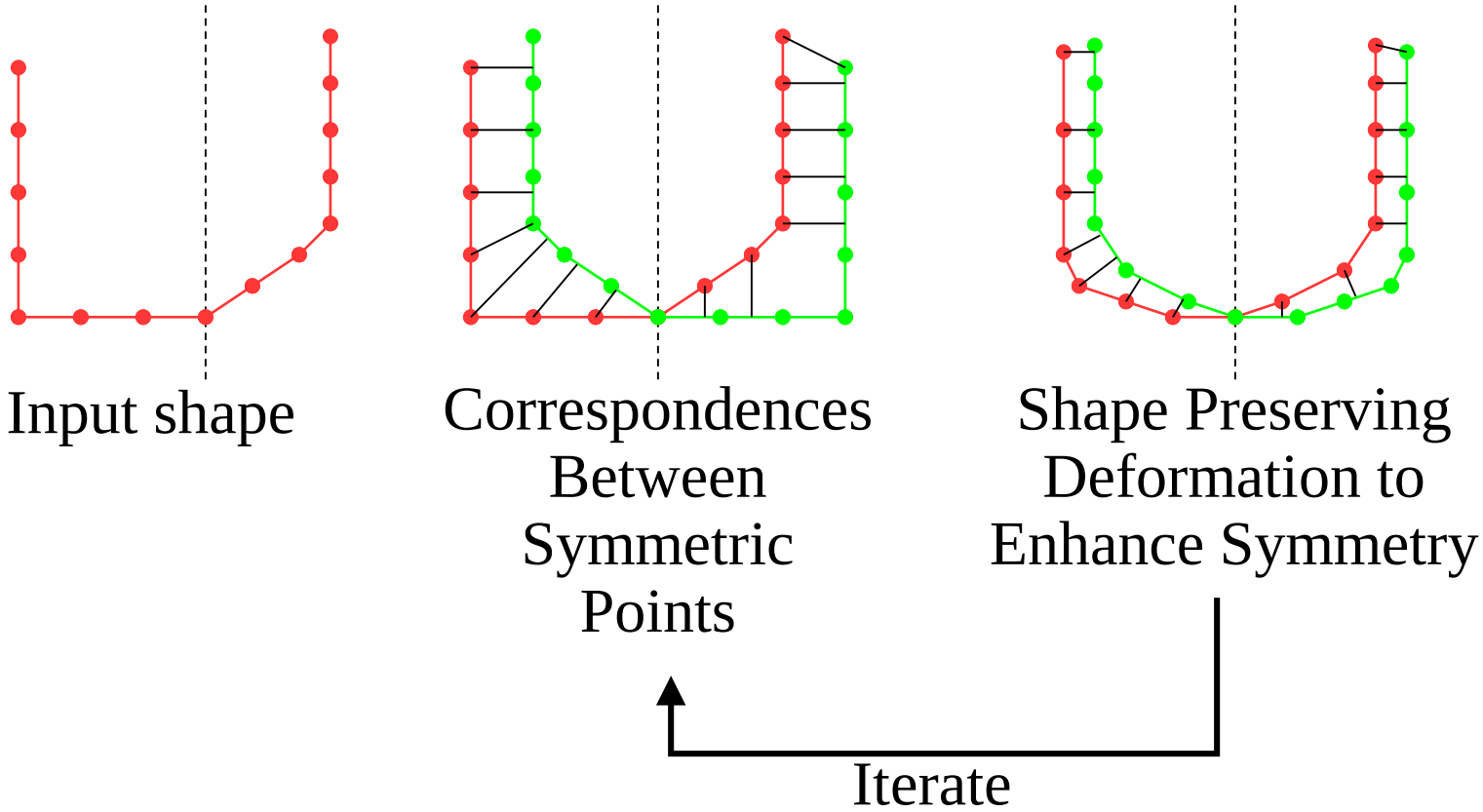
Goal: make meshes of symmetric objects perfectly symmetric



Input Mesh Symmetric Mesh

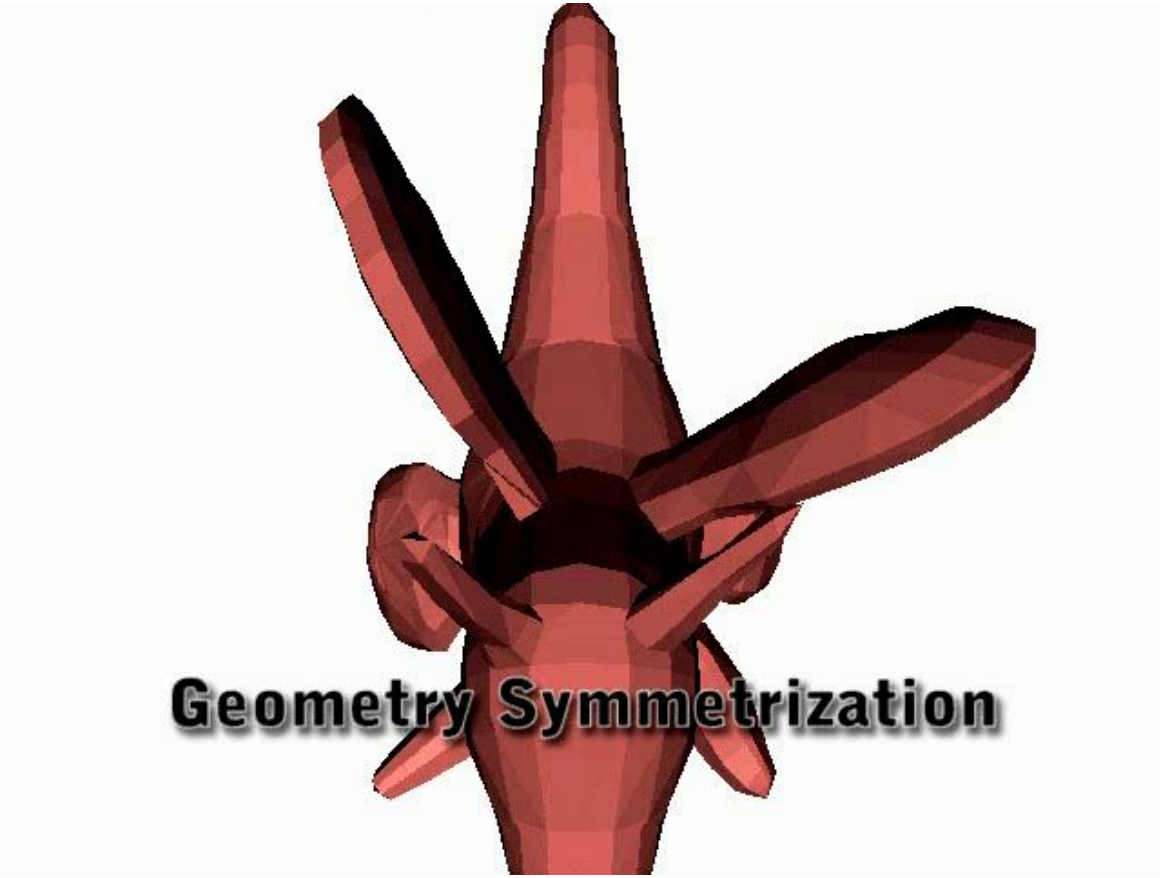
Application: Beautification

Approach: iterative non-rigid deformation to align symmetric points (symmetrization)



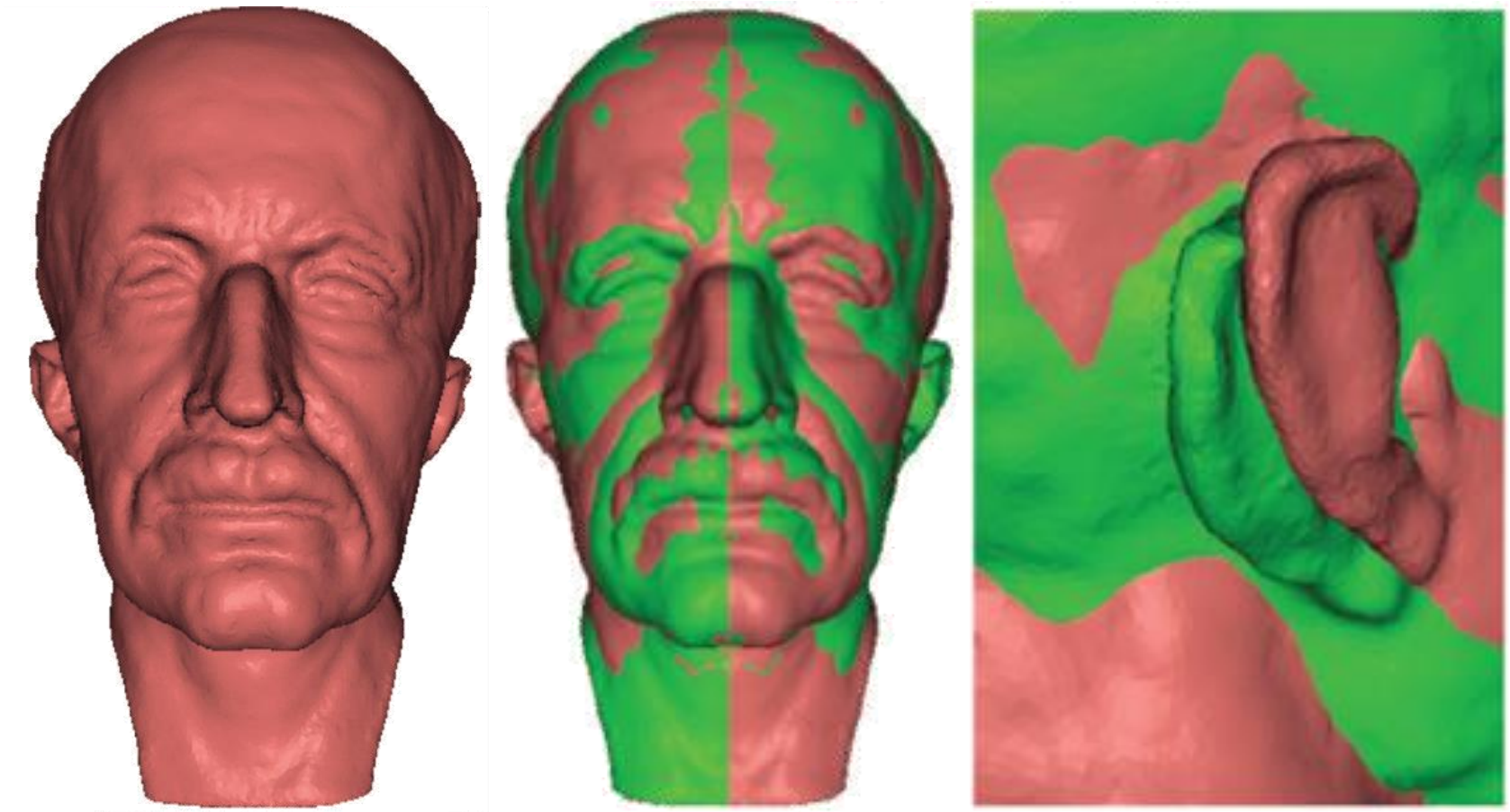
Application: Beautification

Approach: iterative non-rigid deformation to align symmetric points (symmetrization)



Application: Beautification

Results:

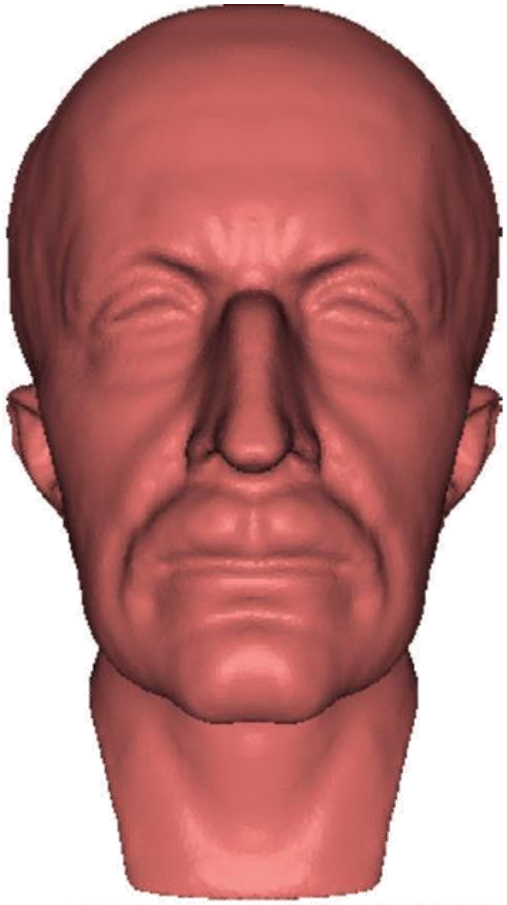


Input Mesh

Mesh overlaid with its reflection

Application: Beautification

Results:

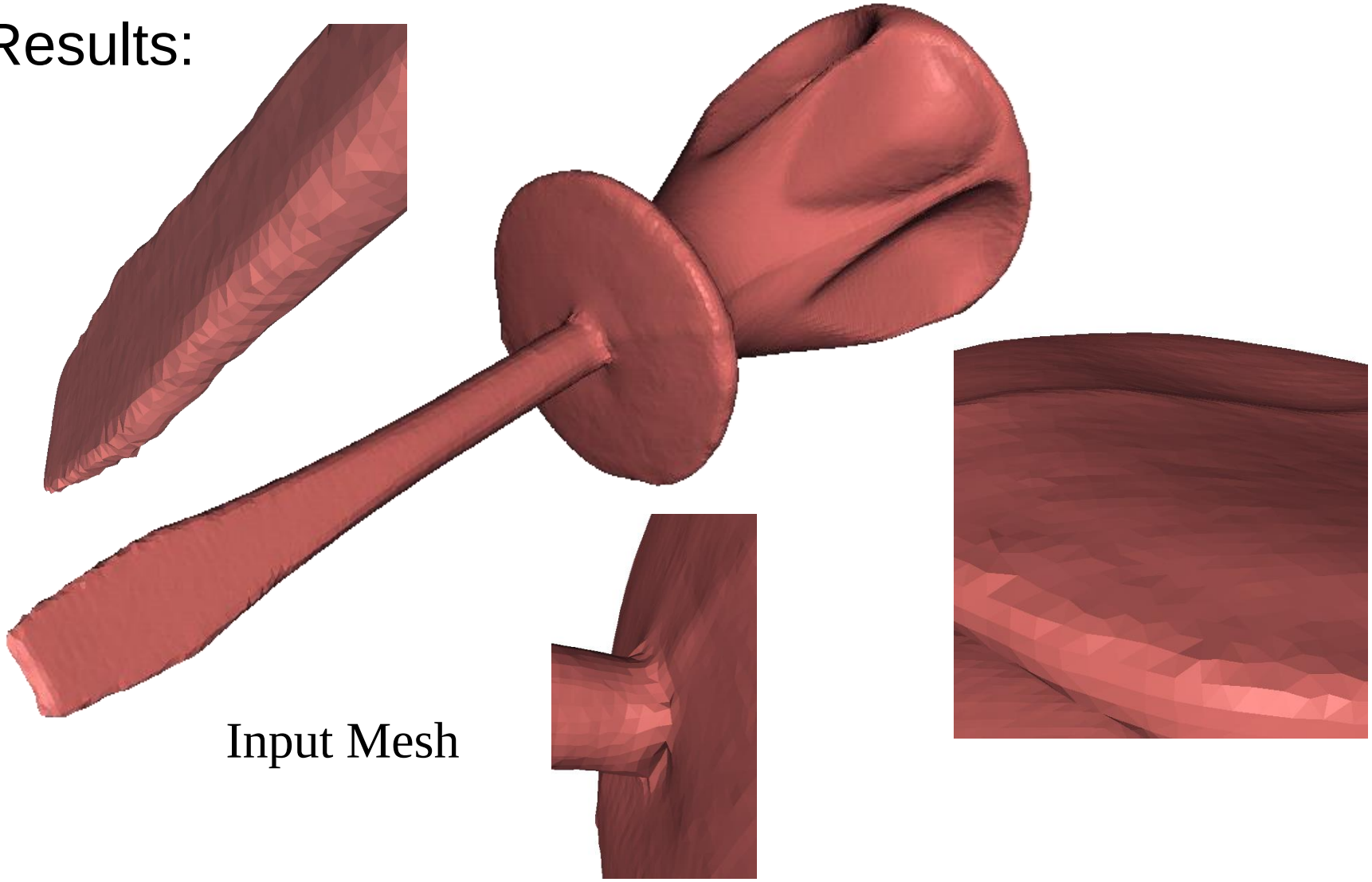


Symmetrized Mesh

Mesh overlaid with its reflection

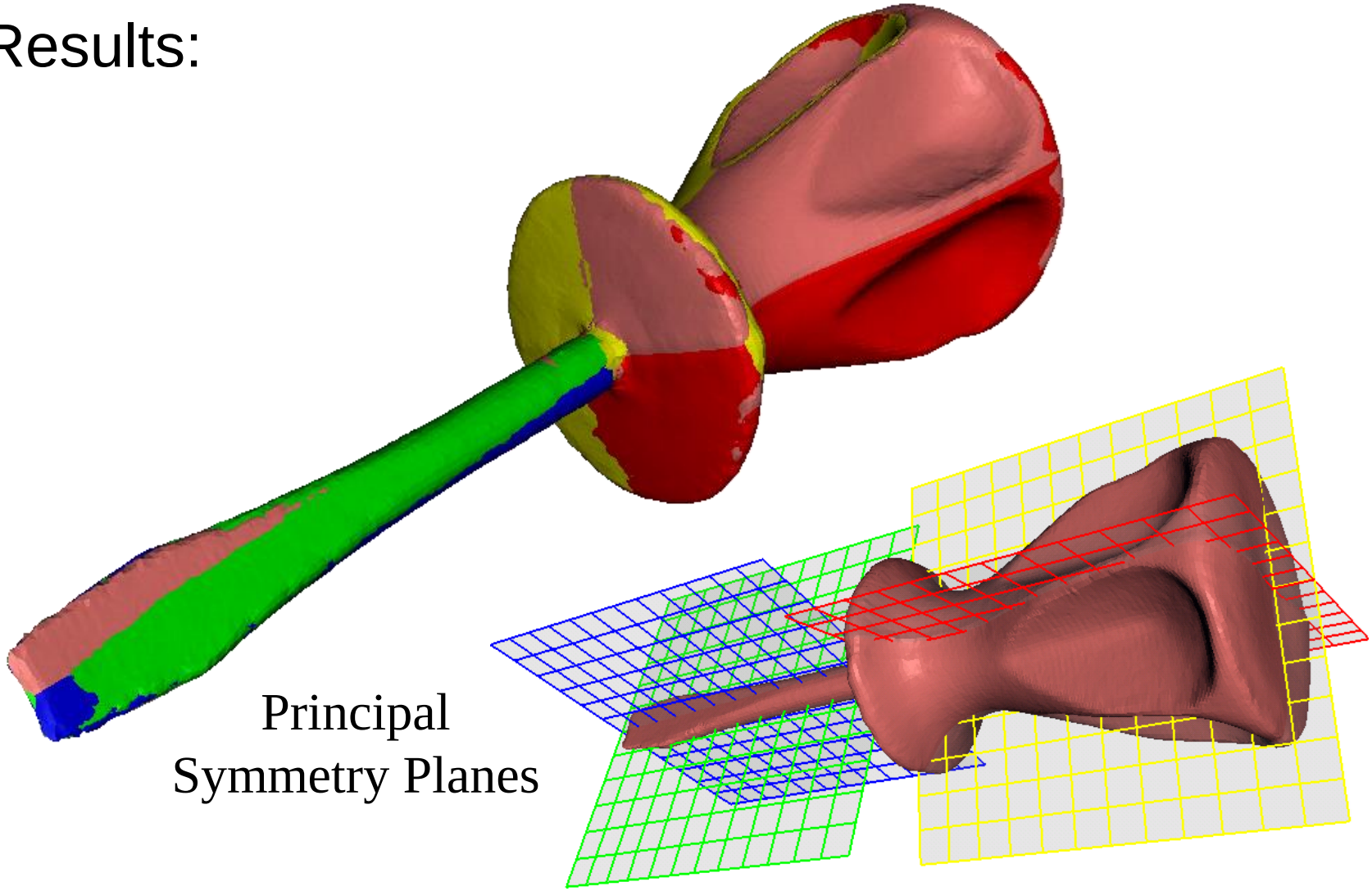
Application: Beautification

Results:



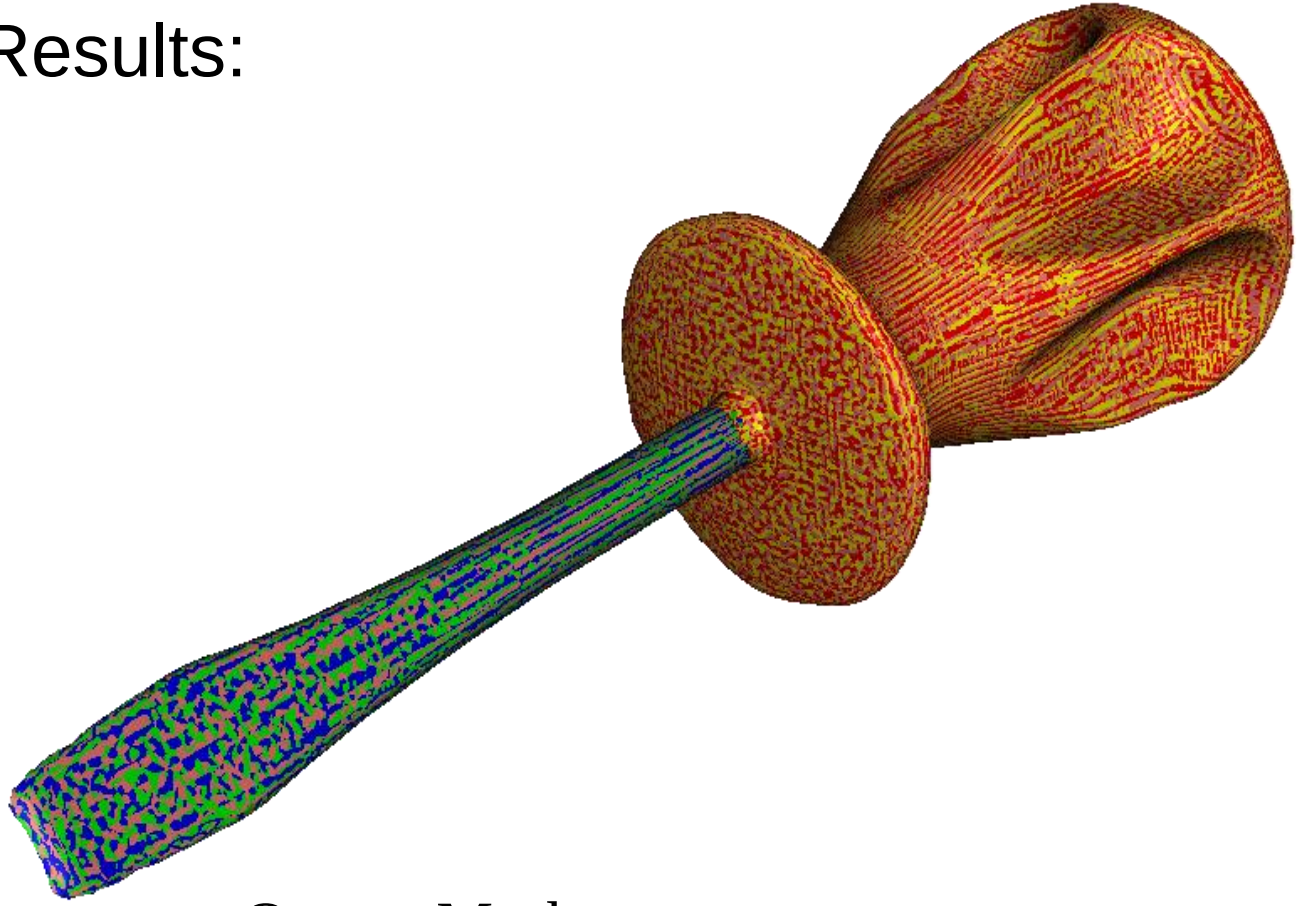
Application: Beautification

Results:



Application: Beautification

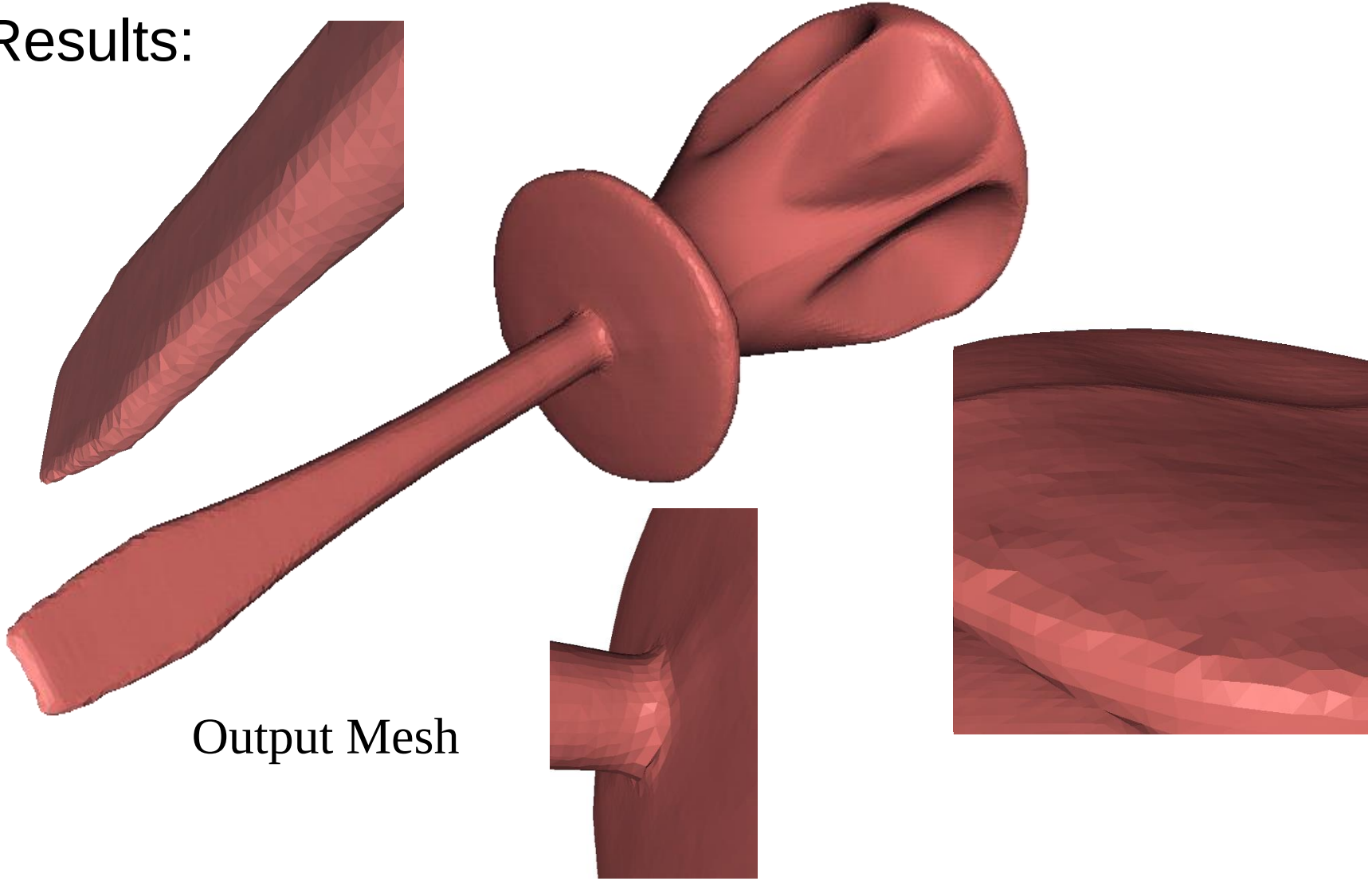
Results:



Output Mesh

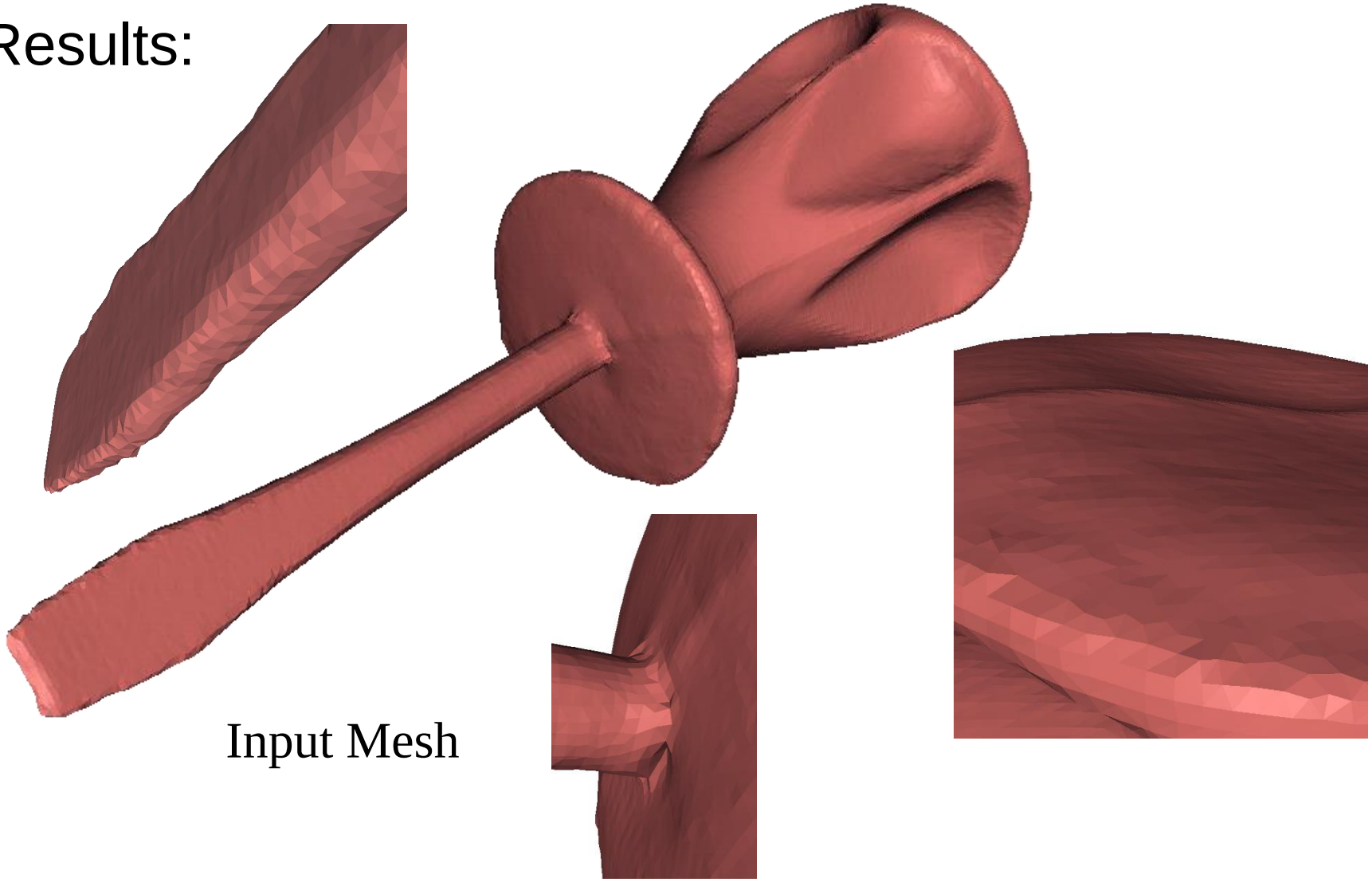
Application: Beautification

Results:



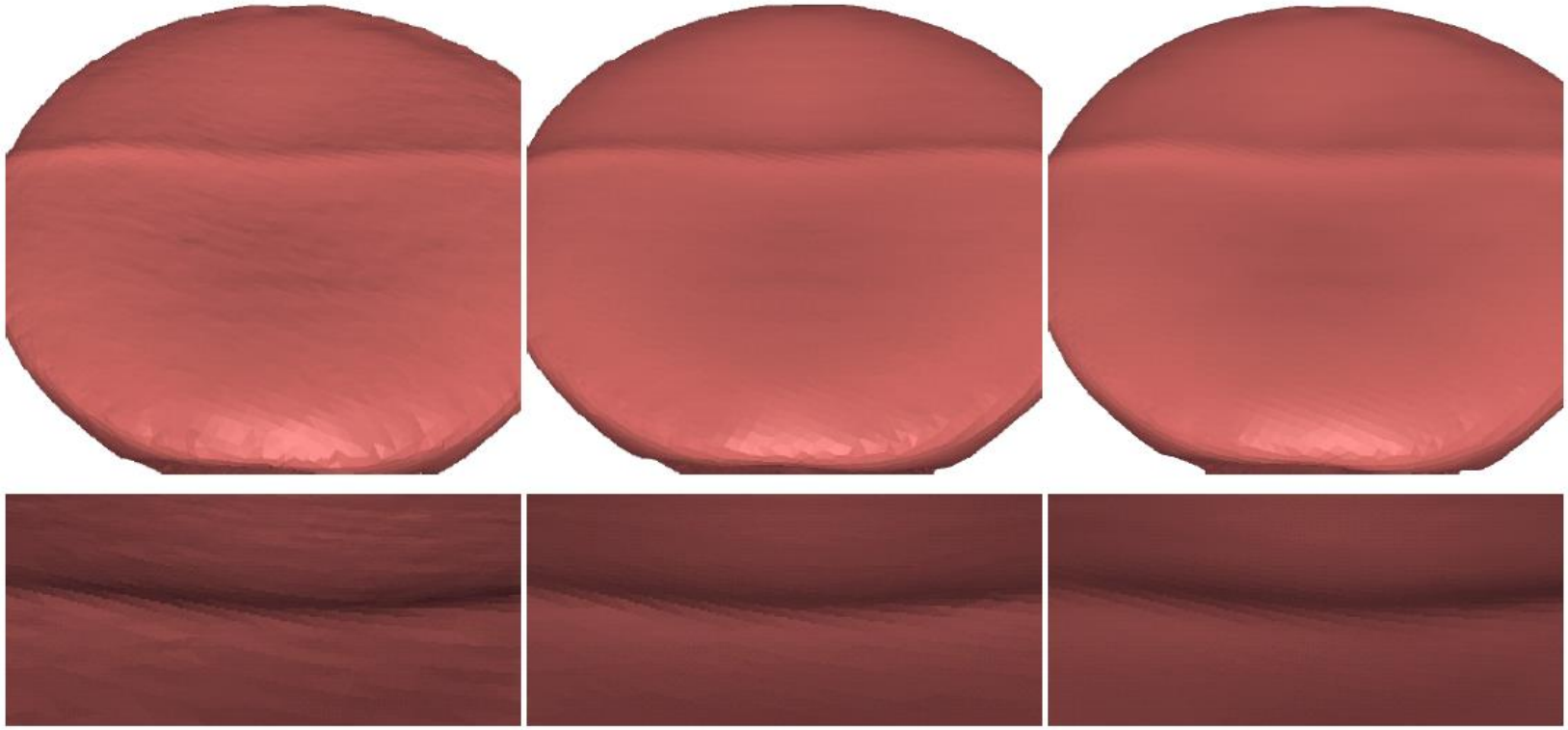
Application: Beautification

Results:



Application: Denoising

Results:



Input Mesh | Symmetrized Mesh | Bilateral Filtering

Applications

Alignment

Matching

Segmentation

Viewpoint selection

Simplification

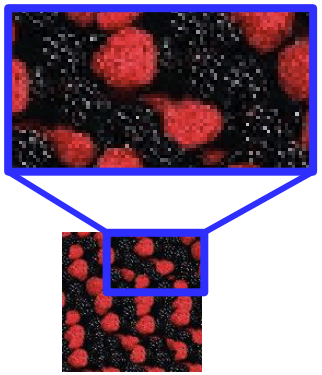
Beautification

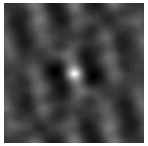
Texture synthesis ←

Application: Texture Synthesis

Maintain symmetries during texture synthesis

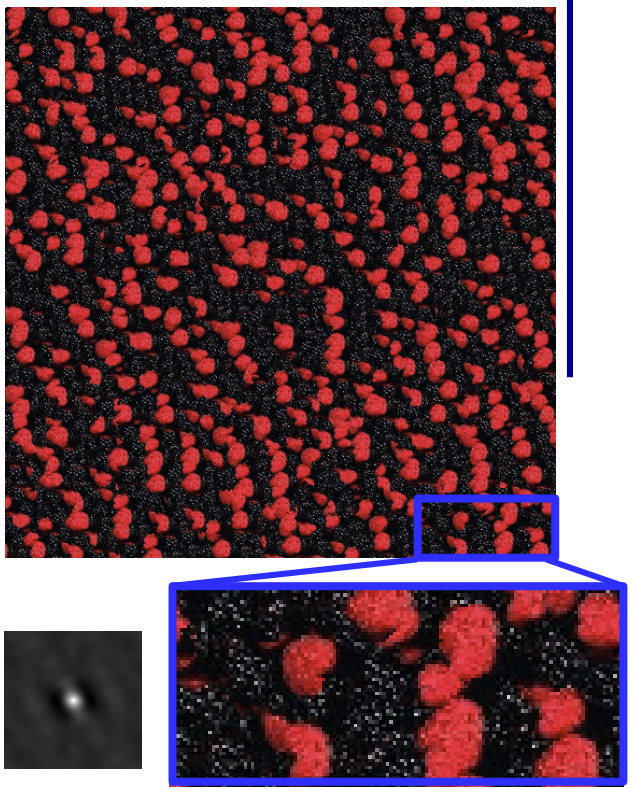
Input



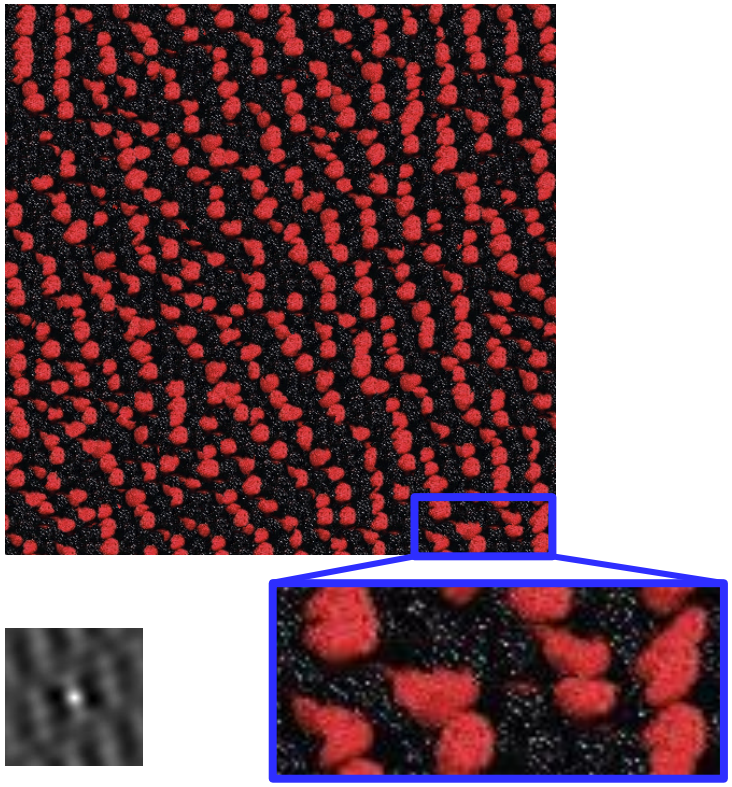


Symmetry Representation

Not Symmetry-guided

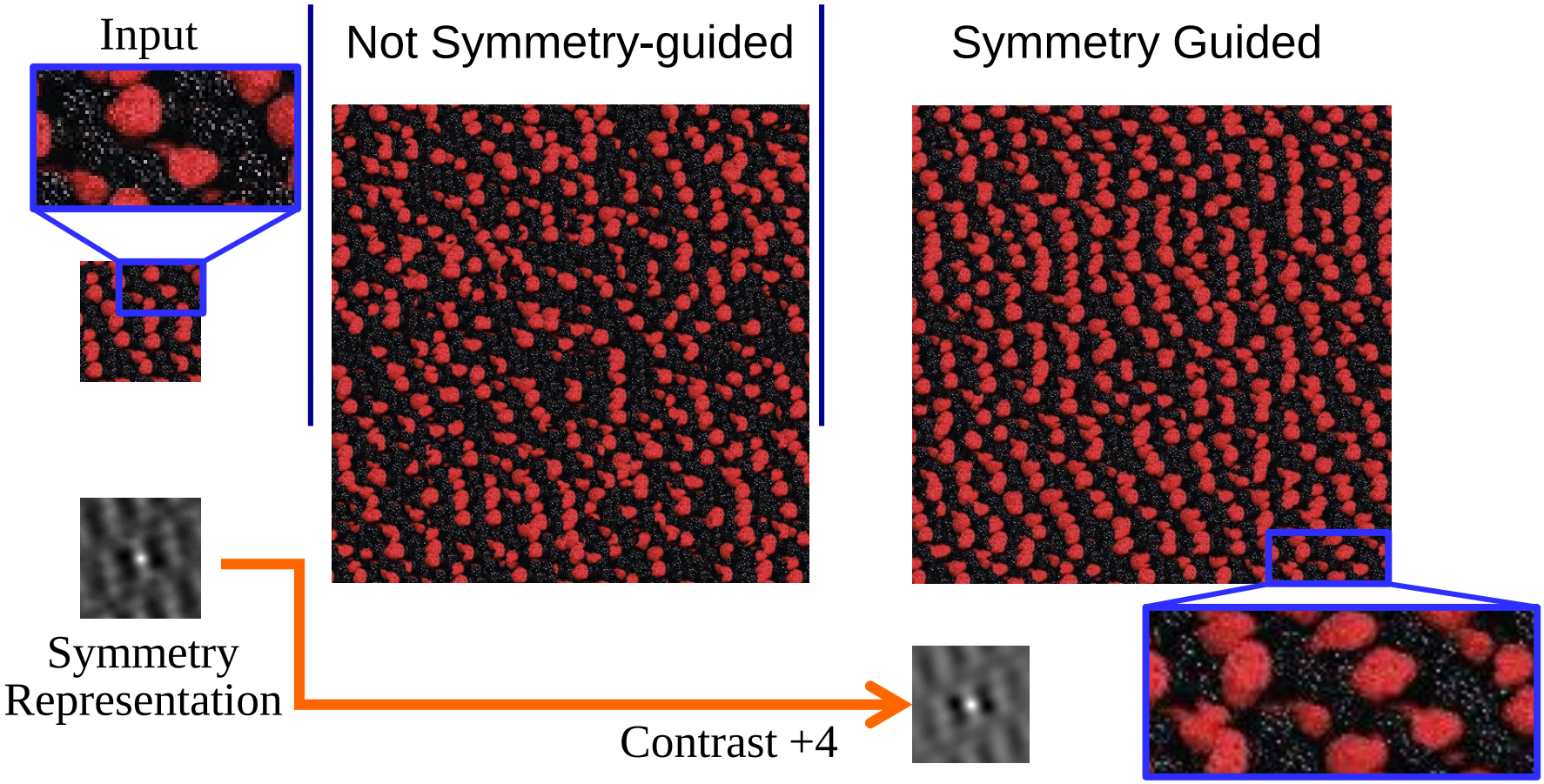


Symmetry Guided



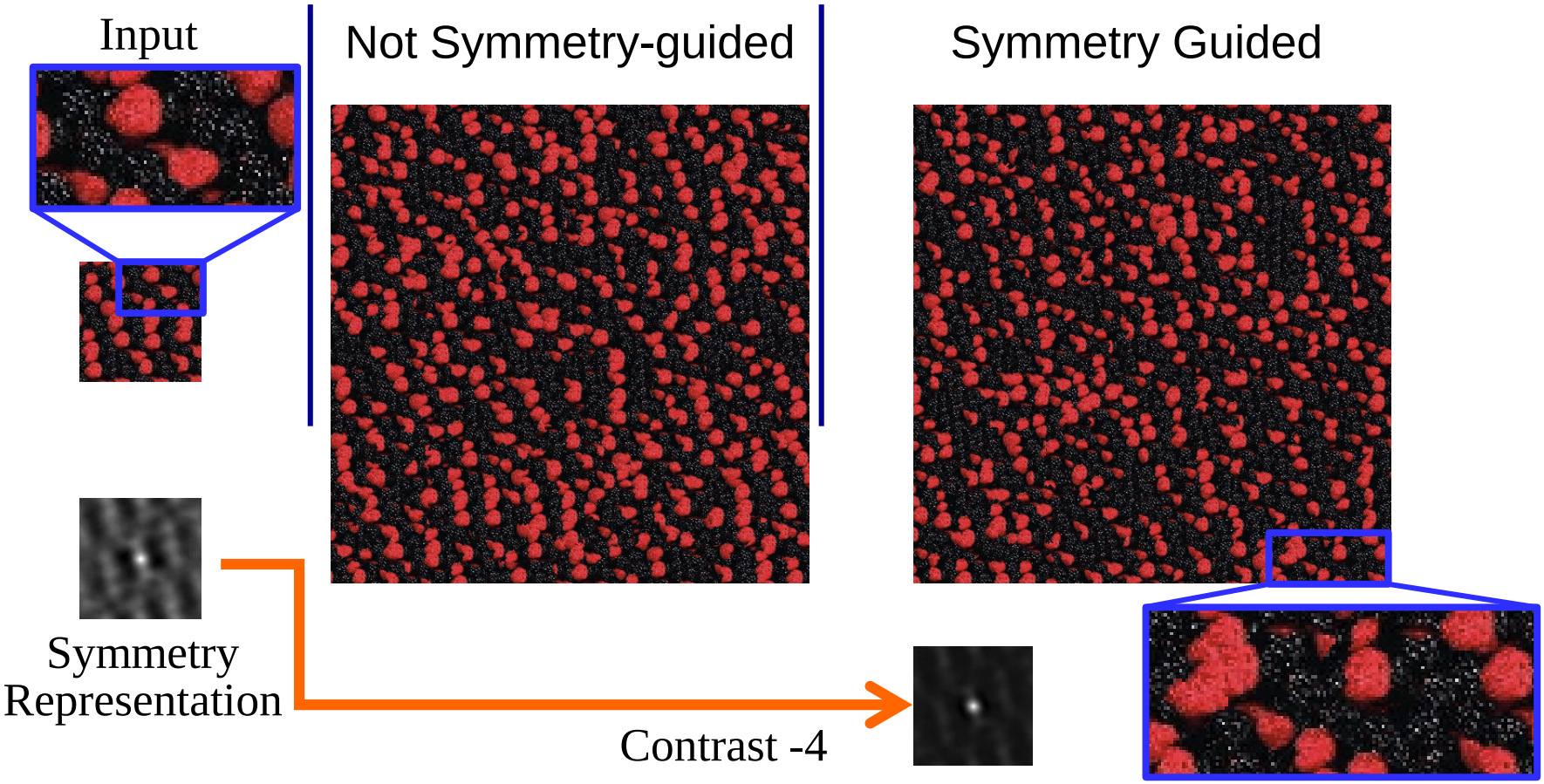
Application: Texture Manipulation

Edit symmetries during texture synthesis



Application: Texture Manipulation

Edit symmetries during texture synthesis



Summary

Representations

- Symmetry descriptor
- Symmetry transform
- Principal symmetries

Applications

- Alignment
- Matching
- Segmentation
- Viewpoint Selection
- Simplification
- Beautification
- Texture synthesis

Acknowledgements:

People

- Misha Kazhdan
- Josh Podolak
- Alex Golovinskiy
- Phil Shilane
- Chris DeCoro
- Syzmon Rusinkiewicz
- Vladimir Kim