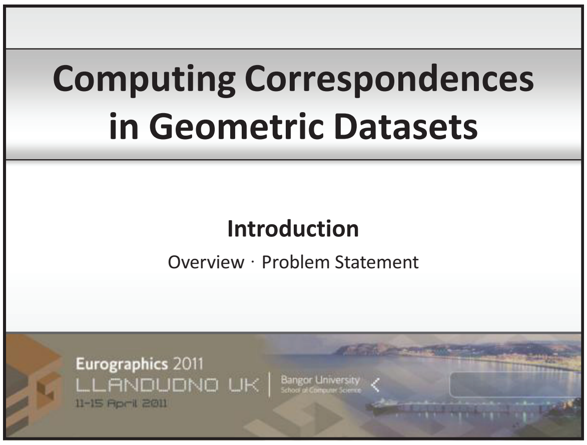
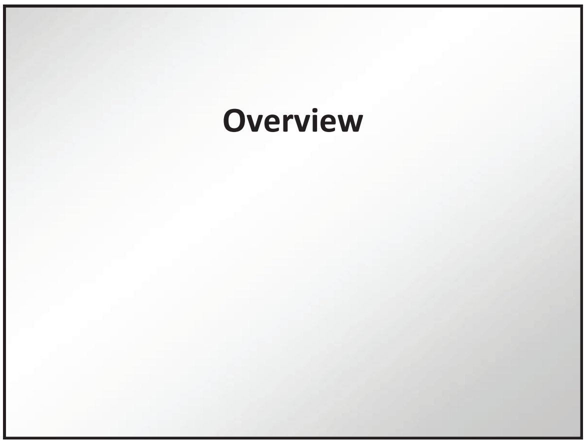


Computing Correspondences in Geometric Datasets

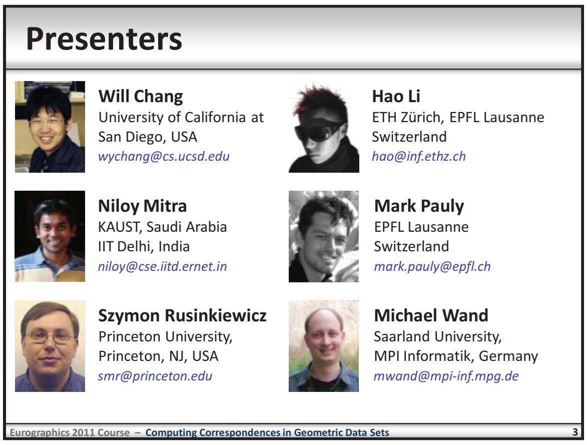
Introduction

Overview · Problem Statement

Eurographics 2011
LLANDUDNO UK
11-15 April 2011
Bangor University
School of Computer Science



Overview



Presenters

Will Chang
University of California at San Diego, USA
wychang@cs.ucsd.edu

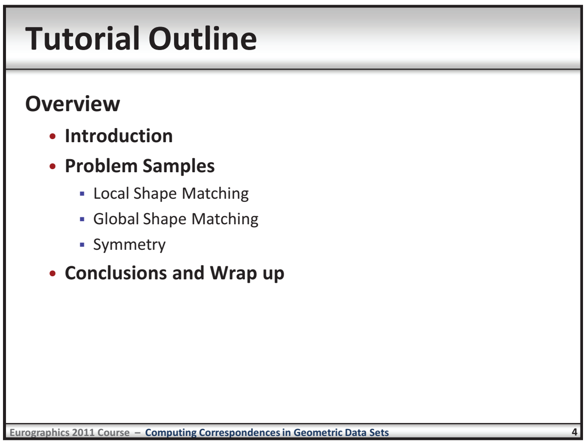
Hao Li
ETH Zürich, EPFL Lausanne
Switzerland
hao@inf.ethz.ch

Niloy Mitra
KAUST, Saudi Arabia
IIT Delhi, India
niloy@cse.iitd.ernet.in

Mark Pauly
EPFL Lausanne
Switzerland
mark.pauly@epfl.ch

Szymon Rusinkiewicz
Princeton University,
Princeton, NJ, USA
smr@princeton.edu

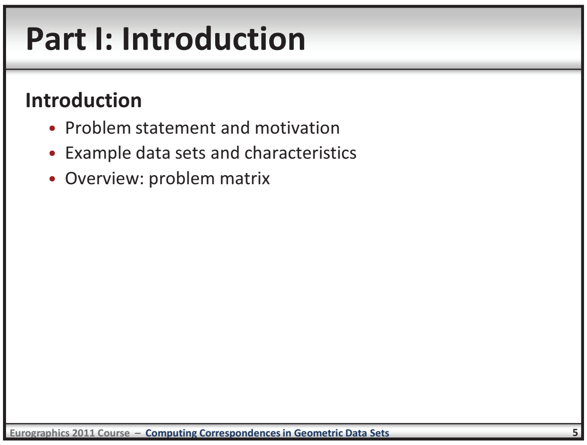
Michael Wand
Saarland University,
MPI Informatik, Germany
mwand@mpi-inf.mpg.de



Tutorial Outline

Overview

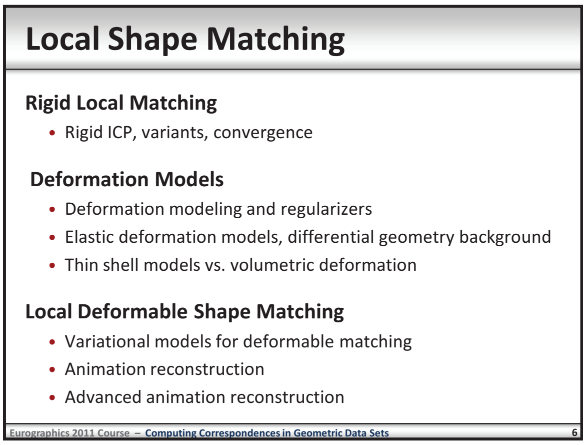
- **Introduction**
- **Problem Samples**
  - Local Shape Matching
  - Global Shape Matching
  - Symmetry
- **Conclusions and Wrap up**



Part I: Introduction

Introduction

- Problem statement and motivation
- Example data sets and characteristics
- Overview: problem matrix



Local Shape Matching

Rigid Local Matching

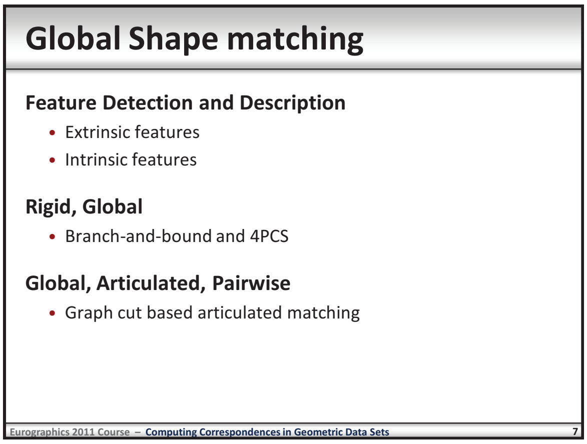
- Rigid ICP, variants, convergence

Deformation Models

- Deformation modeling and regularizers
- Elastic deformation models, differential geometry background
- Thin shell models vs. volumetric deformation

Local Deformable Shape Matching

- Variational models for deformable matching
- Animation reconstruction
- Advanced animation reconstruction



Global Shape matching

Feature Detection and Description

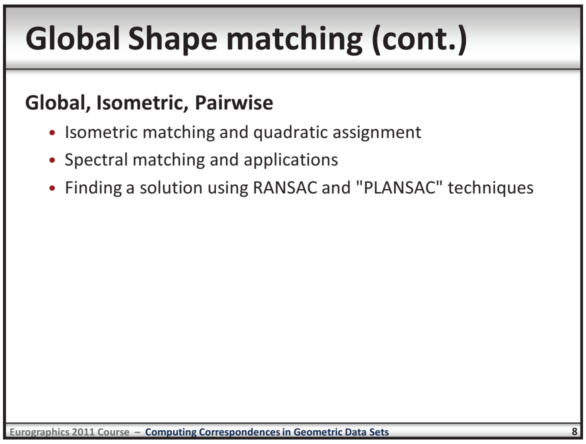
- Extrinsic features
- Intrinsic features

Rigid, Global

- Branch-and-bound and 4PCS

Global, Articulated, Pairwise

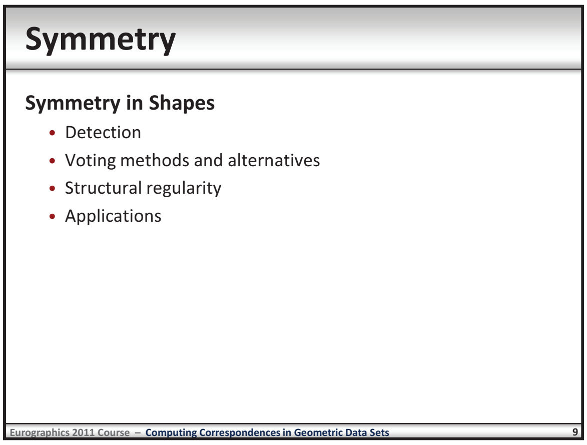
- Graph cut based articulated matching



Global Shape matching (cont.)

Global, Isometric, Pairwise

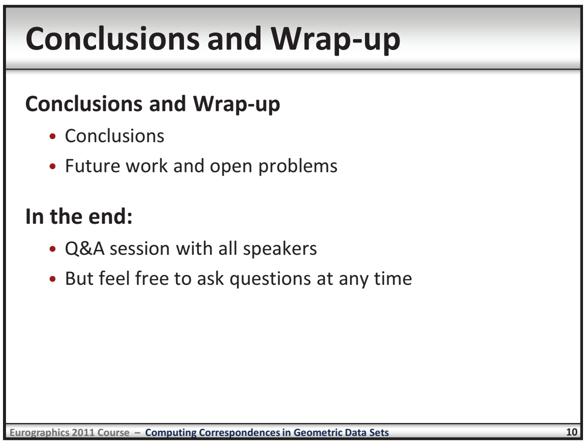
- Isometric matching and quadratic assignment
- Spectral matching and applications
- Finding a solution using RANSAC and "PLANSAC" techniques



Symmetry

Symmetry in Shapes

- Detection
- Voting methods and alternatives
- Structural regularity
- Applications



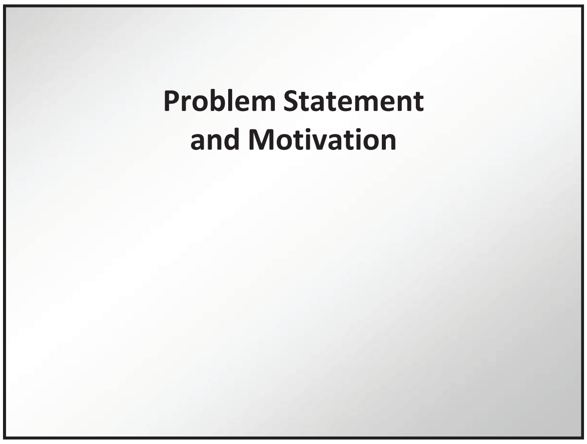
Conclusions and Wrap-up

Conclusions and Wrap-up

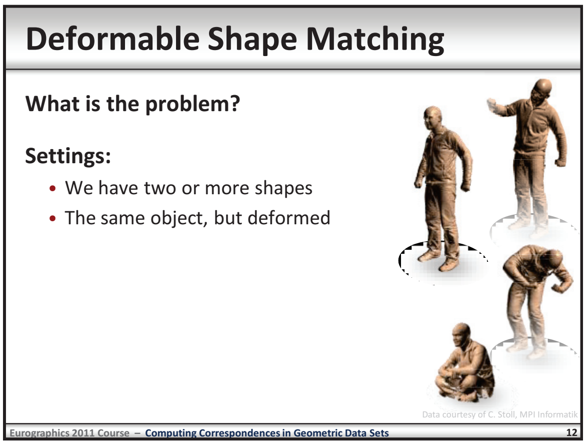
- Conclusions
- Future work and open problems

In the end:

- Q&A session with all speakers
- But feel free to ask questions at any time



Problem Statement and Motivation



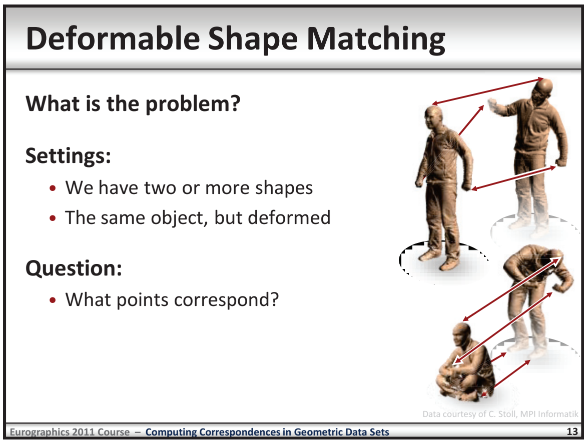
Deformable Shape Matching

What is the problem?

Settings:

- We have two or more shapes
- The same object, but deformed

Data courtesy of C. Stoll, MPI Informatik



Deformable Shape Matching

What is the problem?

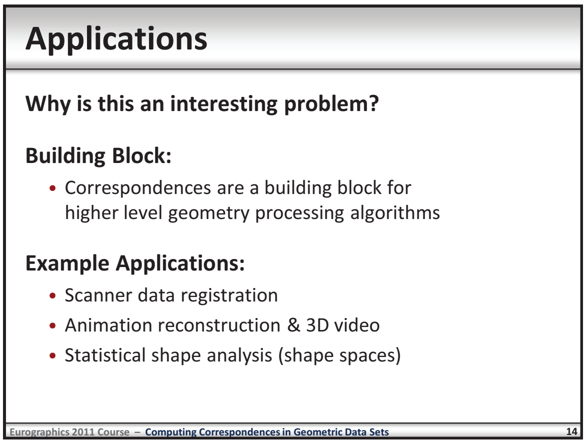
Settings:

- We have two or more shapes
- The same object, but deformed

Question:

- What points correspond?

Data courtesy of C. Stoll, MPI Informatik



Applications

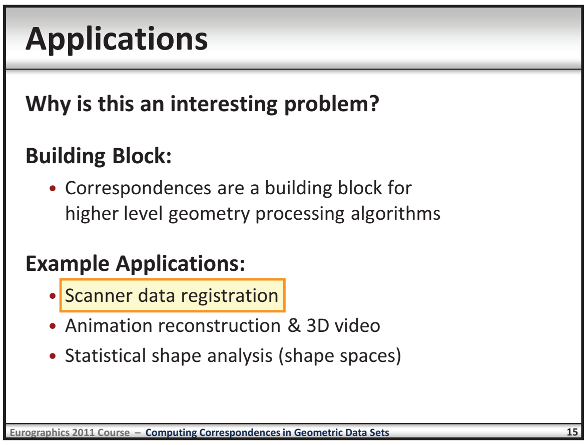
Why is this an interesting problem?

Building Block:

- Correspondences are a building block for higher level geometry processing algorithms

Example Applications:

- Scanner data registration
- Animation reconstruction & 3D video
- Statistical shape analysis (shape spaces)



Applications

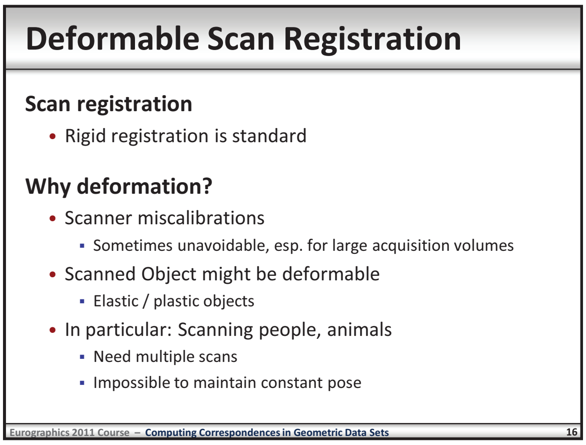
Why is this an interesting problem?

Building Block:

- Correspondences are a building block for higher level geometry processing algorithms

Example Applications:

- Scanner data registration
- Animation reconstruction & 3D video
- Statistical shape analysis (shape spaces)



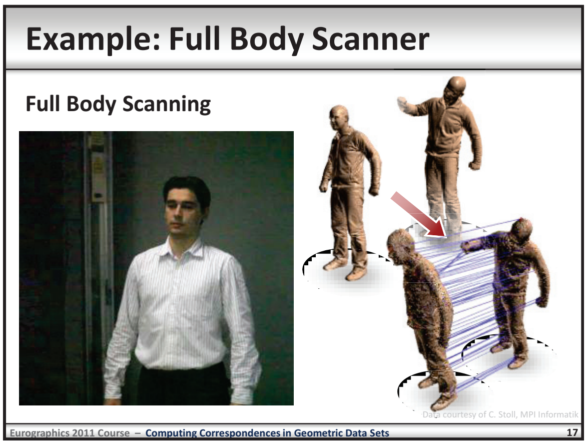
Deformable Scan Registration

Scan registration

- Rigid registration is standard

Why deformation?

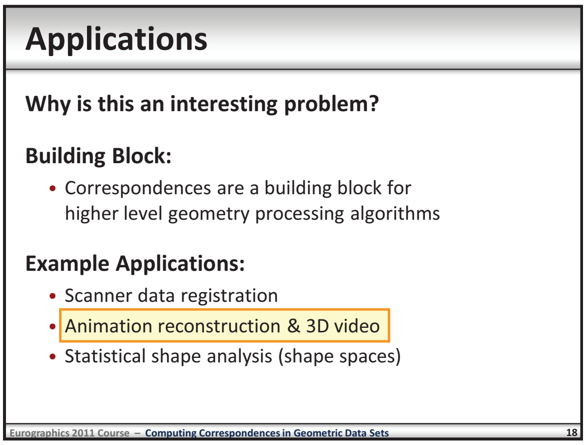
- Scanner miscalibrations
  - Sometimes unavoidable, esp. for large acquisition volumes
- Scanned Object might be deformable
  - Elastic / plastic objects
- In particular: Scanning people, animals
  - Need multiple scans
  - Impossible to maintain constant pose



Example: Full Body Scanner

Full Body Scanning

Data courtesy of C. Stoll, MPI Informatik



Applications

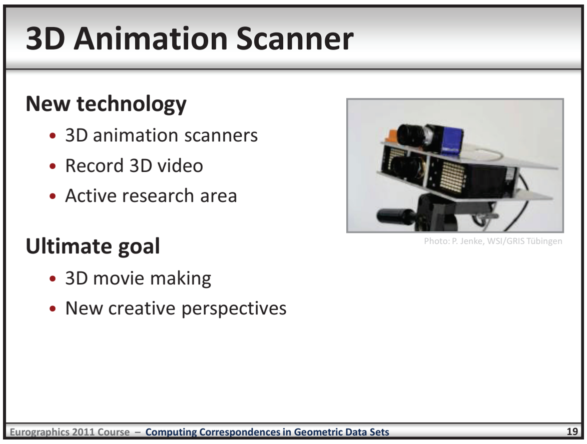
Why is this an interesting problem?

Building Block:

- Correspondences are a building block for higher level geometry processing algorithms

Example Applications:

- Scanner data registration
- Animation reconstruction & 3D video
- Statistical shape analysis (shape spaces)



3D Animation Scanner

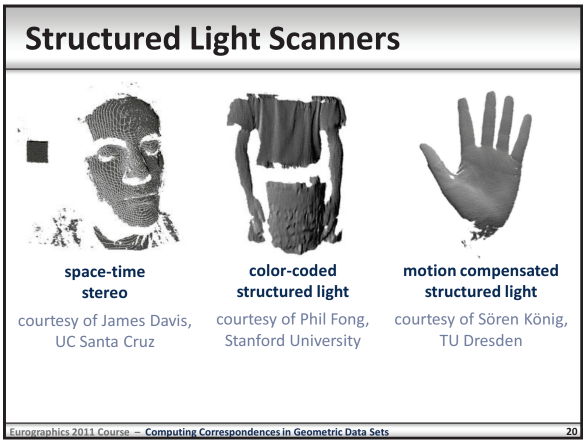
New technology

- 3D animation scanners
- Record 3D video
- Active research area

Photo: P. Jenke, WSI/GRIS Tübingen

Ultimate goal

- 3D movie making
- New creative perspectives



Structured Light Scanners

space-time stereo

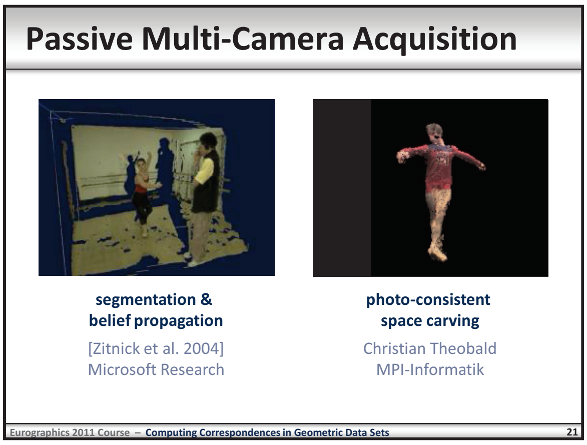
courtesy of James Davis, UC Santa Cruz

color-coded structured light

courtesy of Phil Fong, Stanford University

motion compensated structured light

courtesy of Sören König, TU Dresden



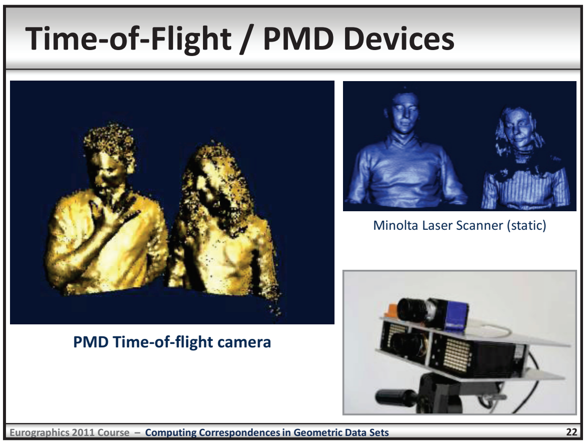
Passive Multi-Camera Acquisition

segmentation & belief propagation

[Zitnick et al. 2004]
Microsoft Research

photo-consistent space carving

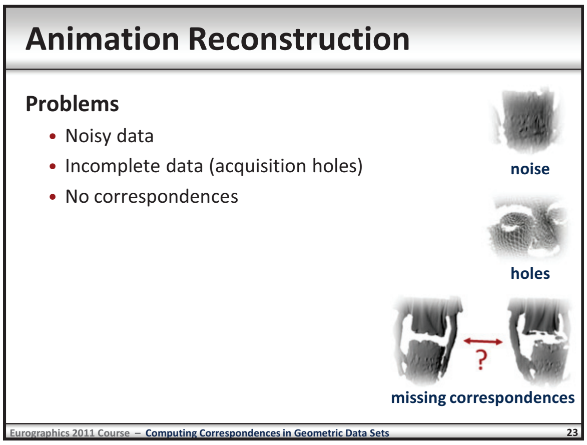
Christian Theobald
MPI-Informatik



Time-of-Flight / PMD Devices

Minolta Laser Scanner (static)

PMD Time-of-flight camera

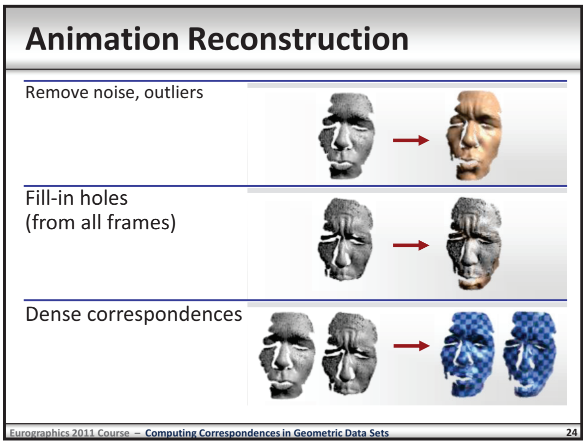


Animation Reconstruction

Problems

- Noisy data
- Incomplete data (acquisition holes)
- No correspondences

noise
holes
?
missing correspondences

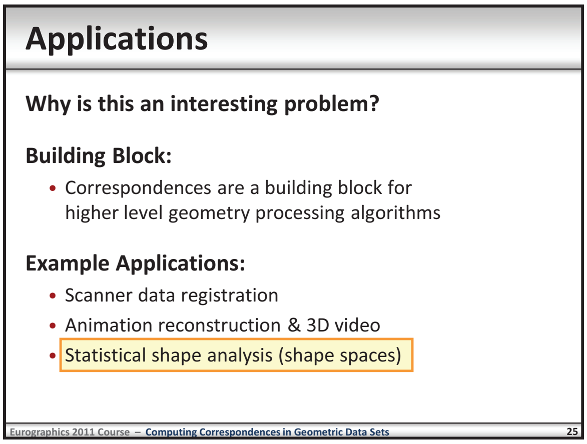


Animation Reconstruction

Remove noise, outliers

Fill-in holes
(from all frames)

Dense correspondences



Applications

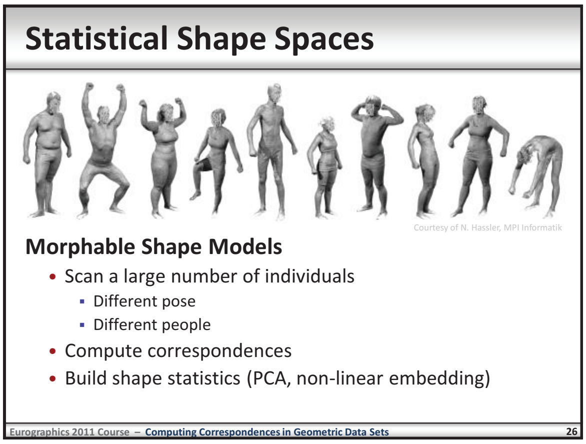
Why is this an interesting problem?

Building Block:

- Correspondences are a building block for higher level geometry processing algorithms

Example Applications:

- Scanner data registration
- Animation reconstruction & 3D video
- Statistical shape analysis (shape spaces)

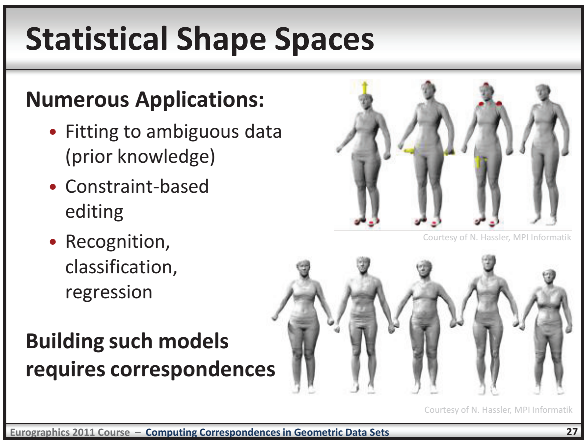


Statistical Shape Spaces

Courtesy of N. Hassler, MPI Informatik

Morphable Shape Models

- Scan a large number of individuals
  - Different pose
  - Different people
- Compute correspondences
- Build shape statistics (PCA, non-linear embedding)



Statistical Shape Spaces

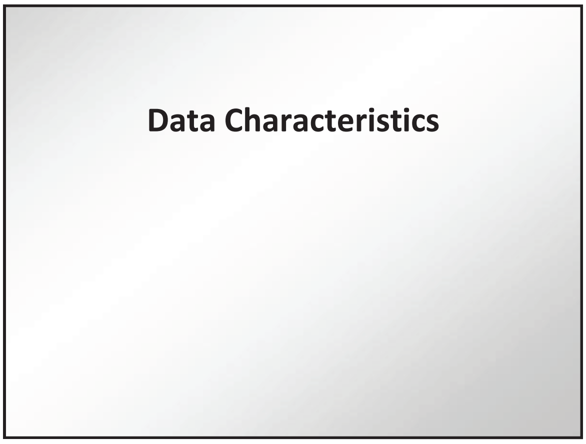
Numerous Applications:

- Fitting to ambiguous data (prior knowledge)
- Constraint-based editing
- Recognition, classification, regression

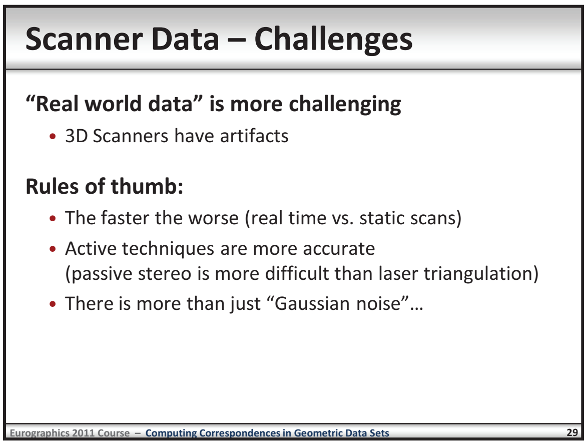
Courtesy of N. Hassler, MPI Informatik

Building such models requires correspondences

Courtesy of N. Hassler, MPI Informatik



Data Characteristics



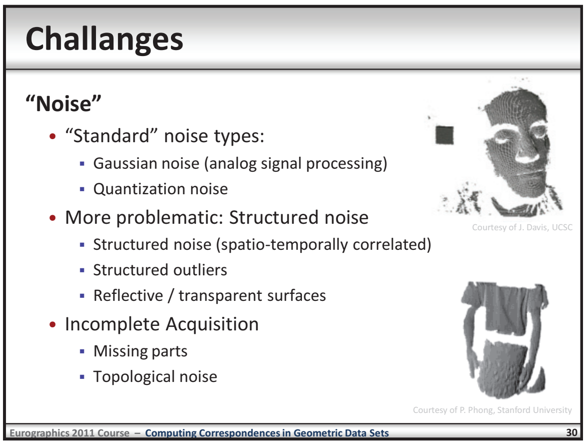
Scanner Data – Challenges

“Real world data” is more challenging

- 3D Scanners have artifacts

Rules of thumb:

- The faster the worse (real time vs. static scans)
- Active techniques are more accurate (passive stereo is more difficult than laser triangulation)
- There is more than just “Gaussian noise”…



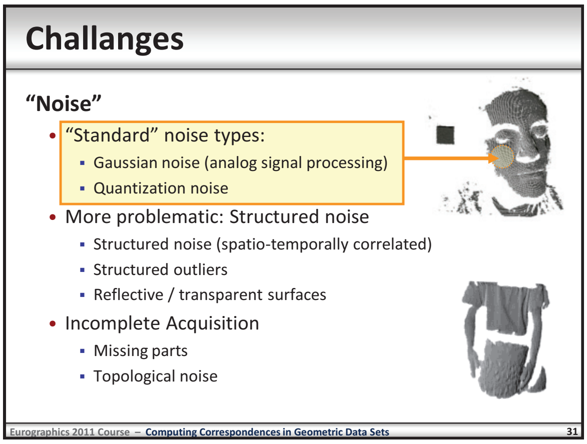
Challanges

“Noise”

- “Standard” noise types:
  - Gaussian noise (analog signal processing)
  - Quantization noise
- More problematic: Structured noise
  - Structured noise (spatio-temporally correlated)
  - Structured outliers
  - Reflective / transparent surfaces
- Incomplete Acquisition
  - Missing parts
  - Topological noise

Courtesy of J. Davis, UCSC

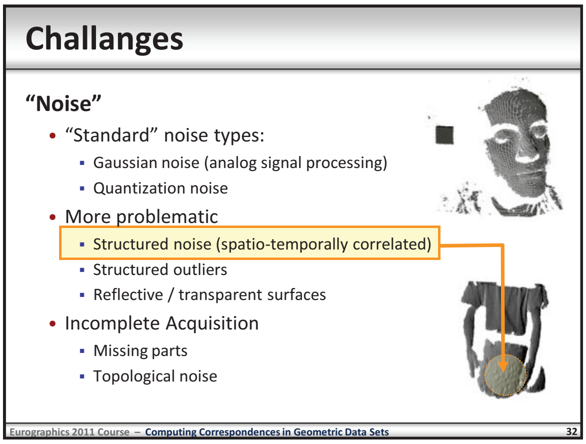
Courtesy of P. Phong, Stanford University



Challanges

“Noise”

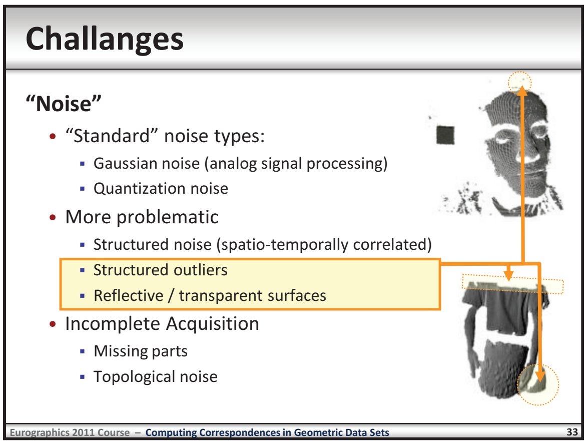
- “Standard” noise types:
  - Gaussian noise (analog signal processing)
  - Quantization noise
- More problematic: Structured noise
  - Structured noise (spatio-temporally correlated)
  - Structured outliers
  - Reflective / transparent surfaces
- Incomplete Acquisition
  - Missing parts
  - Topological noise



Challanges

“Noise”

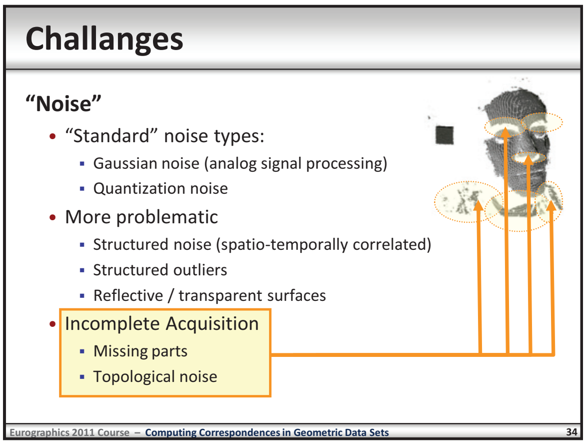
- “Standard” noise types:
  - Gaussian noise (analog signal processing)
  - Quantization noise
- More problematic
  - Structured noise (spatio-temporally correlated)
  - Structured outliers
  - Reflective / transparent surfaces
- Incomplete Acquisition
  - Missing parts
  - Topological noise



Challanges

“Noise”

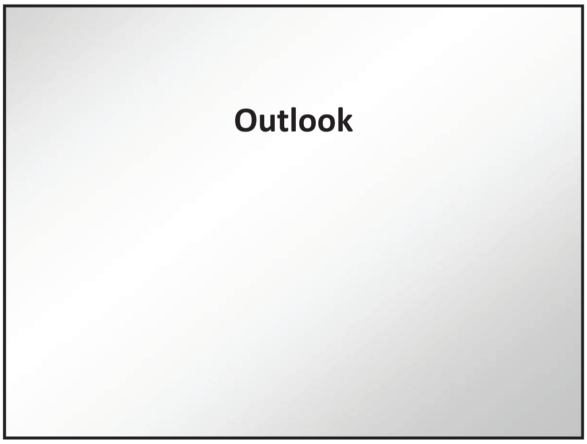
- “Standard” noise types:
  - Gaussian noise (analog signal processing)
  - Quantization noise
- More problematic
  - Structured noise (spatio-temporally correlated)
  - Structured outliers
  - Reflective / transparent surfaces
- Incomplete Acquisition
  - Missing parts
  - Topological noise



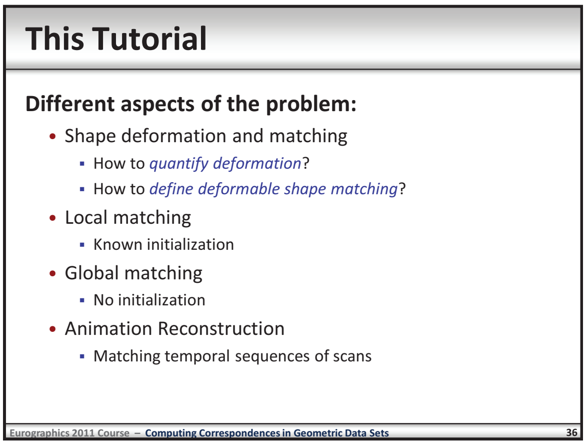
Challanges

“Noise”

- “Standard” noise types:
  - Gaussian noise (analog signal processing)
  - Quantization noise
- More problematic
  - Structured noise (spatio-temporally correlated)
  - Structured outliers
  - Reflective / transparent surfaces
- Incomplete Acquisition
  - Missing parts
  - Topological noise



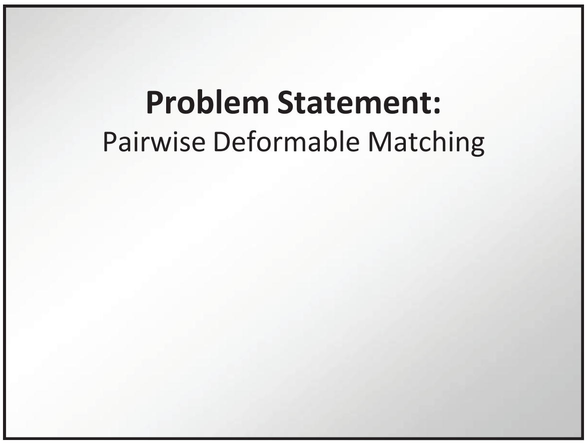
Outlook



This Tutorial

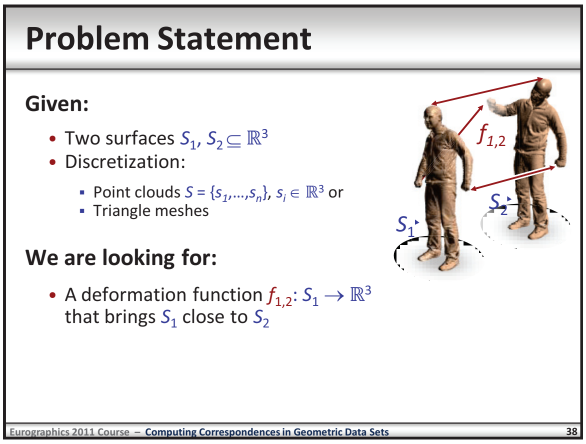
Different aspects of the problem:

- Shape deformation and matching
  - How to *quantify deformation*?
  - How to *define deformable shape matching*?
- Local matching
  - Known initialization
- Global matching
  - No initialization
- Animation Reconstruction
  - Matching temporal sequences of scans



Problem Statement:

Pairwise Deformable Matching



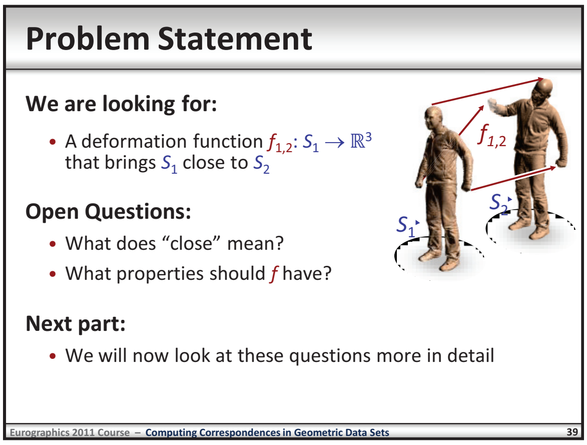
Problem Statement

Given:

- Two surfaces $S_1, S_2 \subseteq \mathbb{R}^3$
- Discretization:
  - Point clouds $S = \{s_1,\ldots,s_n\}$, $s_i \in \mathbb{R}^3$ or
  - Triangle meshes

We are looking for:

- A deformation function $f_{1,2}: S_1 \rightarrow \mathbb{R}^3$ that brings S_1 close to S_2



Problem Statement

We are looking for:

- A deformation function $f_{1,2}: S_1 \rightarrow \mathbb{R}^3$ that brings S_1 close to S_2

Open Questions:

- What does “close” mean?
- What properties should f have?

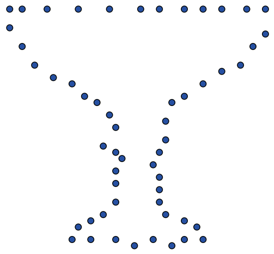
Next part:

- We will now look at these questions more in detail

Computing Correspondences in Geometric Datasets

ICP + Tangent Space optimization for Rigid Motions

Notations

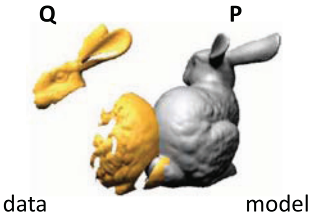


$$P = \{p_i\}$$

Registration Problem

Given

Two point cloud data sets **P** (*model*) and **Q** (*data*) sampled from surfaces Φ_P and Φ_Q respectively.



Assume Φ_Q is a part of Φ_P.

Registration with known Correspondence

$$\{p_i\} \text{ and } \{q_i\} \text{ such that } p_i \rightarrow q_i$$

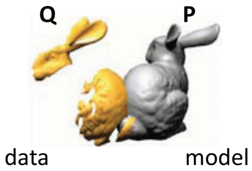
Registration Problem

Given

Two point cloud data sets **P** and **Q**.

Goal

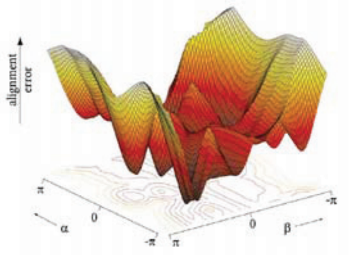
Register **Q** against **P** by minimizing the squared distance between the underlying surfaces using only *rigid transforms.*



Registration with known Correspondence

$$\{p_i\} \text{ and } \{q_i\} \text{ such that } p_i \rightarrow q_i$$

$$p_i \rightarrow Rp_i + t \quad \Rightarrow \min_{R,t} \sum_i \|Rp_i + t - q_i\|^2$$

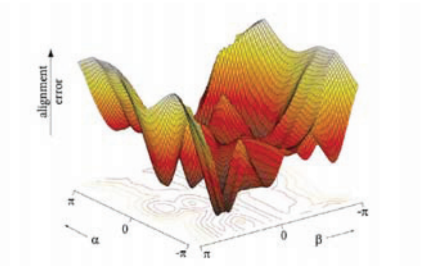


R obtained using SVD of covariance matrix.

Registration with known Correspondence

$$\{p_i\} \text{ and } \{q_i\} \text{ such that } p_i \rightarrow q_i$$

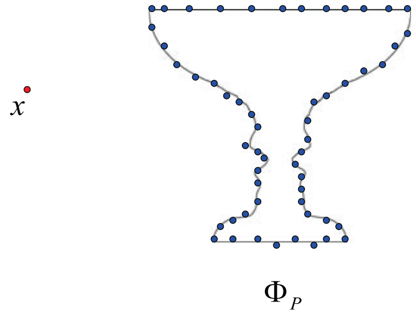
$$p_i \rightarrow Rp_i + t \;\; \Rightarrow \min_{R,t} \sum_{\mathrm{i}} \| Rp_i + t - q_i \|^2$$



R obtained using SVD of covariance matrix.

$$t = \overline{\mathrm{q}} - R\overline{p}$$

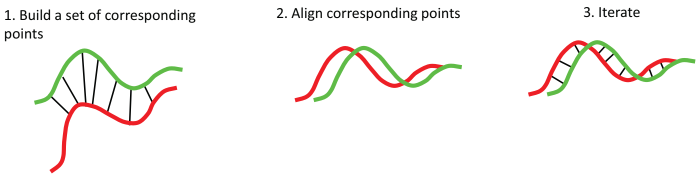
Squared Distance Function (F)



ICP (Iterated Closest Point)

Iterative minimization algorithms (ICP)

[Besl 92, Chen 92]



1. Build a set of corresponding points

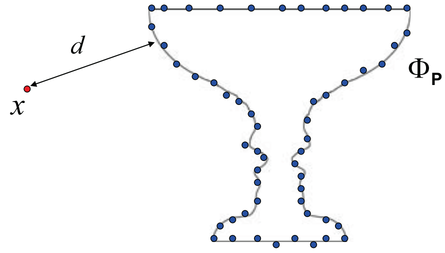
2. Align corresponding points

3. Iterate

Properties

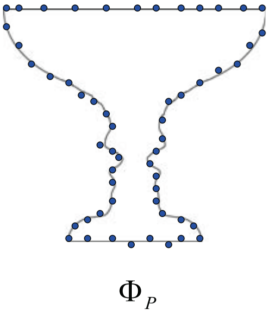
- Dense correspondence sets
- Converges if starting positions are "close"

Squared Distance Function (F)



$$F(x, \Phi_P) = d^2$$

No (explicit) Correspondence



Registration Problem

Rigid transform α that takes points $q_i \rightarrow \alpha(q_i)$

Our goal is to solve for,

$$\min_{\alpha} \sum_{q_i \in Q} F(\alpha(q_i), \Phi_P)$$

An optimization problem in the squared distance field of **P**, the model PCD.

Registration Problem

$$\alpha = \text{rotation}\ (R) + \text{translation}(t)$$

Our goal is to solve for,

$$\min_{R,t} \sum_{q_i \in Q} F(Rq_i + t, \Phi_P)$$

Optimize for **R** and **t**.

ICP in Our Framework

- Point-to-point ICP (good for large *d*)

$$\mathsf{F}(\mathbf{x}, \Phi_{\mathbf{P}}) = (\mathbf{x} - \mathbf{p})^2 \quad \Rightarrow \quad \delta_{\mathrm{j}} = 1$$

- Point-to-plane ICP (good for small *d*)

$$\mathsf{F}(\mathbf{x}, \Phi_{\mathbf{P}}) = (\vec{\mathsf{n}} \cdot (\mathbf{x} - \mathbf{p}))^2 \quad \Rightarrow \quad \delta_{\mathrm{j}} = 0$$

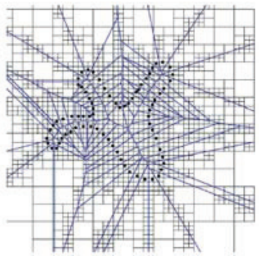
Registration in 2D

- Minimize residual error $\varepsilon(\theta, \mathsf{t}_\mathsf{x}, \mathsf{t}_\mathsf{y})$

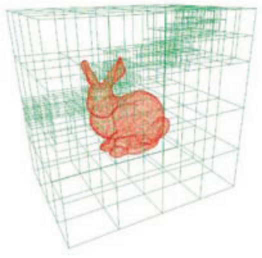
$$\begin{bmatrix} \\ \mathbf{M}_1 \\ \\ \end{bmatrix} \begin{bmatrix} \theta \\ \mathsf{t}_\mathsf{x} \\ \mathsf{t}_\mathsf{y} \end{bmatrix} = \begin{bmatrix} \\ \mathbf{M}_2 \\ \\ \end{bmatrix}$$

depends on **F**⁺ data PCD (**Q**).

Example d2trees



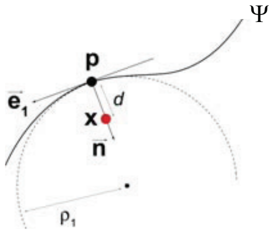
2D



3D

Approximate Squared Distance

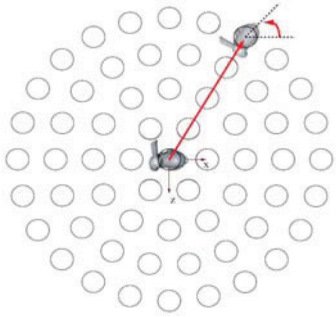
For a curve Ψ,



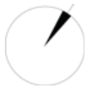
$$\mathbf{F}(\mathbf{x}, \Psi) = \frac{d}{d\text{-}\rho_1} \mathsf{x}_1^2 + \mathsf{x}_2^2 = \delta_1 \mathsf{x}_1^2 + \mathsf{x}_2^2$$

[Pottmann and Hofer 2003]

Convergence Funnel



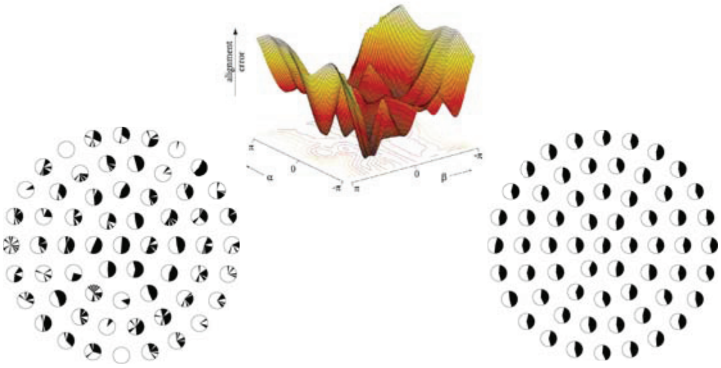
Translation in x-z plane.
Rotation about y-axis.



■ Converges

□ Does not converge

Convergence Funnel



Plane-to-plane ICP

distance-field formulation

Descriptors

$$P = \{p_i\}$$

- closest point → based on Euclidean distance

Descriptors

$$P = \{p_i\}$$

- closest point → based on Euclidean distance

$$P = \{p_i, a_i, b_i, \ldots\}$$

- closest point → based on Euclidean distance between point + descriptors (attributes)

(Invariant) Descriptors

$$P = \{p_i\}$$

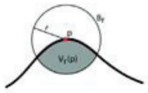
- closest point → based on Euclidean distance

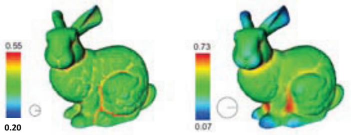
$$P = \{p_i, a_i, b_i, \ldots\}$$

- closest point → based on Euclidean distance between point + descriptors (attributes)

Integral Volume Descriptor

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





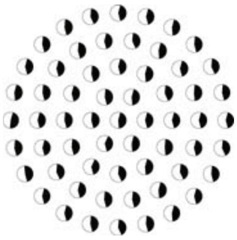
Relation to mean curvature

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

When Objects are *Poorly* Aligned

- **Use descriptors for global registrations**

global alignment → refinement with local (e.g., ICP)



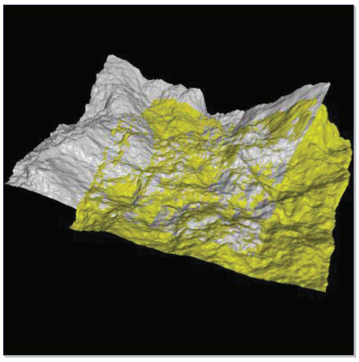
Computing Correspondences in Geometric Datasets

Local, Rigid, Pairwise

The ICP algorithm and its extensions

Pairwise Rigid Registration Goal

Align two partially-overlapping meshes given initial guess for relative transform



Outline

ICP: Iterative Closest Points

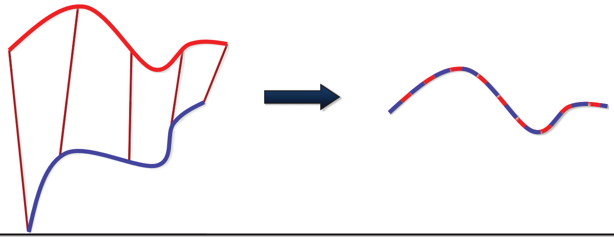
Classification of ICP variants

- Faster alignment
- Better robustness

ICP as function minimization

Aligning 3D Data

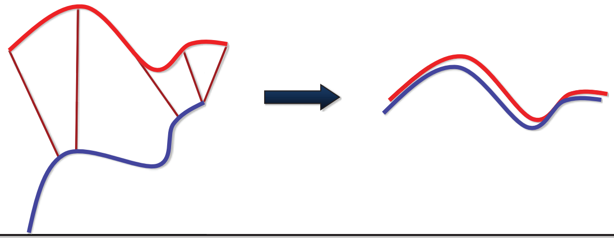
If correct correspondences are known, can find correct relative rotation/translation



Aligning 3D Data

How to find correspondences: User input? Feature detection? Signatures?

Alternative: assume closest points correspond

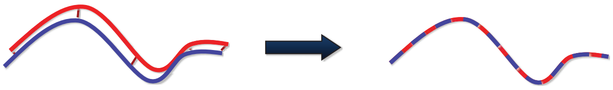


Aligning 3D Data

… and iterate to find alignment

- Iterative Closest Points (ICP) [Besl & McKay 92]

Converges if starting position "close enough"



Basic ICP

Select e.g. 1000 random points

Match each to closest point on other scan, using data structure such as *k*-d tree

Reject pairs with distance > k times median

Construct error function:

$$E = \sum \left| Rp_i + t - q_i \right|^2$$

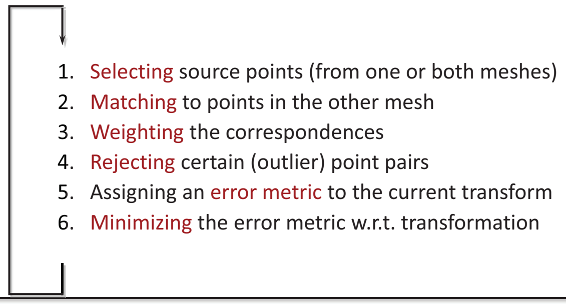
Minimize (closed form solution in [Horn 87])

ICP Variants

1. Selecting source points (from one or both meshes)
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. transformation

ICP Variants

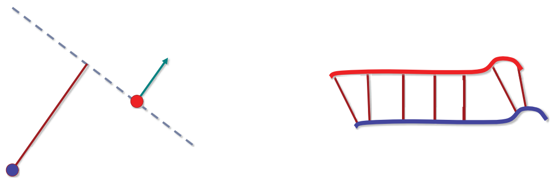
Variants on the following stages of ICP have been proposed:



1. Selecting source points (from one or both meshes)
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. transformation

Point-to-Plane Error Metric

Using point-to-plane distance instead of point-to-point lets flat regions slide along each other [Chen & Medioni 91]



Performance of Variants

Can analyze various aspects of performance:

- Speed
- Stability
- Tolerance of noise and/or outliers
- Maximum initial misalignment

Comparisons of many variants in [Rusinkiewicz & Levoy, 3DIM 2001]

Point-to-Plane Error Metric

Error function:

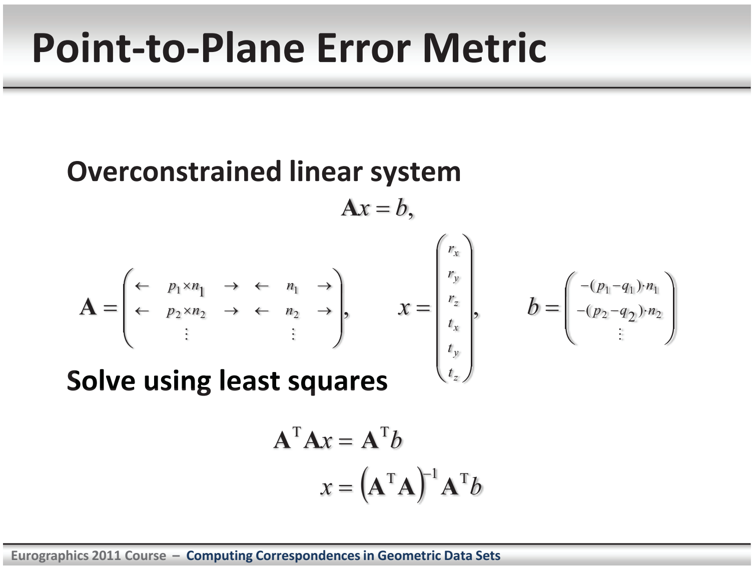
$$E = \sum \left((Rp_i + t - q_i) \cdot n_i \right)^2$$

where R is a rotation matrix, t is translation vector

Linearize (i.e. assume that sin $\theta \approx \theta$, cos $\theta \approx 1$):

$$E \approx \sum \left((p_i - q_i) \cdot n_i + r \cdot (p_i \times n_i) + t \cdot n_i \right)^2, \qquad \text{where } r = \begin{pmatrix} r_x \\ r_y \\ r_z \end{pmatrix}$$

Result: overconstrained linear system



Point-to-Plane Error Metric

Overconstrained linear system

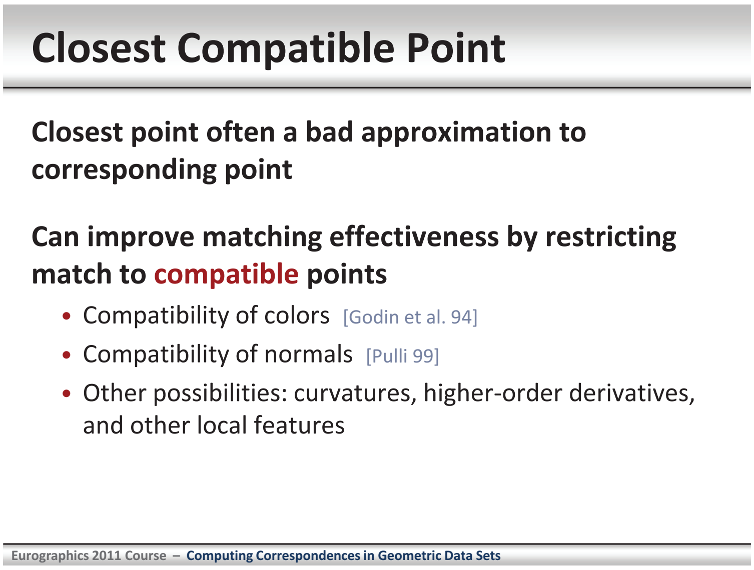
$$\mathbf{A}x = b,$$

$$\mathbf{A} = \begin{pmatrix} \leftarrow & p_1 \times n_1 & \rightarrow & \leftarrow & n_1 & \rightarrow \\ \leftarrow & p_2 \times n_2 & \rightarrow & \leftarrow & n_2 & \rightarrow \\ & \vdots & & & \vdots & \end{pmatrix}, \qquad x = \begin{pmatrix} r_x \\ r_y \\ r_z \\ t_x \\ t_y \\ t_z \end{pmatrix}, \qquad b = \begin{pmatrix} -(p_1 - q_1) \cdot n_1 \\ -(p_2 - q_2) \cdot n_2 \\ \vdots \end{pmatrix}$$

Solve using least squares

$$\mathbf{A}^{\mathrm{T}}\mathbf{A}x = \mathbf{A}^{\mathrm{T}}b$$

$$x = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{T}}b$$

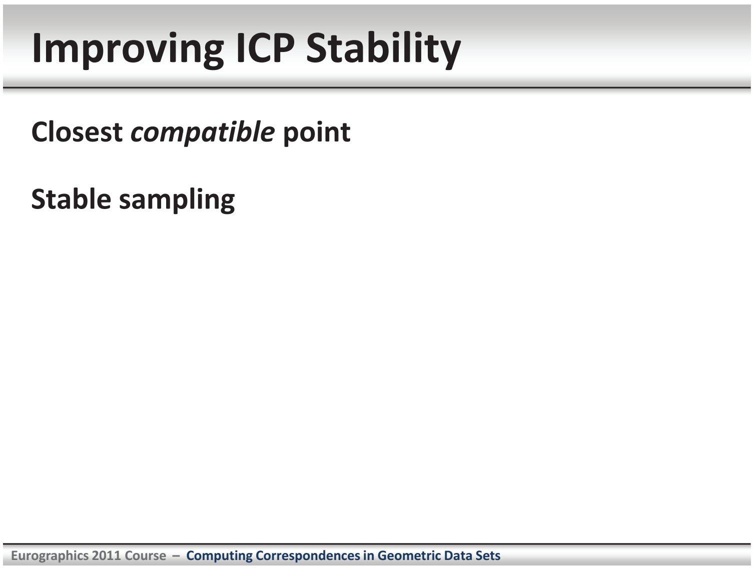


Closest Compatible Point

Closest point often a bad approximation to corresponding point

Can improve matching effectiveness by restricting match to compatible points

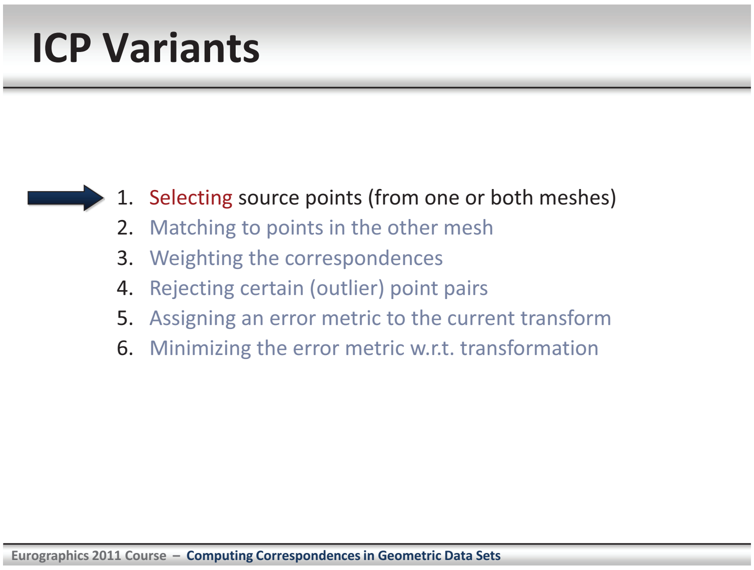
- Compatibility of colors [Godin et al. 94]
- Compatibility of normals [Pulli 99]
- Other possibilities: curvatures, higher-order derivatives, and other local features



Improving ICP Stability

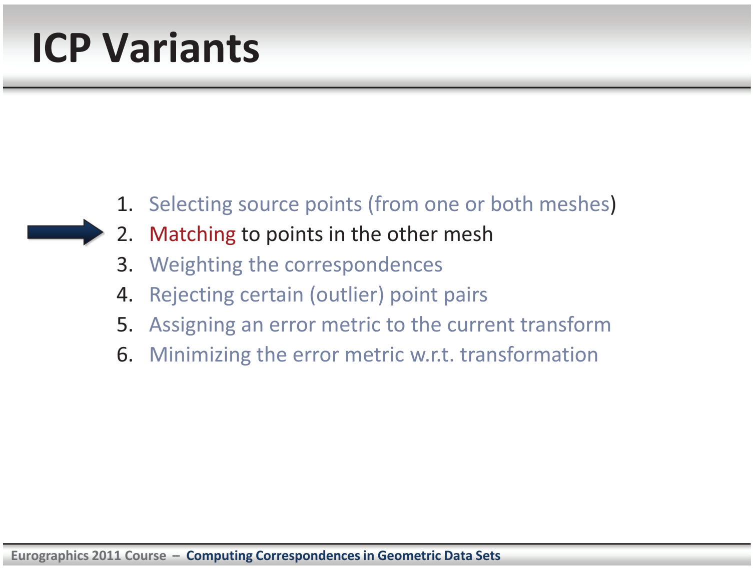
Closest *compatible* point

Stable sampling



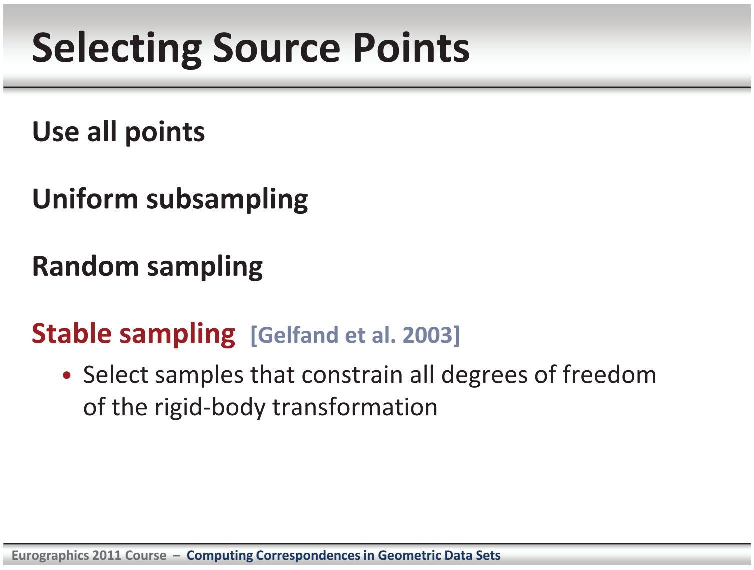
ICP Variants

➡ 1. Selecting source points (from one or both meshes)
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. transformation



ICP Variants

1. Selecting source points (from one or both meshes)
➡ 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. transformation



Selecting Source Points

Use all points

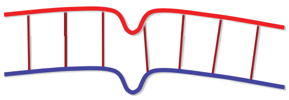
Uniform subsampling

Random sampling

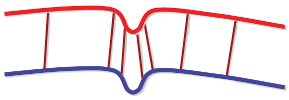
Stable sampling [Gelfand et al. 2003]

- Select samples that constrain all degrees of freedom of the rigid-body transformation

Stable Sampling

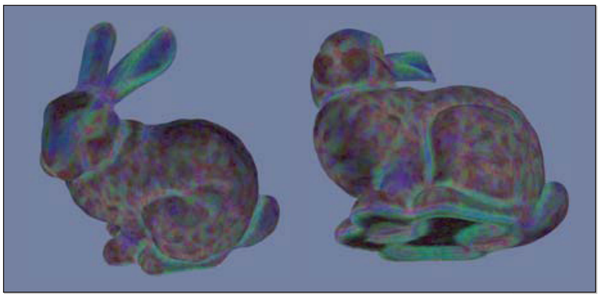


Uniform Sampling



Stable Sampling

Stability Analysis



Key:

⬜	3 DOFs stable	🟥	5 DOFs stable
🟦	4 DOFs stable	🟪	6 DOFs stable

Covariance Matrix

Aligning transform is given by $\mathbf{A}^{\mathrm{T}}\mathbf{A}x = \mathbf{A}^{\mathrm{T}}b$, where

$$\mathbf{A} = \begin{pmatrix} \leftarrow & p_1 \times n_1 & \rightarrow & \leftarrow & n_1 & \rightarrow \\ \leftarrow & p_2 \times n_2 & \rightarrow & \leftarrow & n_2 & \rightarrow \\ & \vdots & & & \vdots & \end{pmatrix}, \qquad x = \begin{pmatrix} r_x \\ r_y \\ r_z \\ t_x \\ t_y \\ t_z \end{pmatrix}, \qquad b = \begin{pmatrix} -(p_1 - q_1) \cdot n_1 \\ -(p_2 - q_2) \cdot n_2 \\ \vdots \end{pmatrix}$$

Covariance matrix $\mathbf{C} = \mathbf{A}^{\mathrm{T}}\mathbf{A}$ determines the change in error when surfaces are moved from optimal alignment

Sample Selection

Select points to prevent small eigenvalues

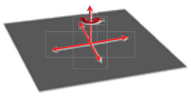
- Based on C obtained from sparse sampling

Simpler variant: normal-space sampling

- Select points with uniform distribution of normals
- Pro: faster, does not require eigenanalysis
- Con: only constrains translation

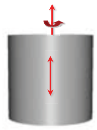
Sliding Directions

Eigenvectors of C with small eigenvalues correspond to sliding transformations



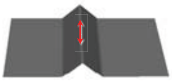
3 small eigenvalues
2 translation
1 rotation

3 small eigenvalues
3 rotation



2 small eigenvalues
1 translation
1 rotation

1 small eigenvalue
1 rotation

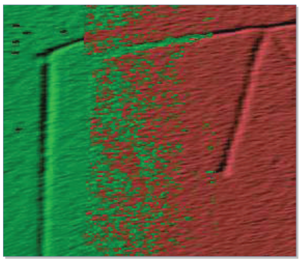


1 small eigenvalue
1 translation

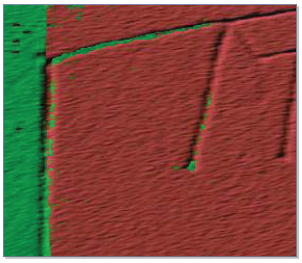
[Gelfand]

Result

Stability-based or normal-space sampling important for smooth areas with small features



Random sampling



Normal-space sampling

Selection vs. Weighting

Could achieve same effect with weighting

Hard to ensure enough samples in features except at high sampling rates

However, have to build special data structure

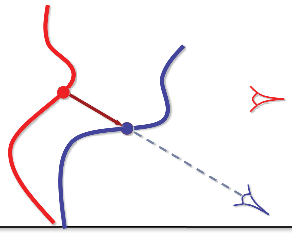
Preprocessing / run-time cost tradeoff

Projection to Find Correspondences

Idea: use a simpler algorithm to find correspondences

For range images, can simply project point [Blais 95]

- Constant-time
- Does not require precomputing a spatial data structure



Improving ICP Speed

Projection-based matching

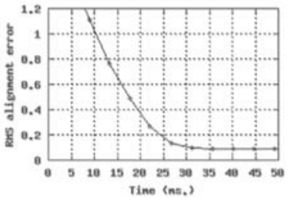
1. Selecting source points (from one or both meshes)
2. Matching to points in the other mesh
3. Weighting the correspondences
4. Rejecting certain (outlier) point pairs
5. Assigning an error metric to the current transform
6. Minimizing the error metric w.r.t. transformation

Projection-Based Matching

Slightly worse performance per iteration

Each iteration is one to two orders of magnitude faster than closest-point

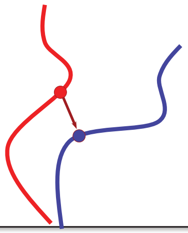
Result: can align two range images in a few milliseconds, vs. a few seconds



Finding Corresponding Points

Finding closest point is most expensive stage of the ICP algorithm

- Brute force search – O(n)
- Spatial data structure (e.g., k-d tree) – O(log n)



Application

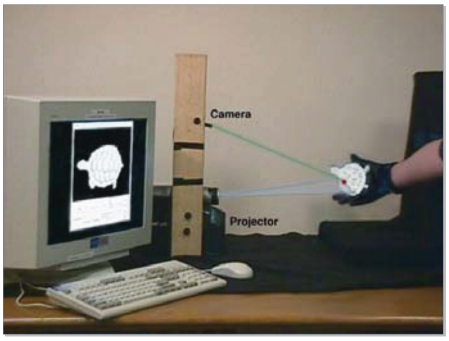
Given:

- A scanner that returns range images in real time
- Fast ICP
- Real-time merging and rendering

Result: 3D model acquisition

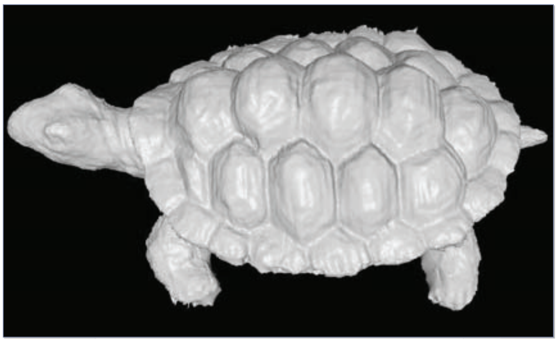
- Tight feedback loop with user
- Can see and fill holes while scanning

Scanner Layout



Photograph

Real-Time Result



Theoretical Analysis of ICP Variants

One way of studying performance is via empirical tests on various scenes

How to analyze performance analytically?

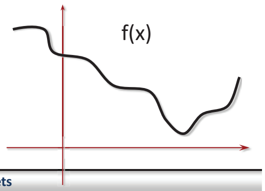
For example, when does point-to-plane help? Under what conditions does projection-based matching work?

What Does ICP Do?

Two ways of thinking about ICP:

- Solving the correspondence problem
- Minimizing point-to-surface squared distance

ICP is like (Gauss-) Newton method on an approximation of the distance function

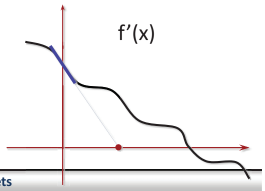


What Does ICP Do?

Two ways of thinking about ICP:

- Solving the correspondence problem
- Minimizing point-to-surface squared distance

ICP is like Newton’s method on an approximation of the distance function



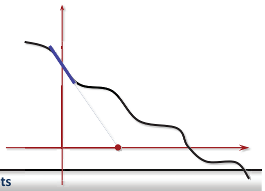
What Does ICP Do?

Two ways of thinking about ICP:

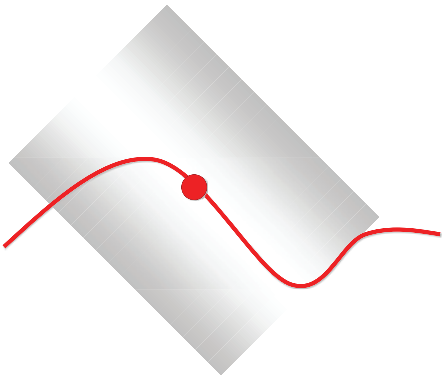
- Solving the correspondence problem
- Minimizing point-to-surface squared distance

ICP is like Newton's method on an approximation of the distance function

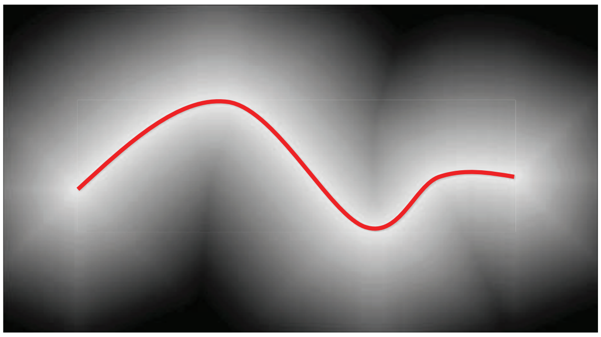
- ICP variants affect shape of global error function or local approximation



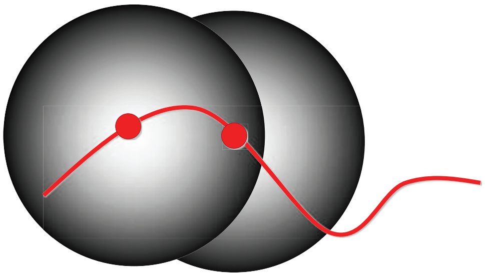
Point-to-Plane Distance



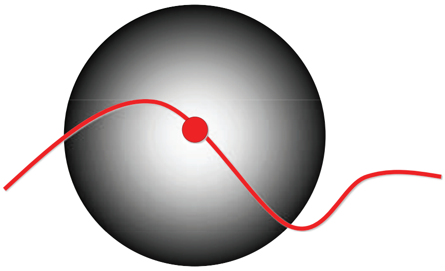
Point-to-Surface Distance



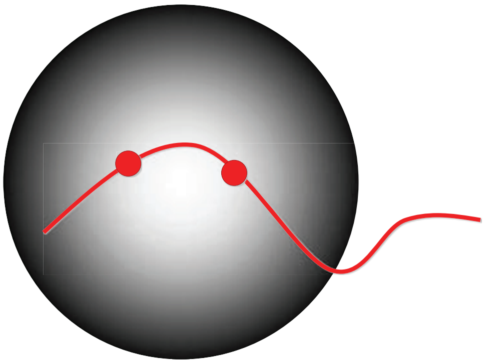
Point-to-Multiple-Point Distance



Point-to-Point Distance



Point-to-Multiple-Point Distance



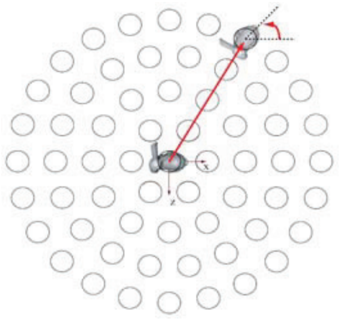
Soft Matching and Distance Functions

Soft matching equivalent to standard ICP on (some) filtered surface

Produces filtered version of distance function ⇒ fewer local minima

Multiresolution minimization [Turk & Levoy 94] **or softassign with simulated annealing (good description in** [Chui 03]**)**

Convergence Funnel



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

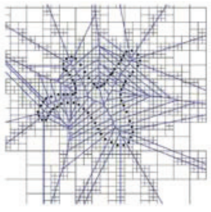
■ Converges

□ Does not converge

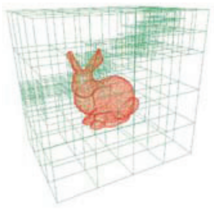
Mitra et al.'s Optimization

Precompute piecewise-quadratic approximation to distance field throughout space

Store in "d2tree" data structure



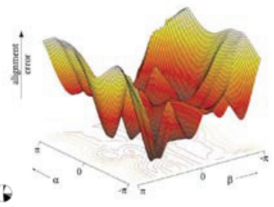
2D

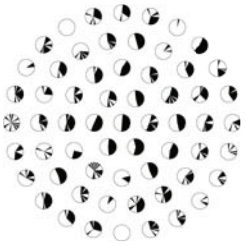


3D

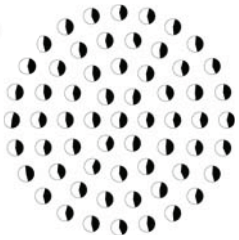
[Mitra et al. 2004]

Convergence Funnel





Plane-to-plane ICP



distance-field formulation

Mitra et al.'s Optimization

Precompute piecewise-quadratic approximation to distance field throughout space

Store in "d2tree" data structure

At run time, look up quadratic approximants and optimize using Newton's method

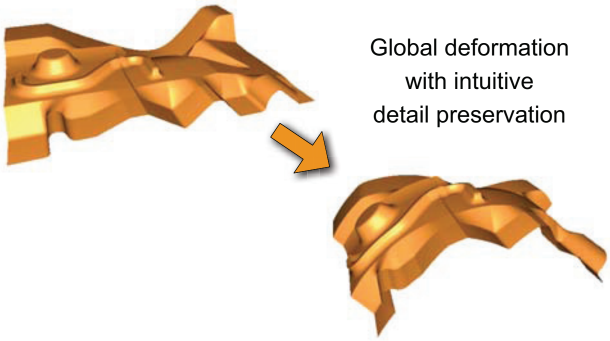
- More robust, wider basin of convergence
- Often fewer iterations, but more precomputation

Computing Correspondences in Geometric Datasets

Deformation Models

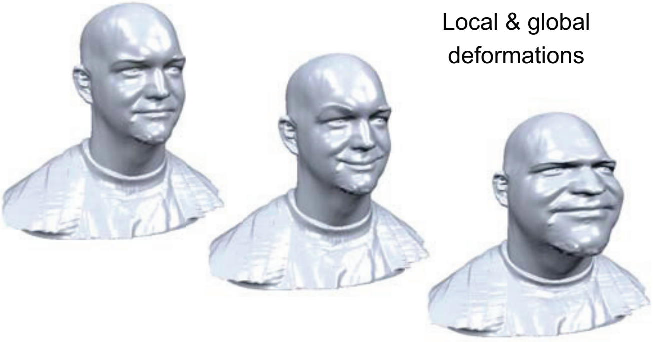
Eurographics 2011
LLANDUDNO UK | Bangor University School of Computer Science
11-15 April 2011

Mesh Deformation



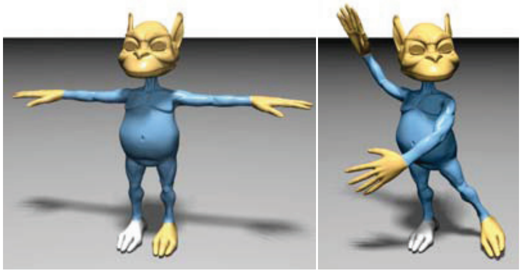
Global deformation
with intuitive
detail preservation

Mesh Deformation



Local & global
deformations

Mesh Deformation



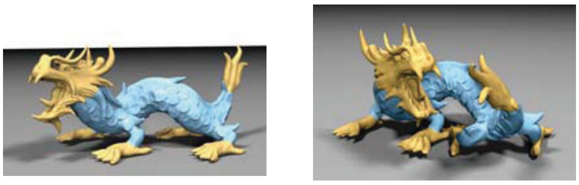
Character posing

Mesh Deformation

Editing of
“bad” meshes

Mesh Deformation

Editing of complex meshes



Mesh Deformation

Reconstruction of deforming objects

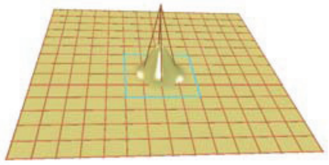
Overview

- Surface-Based Deformation
- Space Deformation
- Multiresolution Deformation
- Differential Coordinates
- Outlook: Nonlinear Methods

Spline Surfaces

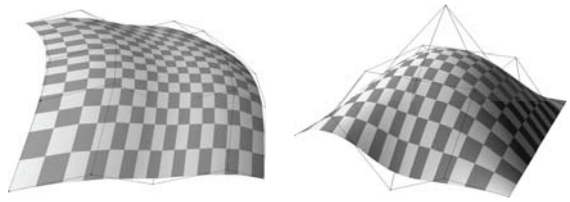
- Tensor product surfaces (“curves of curves”)
  - Rectangular grid of control points

$$\mathbf{s}(u,v) = \sum_{i=0}^{k}\sum_{j=0}^{l} \mathbf{d}_{i,j} N_i^n(u) N_j^n(v)$$



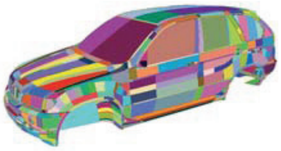
Spline Surfaces

- Tensor product surfaces (“curves of curves”)
  - Rectangular grid of control points
  - Rectangular surface patch



Spline Surfaces

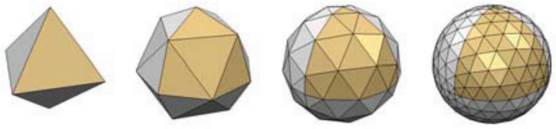
- Tensor product surfaces (“curves of curves”)
  - Rectangular grid of control points
  - Rectangular surface patch



- Problems:
  - Many patches for complex models
  - Smoothness across patch boundaries
  - Trimming for non-rectangular patches

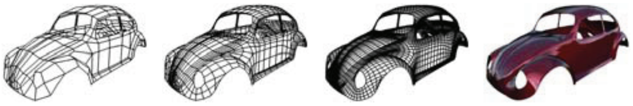
Subdivision Surfaces

- Generalization of spline curves / surfaces
  - Arbitrary control meshes
  - Successive refinement (subdivision)
  - Converges to smooth limit surface
  - Connection between splines and meshes



Subdivision Surfaces

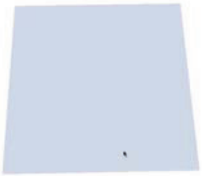
- Generalization of spline curves / surfaces
  - Arbitrary control meshes
  - Successive refinement (subdivision)
  - Converges to smooth limit surface
  - Connection between splines and meshes



Spline & Subdivision Surfaces

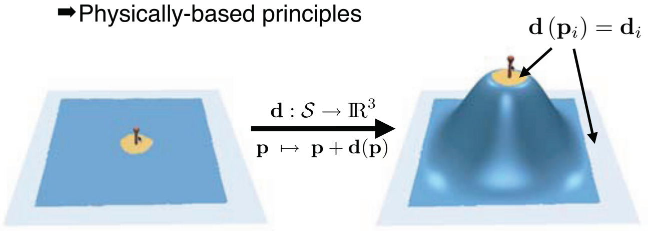
- Basis functions are smooth bumps
  - Fixed support
  - Fixed control grid
- Bound to control points
  - Initial patch layout is crucial
  - Requires experts!
- Decouple deformation from surface representation!

Modeling Metaphor



Modeling Metaphor

- Mesh deformation by displacement function **d**
  - Interpolate prescribed constraints
  - Smooth, intuitive deformation
  - ➡Physically-based principles



$\mathbf{d}\left(\mathbf{p}_i\right) = \mathbf{d}_i$

$\mathbf{d} : \mathcal{S} \to \mathbb{R}^3$

$\mathbf{p} \mapsto \mathbf{p} + \mathbf{d}(\mathbf{p})$

Physically-Based Deformation

- Non-linear stretching & bending energies

$$\int_{\Omega} k_s \underbrace{\boxed{\left\|\mathbf{I} - \mathbf{I}'\right\|^2}}_{\text{stretching}} + k_b \underbrace{\boxed{\left\|\mathbf{II} - \mathbf{II}'\right\|^2}}_{\text{bending}} \,\mathrm{d}u\mathrm{d}v$$

- Linearize energies

$$\int_{\Omega} k_s \underbrace{\boxed{\left(\|\mathbf{d}_u\|^2 + \|\mathbf{d}_v\|^2\right)}}_{\text{stretching}} + k_b \underbrace{\boxed{\left(\|\mathbf{d}_{uu}\|^2 + 2\,\|\mathbf{d}_{uv}\|^2 + \|\mathbf{d}_{vv}\|^2\right)}}_{\text{bending}} \mathrm{d}u\mathrm{d}v$$

Physically-Based Deformation

- Minimize linearized bending energy

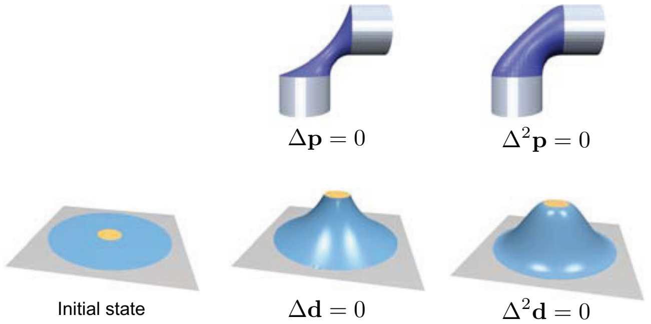
$$E(\mathbf{d}) \;=\; \int_{\mathcal{S}} \|\mathbf{d}_{uu}\|^2 + 2\,\|\mathbf{d}_{uv}\|^2 + \|\mathbf{d}_{vv}\|^2 \,\mathrm{d}\mathcal{S} \qquad \boxed{f(x) \to \min}$$

- Variational calculus, Euler-Lagrange PDE

$$\Delta^2 \mathbf{d} \;:=\; \mathbf{d}_{uuuu} + 2\mathbf{d}_{uuvv} + \mathbf{d}_{vvvv} \;=\; 0 \qquad \boxed{f'(x) = 0}$$

➡ “Best” deformation that satisfies constraints

Deformation Energies



$\Delta \mathbf{p} = 0$ | $\Delta^2 \mathbf{p} = 0$

Initial state

$\Delta \mathbf{d} = 0$
(Membrane)

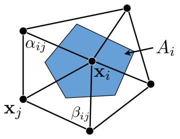
$\Delta^2 \mathbf{d} = 0$
(Thin plate)

Discretization

- Laplace discretization

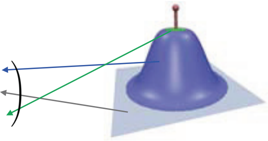
$$\Delta \mathbf{d}_i = \frac{1}{2A_i} \sum_{j \in \mathcal{N}_i} (\cot \alpha_{ij} + \cot \beta_{ij})(\mathbf{d}_j - \mathbf{d}_i)$$

$$\Delta^2 \mathbf{d}_i = \Delta(\Delta \mathbf{d}_i)$$



- Sparse linear system

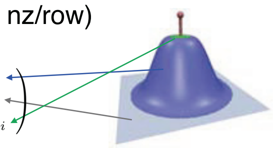
$$\underbrace{\begin{pmatrix} & \Delta^2 & \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix}}_{=:\mathbf{M}} \begin{pmatrix} \vdots \\ \mathbf{d}_i \\ \vdots \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \delta \mathbf{h}_i \end{pmatrix}$$



Discretization

- Sparse linear system (19 nz/row)

$$\underbrace{\begin{pmatrix} & \Delta^2 & \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix}}_{=:\mathbf{M}} \begin{pmatrix} \vdots \\ \mathbf{d}_i \\ \vdots \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \delta \mathbf{h}_i \end{pmatrix}$$



- Can be turned into symm. pos. def. system
  - Right hand sides changes each frame!
  - Use efficient linear solvers...

Literature

- Botsch & Kobbelt, *"An intuitive framework for real-time freeform modeling"*, SIGGRAPH 2004
- Botsch & Sorkine, *"On linear variational surface deformation methods"*, TVCG 2007
- Botsch et al, *"Efficient linear system solvers for mesh processing"*, IMA Math. of Surfaces 2005

Overview

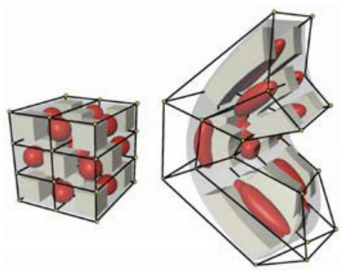
- Surface-Based Deformation
- **Space Deformation**
- Multiresolution Deformation
- Differential Coordinates
- Outlook: Nonlinear Methods

Surface-Based Deformation

- Problems with
  - Highly complex models
  - Topological inconsistencies
  - Geometric degeneracies

Freeform Deformation

- Deform object's bounding box
  - Implicitly deforms embedded objects



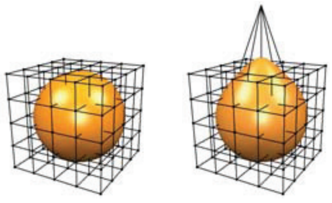
Freeform Deformation

- Deform object's bounding box
  - Implicitly deforms embedded objects
- Tri-variate tensor-product spline

$$\mathbf{d}(u,v,w) = \sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{k=0}^{n} \mathbf{d}_{ijk} N_i(u)\, N_j(v)\, N_k(w)$$

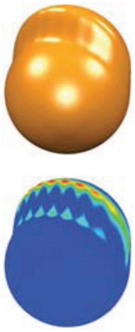
Freeform Deformation

- Deform object's bounding box
  - Implicitly deforms embedded objects
- Tri-variate tensor-product spline



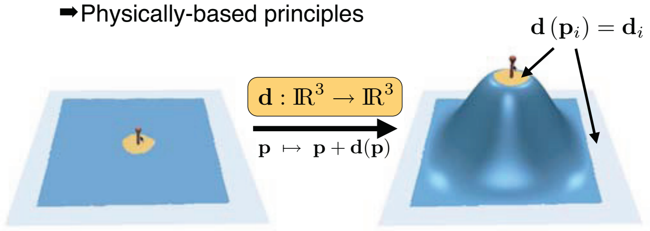
Freeform Deformation

- Deform object's bounding box
  - Implicitly deforms embedded objects
- Tri-variate tensor-product spline
  - Aliasing artifacts
- Interpolate deformation constraints?
  - Only in least squares sense



Modeling Metaphor

- Mesh deformation by displacement function **d**
  - Interpolate prescribed constraints
  - Smooth, intuitive deformation
  - ➡Physically-based principles



$\mathbf{d}(\mathbf{p}_i) = \mathbf{d}_i$

$\mathbf{d} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$

$\mathbf{p} \mapsto \mathbf{p} + \mathbf{d}(\mathbf{p})$

Volumetric Energy Minimization

- Minimize similar energies to surface case

$$\int_{\mathbb{R}^3} \|\mathbf{d}_{uu}\|^2 + \|\mathbf{d}_{uv}\|^2 + \ldots + \|\mathbf{d}_{ww}\|^2 \,\mathrm{d}V \;\rightarrow\; \min$$

- But displacements function lives in 3D...
  - Need a volumetric space tessellation?
  - No, same functionality provided by RBFs

Radial Basis Functions

- Represent deformation by RBFs

$$\mathbf{d}(\mathbf{x}) = \sum_j \mathbf{w}_j \cdot \varphi\left(\|\mathbf{c}_j - \mathbf{x}\|\right) + \mathbf{p}(\mathbf{x})$$

- Triharmonic basis function $\varphi(r) = r^3$
  - C^2 boundary constraints
  - Highly smooth / fair interpolation

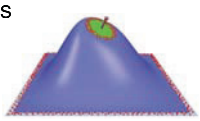
$$\int_{\mathbb{R}^3} \|\mathbf{d}_{uuu}\|^2 + \|\mathbf{d}_{vuu}\|^2 + \ldots + \|\mathbf{d}_{www}\|^2 \, du\, dv\, dw \;\to\; \min$$

RBF Fitting

- Represent deformation by RBFs

$$\mathbf{d}(\mathbf{x}) = \sum_j \mathbf{w}_j \cdot \varphi\left(\|\mathbf{c}_j - \mathbf{x}\|\right) + \mathbf{p}(\mathbf{x})$$

- RBF fitting
  - Interpolate displacement constraints
  - Solve linear system for $\mathbf{w}_j$ and $\mathbf{p}$

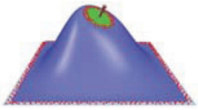


RBF Fitting

- Represent deformation by RBFs

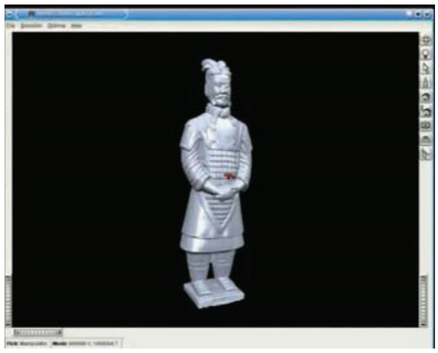
$$\mathbf{d}(\mathbf{x}) = \sum_j \mathbf{w}_j \cdot \varphi\left(\|\mathbf{c}_j - \mathbf{x}\|\right) + \mathbf{p}(\mathbf{x})$$

- RBF evaluation
  - Function $\mathbf{d}$ transforms points
  - Jacobian $\nabla\mathbf{d}$ transforms normals
  - Precompute basis functions
  - Evaluate on the GPU!

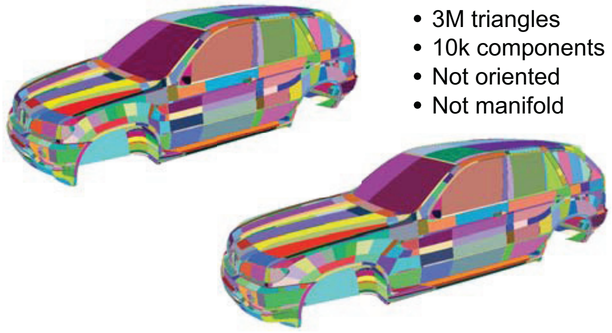


RBF Deformation

1M vertices

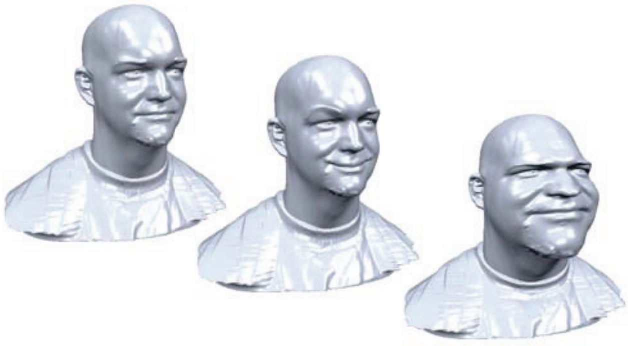


"Bad Meshes"



- 3M triangles
- 10k components
- Not oriented
- Not manifold

Local & Global Deformations



Literature

- Sederberg & Parry, *“Free-Form Deformation of Solid Geometric Models”*, SIGGRAPH 1986
- Botsch & Kobbelt, *“Real-time shape editing using radial basis functions”*, Eurographics 2005

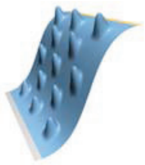
Overview

- Surface-Based Deformation
- Space Deformation
- **Multiresolution Deformation**
- Differential Coordinates
- Outlook: Nonlinear Methods

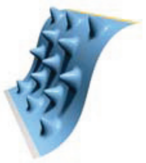
Multiresolution Modeling

- Even pure translations induce local rotations!
  - Inherently non-linear coupling
- Or: linear model + multi-scale decomposition...

Original

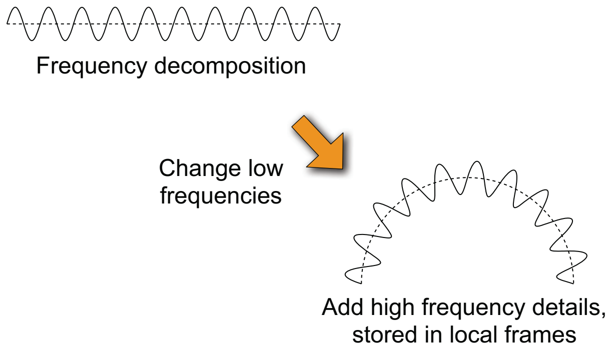


Linear deform.



Non-linear deform.

Multiresolution Editing

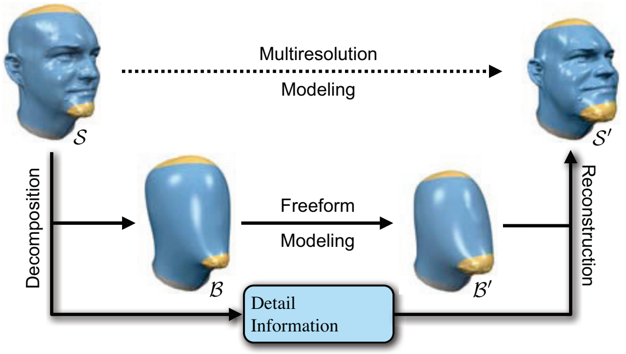


Frequency decomposition

Change low frequencies

Add high frequency details, stored in local frames

Multiresolution Editing

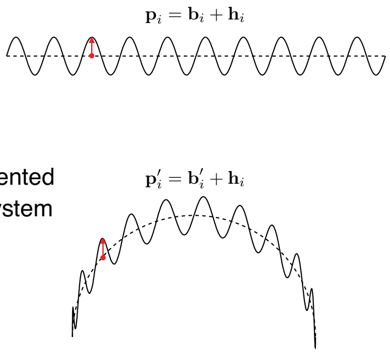


Detail Information

Global Frame Details

- S and B have identical connectivity
- Vertices $\mathbf{p}_i$ and $\mathbf{b}_i$ are corresponding
- Detail vector $\mathbf{h}_i$ represented in <u>global</u> coordinate system

➡Details don’t rotate

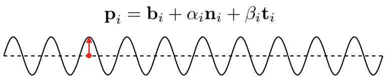


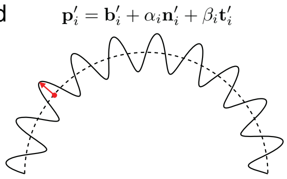
$\mathbf{p}_i = \mathbf{b}_i + \mathbf{h}_i$
$\mathbf{p}'_i = \mathbf{b}'_i + \mathbf{h}_i$

Local Frame Details

- S and B have identical connectivity
- Vertices $\mathbf{p}_i$ and $\mathbf{b}_i$ are corresponding
- Detail vector $\mathbf{h}_i$ represented in <u>local</u> coordinate system (normal & tangent vectors)

➡Details rotate

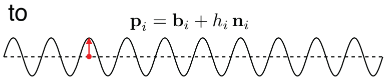

$\mathbf{p}_i = \mathbf{b}_i + \alpha_i \mathbf{n}_i + \beta_i \mathbf{t}_i$

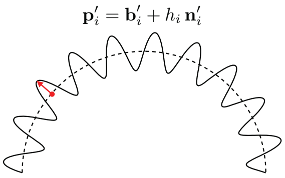

$\mathbf{p}_i' = \mathbf{b}_i' + \alpha_i \mathbf{n}_i' + \beta_i \mathbf{t}_i'$

Normal Displacements

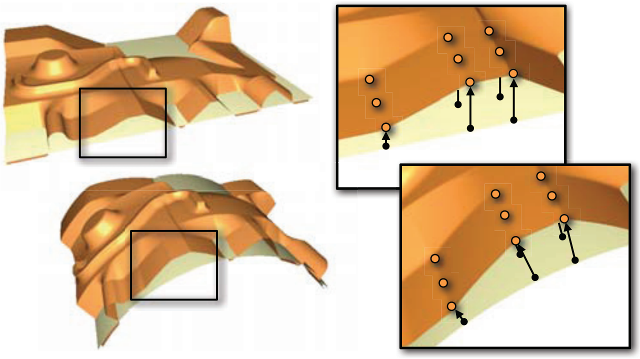
- Displacements parallel to normal field of B
- Barycentric interpolation yields smooth normal field (like Phong shading)
- Base point $\mathbf{b}_i$ now is not necessarily a vertex of B

➡Details rotate, no tangent component required


$\mathbf{p}_i = \mathbf{b}_i + h_i \, \mathbf{n}_i$

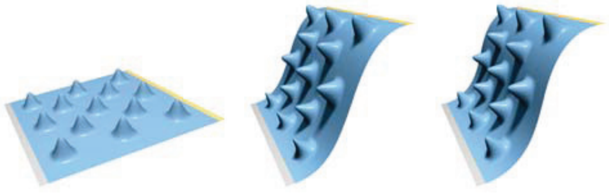

$\mathbf{p}_i' = \mathbf{b}_i' + h_i \, \mathbf{n}_i'$

Normal Displacements



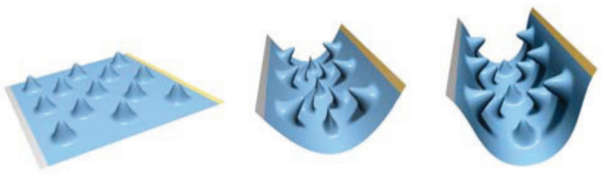
Limitations

- Neighboring displacements are not coupled
  - Surface bending changes their angle
  - Leads to volume changes or self-intersections



Limitations

- Neighboring displacements are not coupled
  - Surface bending changes their angle
  - Leads to volume changes or self-intersections



Limitations

- Neighboring displacements are not coupled
  - Surface bending changes their angle
  - Leads to volume changes or self-intersections
- Multiresolution hierarchy difficult to compute for meshes of complex topology / geometry
- Might require more hierarchy levels

Overview

- Surface-Based Deformation
- Space Deformation
- Multiresolution Deformation
- **Differential Coordinates**
- Outlook: Nonlinear Methods

Differential Coordinates

- Manipulate *differential coordinates* instead of spatial coordinates
  - Gradients, Laplacians, ...
- Then find mesh with desired differential coords
  - Basically an integration step

Gradient-Based Editing

- Manipulate gradient field of a function (surface)

$$\mathbf{g} = \nabla \mathbf{f} \qquad \mathbf{g} \mapsto \mathbf{g}'$$

- Find function whose gradient is (close to) $\mathbf{g}$'

$$\mathbf{f}' = \underset{\mathbf{f}}{\operatorname{argmin}} \int_\Omega \|\nabla \mathbf{f} - \mathbf{g}'\|^2 \, \mathrm{d}u\mathrm{d}v$$

- Variational calculus yields Euler-Lagrange PDE

$$\Delta \mathbf{f}' = \operatorname{div} \mathbf{g}'$$

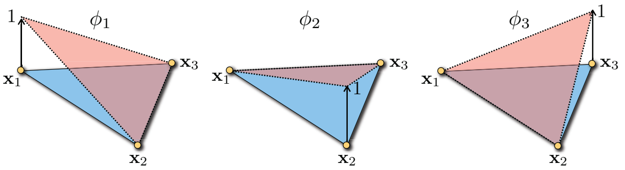
Gradient-Based Editing

- Use piecewise linear coordinate function

$$\mathbf{p}(u,v) = \sum_{v_i} \mathbf{p}_i \cdot \phi_i(u,v)$$

- Its gradient is

$$\nabla \mathbf{p}(u,v) = \sum_{v_i} \mathbf{p}_i \cdot \nabla \phi_i(u,v)$$



Gradient-Based Editing

- Use piecewise linear coordinate function

$$\mathbf{p}(u,v) = \sum_{v_i} \mathbf{p}_i \cdot \phi_i(u,v)$$

- Its gradient is

$$\nabla \mathbf{p}(u,v) = \sum_{v_i} \mathbf{p}_i \cdot \nabla \phi_i(u,v)$$

- It is constant per triangle

$$\nabla \mathbf{p}|_{f_j} \; =: \; \mathbf{G}_j \in \mathbb{R}^{3\times 3}$$

Gradient-Based Editing

- Gradient of coordinate function $\mathbf{p}$
  - Constant per triangle $\nabla \mathbf{p}|_{f_j} \; =: \; \mathbf{G}_j \in \mathbb{R}^{3\times 3}$

$$\begin{pmatrix} \mathbf{G}_1 \\ \vdots \\ \mathbf{G}_F \end{pmatrix} = \underbrace{\mathbf{G}}_{\in \mathbb{R}^{3F\times V}} \cdot \begin{pmatrix} \mathbf{p}_1^T \\ \vdots \\ \mathbf{p}_V^T \end{pmatrix}$$

- Manipulate per-face gradients

$$\mathbf{G}_j \mapsto \mathbf{G}_j'$$

Gradient-Based Editing

- Reconstruct mesh from changed gradients
  - Overdetermined problem $\mathbf{G} \in \mathbb{R}^{3F\times V}$

$$\mathbf{G}\cdot\begin{pmatrix}\mathbf{p}_1'^{\,T}\\ \vdots \\ \mathbf{p}_V'^{\,T}\end{pmatrix} = \begin{pmatrix}\mathbf{G}_1'\\ \vdots \\ \mathbf{G}_F'\end{pmatrix}$$

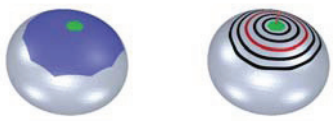
Gradient-Based Editing

- Reconstruct mesh from changed gradients
  - Overdetermined problem $\mathbf{G} \in \mathbb{R}^{3F\times V}$
  - Weighted least squares system
  - Linear Laplace system

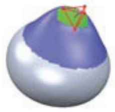
$$\underset{\operatorname{div}\nabla=\Delta}{\boxed{\mathbf{G}^T\mathbf{D}\mathbf{G}}}\cdot\begin{pmatrix}\mathbf{p}_1'^{\,T}\\ \vdots \\ \mathbf{p}_V'^{\,T}\end{pmatrix} = \underset{\operatorname{div}}{\boxed{\mathbf{G}^T\mathbf{D}}}\cdot\begin{pmatrix}\mathbf{G}_1'\\ \vdots \\ \mathbf{G}_F'\end{pmatrix}$$

Manipulate Gradients

- Manipulate per-face gradients $\mathbf{G}_j \mapsto \mathbf{G}_j'$
  1. Compute gradient of handle deformation
  2. Extract rotation and scale/shear components
  3. Compute smooth scalar blending field
  4. Apply damped rotations to gradients



Deformation Gradient



- Handle has been transformed affinely

$$\mathbf{T}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{t}$$

- Deformation gradient is

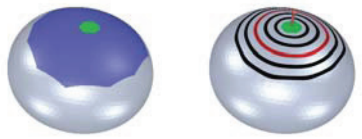
$$\nabla\mathbf{T}(\mathbf{x}) = \mathbf{A}$$

- Polar decomposition gives rotation and scale/shear components $\mathbf{R}$ and $\mathbf{S}$

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T \quad\rightarrow\quad \mathbf{A} = \mathbf{R}\mathbf{S}\,,\ \mathbf{R} = \mathbf{U}\mathbf{V}^T\,,\ \mathbf{S} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{V}^T$$

Construct Scalar Field

- Construct smooth scalar field [0,1]
  - $s(\mathbf{x})=1$: Full deformation (handle)
  - $s(\mathbf{x})=0$: No deformation (fixed part)
  - $s(\mathbf{x})\in(0,1)$: Damp handle transformation (in between)



Construct Scalar Field

- How to construct scalar field?
  - Either use Euclidean/geodesic distance

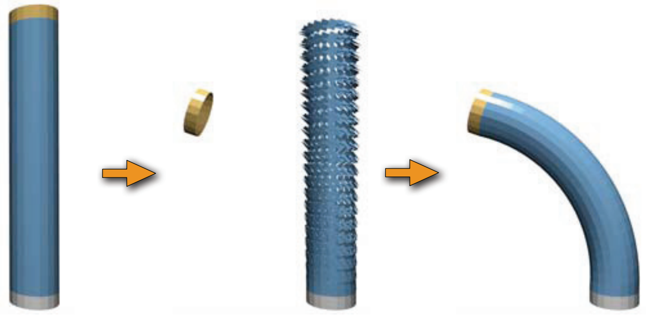
$$s(\mathbf{p}) = \frac{\operatorname{dist}_0(\mathbf{p})}{\operatorname{dist}_0(\mathbf{p}) + \operatorname{dist}_1(\mathbf{p})}$$

  - Or use harmonic field
    - Solve $\Delta(s) = 0$
    - with $s(\mathbf{p}) = \begin{cases} 1 & \mathbf{p} \in \text{handle} \\ 0 & \mathbf{p} \in \text{fixed} \end{cases}$

Damp Handle Transformation

- Original gradient of handle transformation
  - Rotation: $\mathrm{R}(\mathbf{c}, \mathbf{a}, \alpha)$
  - Scaling: $\mathrm{S}(\sigma)$
- Damping for triangle (v_i, v_j, v_k) is $\lambda = s((\mathbf{p}_i+\mathbf{p}_j+\mathbf{p}_k)/3)$
- Gradient damped by scalar λ
  - Rotation: $\mathrm{R}(\mathbf{c}, \mathbf{a}, \lambda\cdot\alpha)$
  - Scaling: $\mathrm{S}(\lambda\cdot\sigma + (1-\lambda)\cdot 1)$

Gradient-Based Editing



Laplacian-Based Editing

- Manipulate Laplacians field of a surface

$$\boldsymbol{\delta}_i = \Delta_{\mathcal{S}}(\mathbf{p}_i) \ , \quad \boldsymbol{\delta}_i \mapsto \boldsymbol{\delta}'_i$$

- Find surface whose Laplacian is (close to) $\boldsymbol{\delta}$'

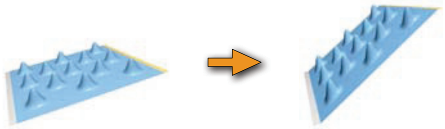
$$\mathbf{p}' = \underset{\mathbf{p}}{\operatorname{argmin}} \int_{\Omega} \left\| \Delta_{\mathcal{S}}\mathbf{p} - \boldsymbol{\delta}' \right\|^2 \mathrm{d}u\mathrm{d}v$$

- Variational calculus yields Euler-Lagrange PDE

$$\Delta^2_{\mathcal{S}}\mathbf{p}' = \Delta_{\mathcal{S}}\boldsymbol{\delta}'$$

Limitations

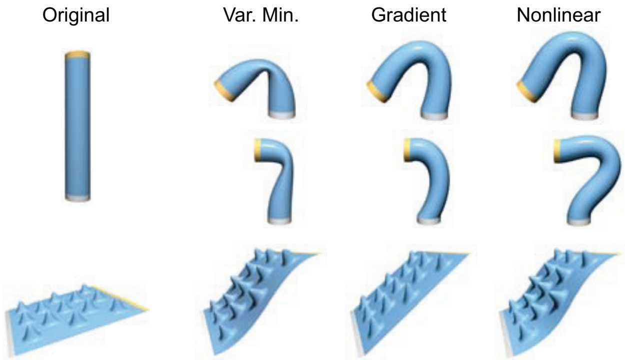
- Differential coordinates work well for rotations
  - Represented by deformation gradient
- Translations don't change deformation gradient
  - Translations don't change surface gradients / Lapl.
  - *"Translation insensitivity"*



Overview

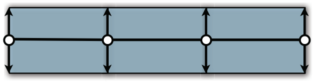
- Surface-Based Deformation
- Space Deformation
- Multiresolution Deformation
- Differential Coordinates
- **Outlook: Nonlinear Methods**

Comparison



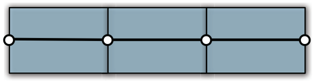
PriMo

- Qualitatively emulate thin-shell behavior
- Thin volumetric layer around center surface
- Extrude polygonal cell per mesh face



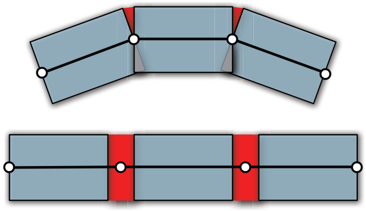
PriMo

- How to deform cells?
  - – FEM has problems if elements degenerate...
- Prevent cells from degenerating
  - ➡ Keep them *rigid*



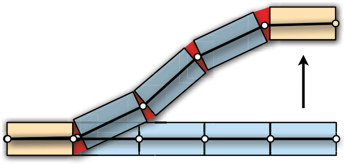
PriMo

- Connect cells along their faces
  - – Nonlinear elastic energy
  - – Measures bending, stretching, twisting, ...

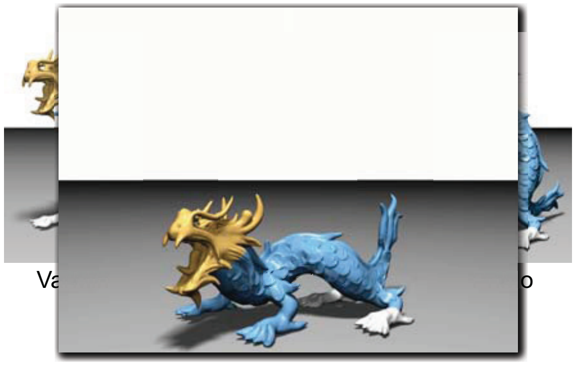


PriMo

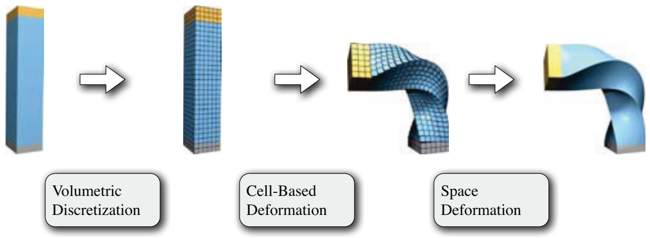
1. Extrude Prisms
2. Prescribes position/orientation for cells
3. Find optimal rigid motions per cell
4. Update vertices by average cell transformations



PriMo



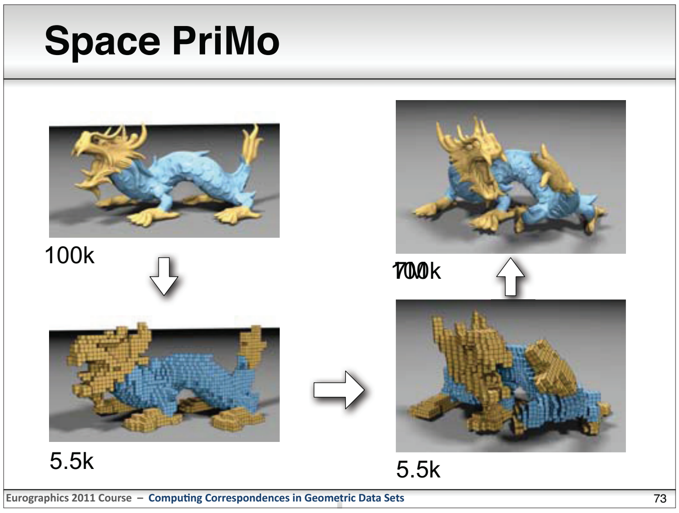
Space PriMo



Volumetric Discretization

Cell-Based Deformation

Space Deformation



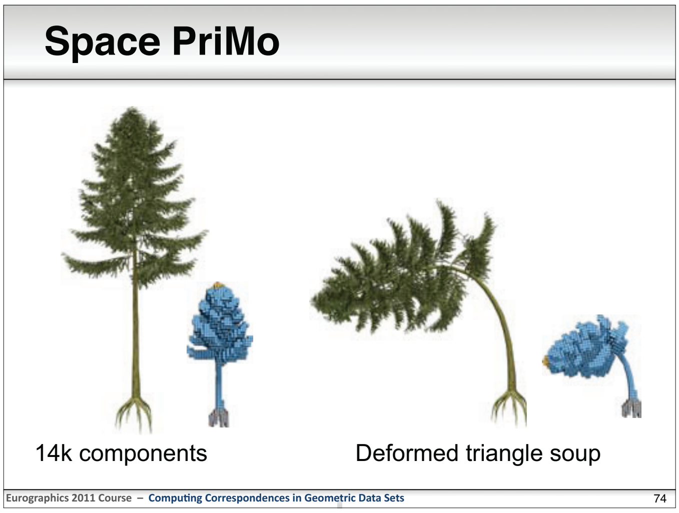
Space PriMo

100k

700k

5.5k

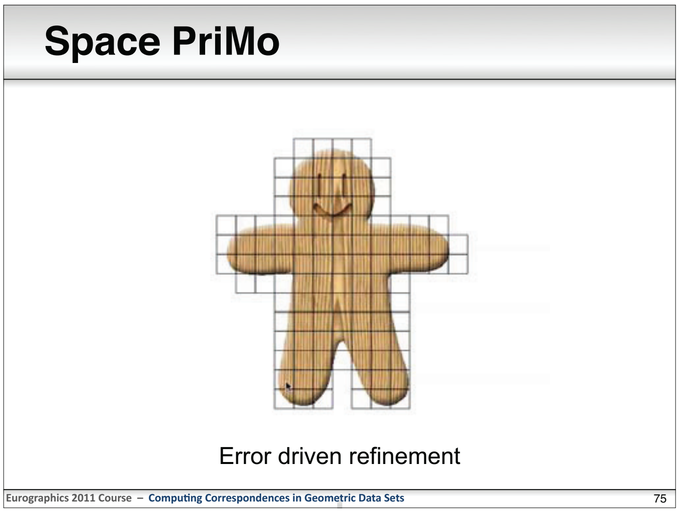
5.5k



Space PriMo

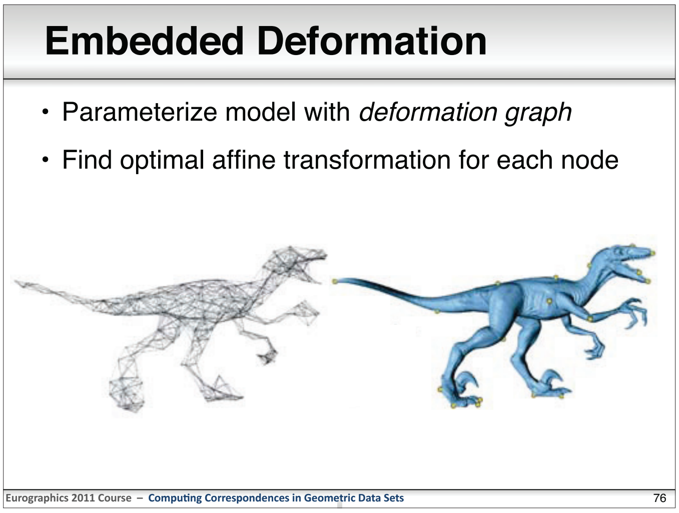
14k components

Deformed triangle soup



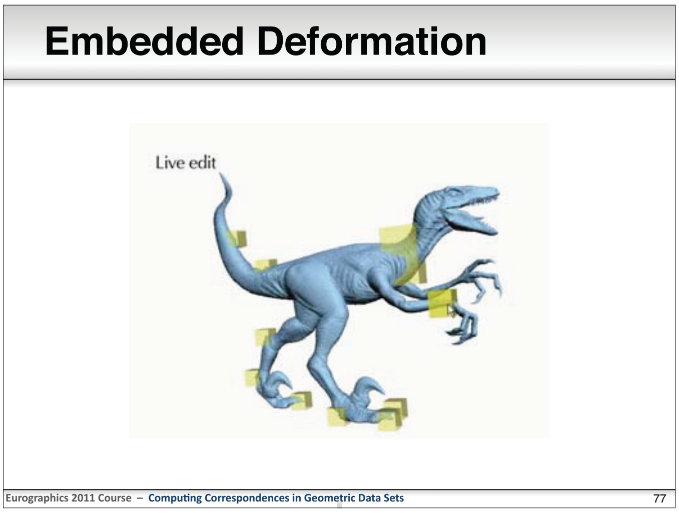
Space PriMo

Error driven refinement



Embedded Deformation

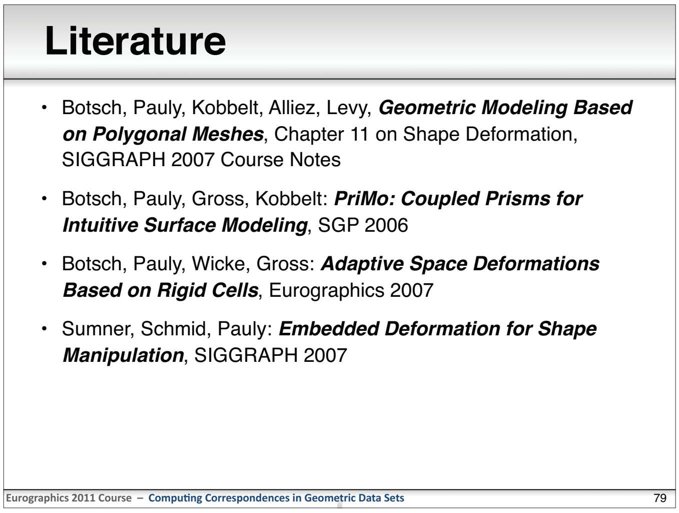
- Parameterize model with *deformation graph*
- Find optimal affine transformation for each node



Embedded Deformation

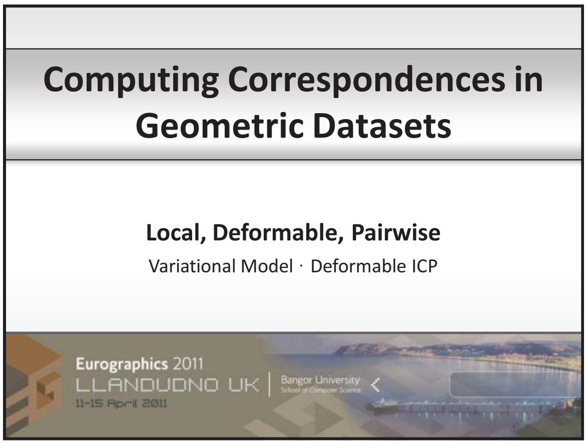
Overview

- Surface-Based Deformation
- Space Deformation
- Multiresolution Deformation
- Differential Coordinates
- Outlook: Nonlinear Methods



Literature

- Botsch, Pauly, Kobbelt, Alliez, Levy, ***Geometric Modeling Based on Polygonal Meshes***, Chapter 11 on Shape Deformation, SIGGRAPH 2007 Course Notes
- Botsch, Pauly, Gross, Kobbelt: ***PriMo: Coupled Prisms for Intuitive Surface Modeling***, SGP 2006
- Botsch, Pauly, Wicke, Gross: ***Adaptive Space Deformations Based on Rigid Cells***, Eurographics 2007
- Sumner, Schmid, Pauly: ***Embedded Deformation for Shape Manipulation***, SIGGRAPH 2007

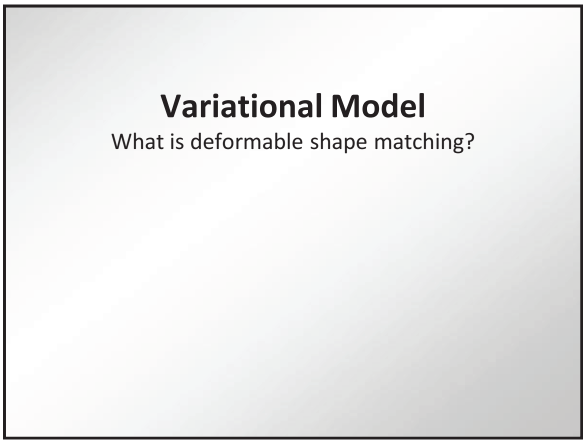


Computing Correspondences in Geometric Datasets

Local, Deformable, Pairwise

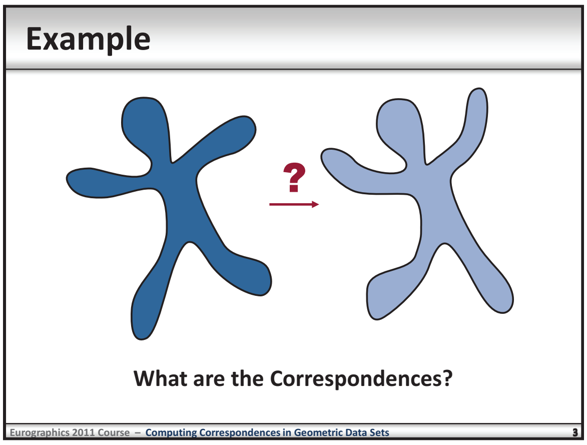
Variational Model · Deformable ICP

Eurographics 2011
LLANDUDNO UK
11-15 April 2011
Bangor University
School of Computer Science



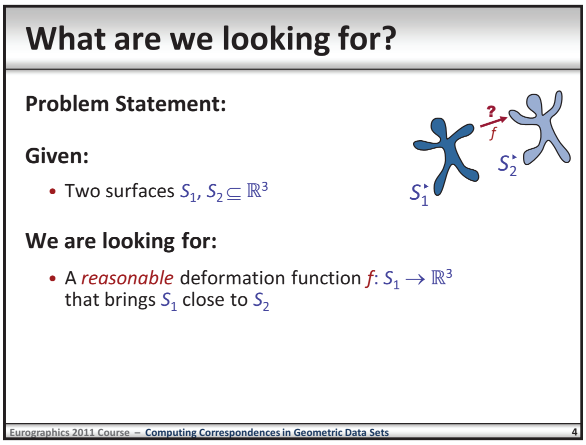
Variational Model

What is deformable shape matching?



Example

What are the Correspondences?



What are we looking for?

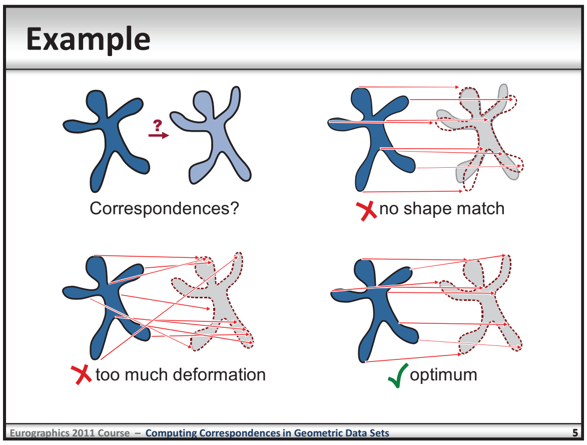
Problem Statement:

Given:

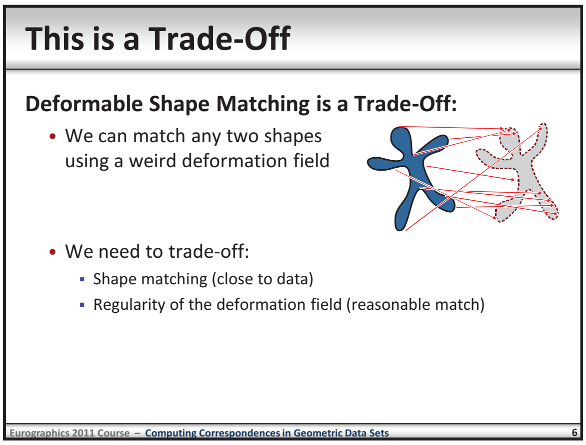
- Two surfaces $S_1, S_2 \subseteq \mathbb{R}^3$

We are looking for:

- A *reasonable* deformation function $f: S_1 \to \mathbb{R}^3$ that brings S_1 close to S_2



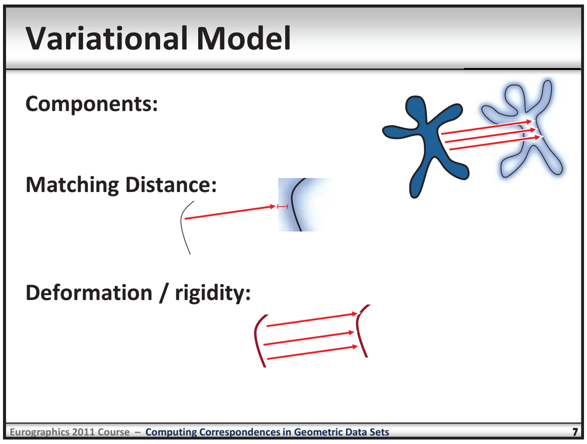
Example



This is a Trade-Off

Deformable Shape Matching is a Trade-Off:

- We can match any two shapes using a weird deformation field
- We need to trade-off:
  - Shape matching (close to data)
  - Regularity of the deformation field (reasonable match)

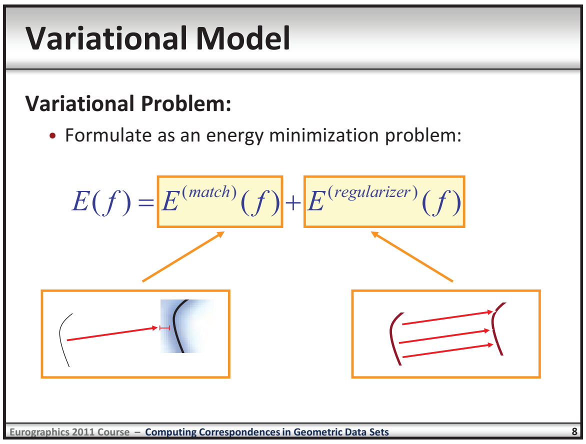


Variational Model

Components:

Matching Distance:

Deformation / rigidity:

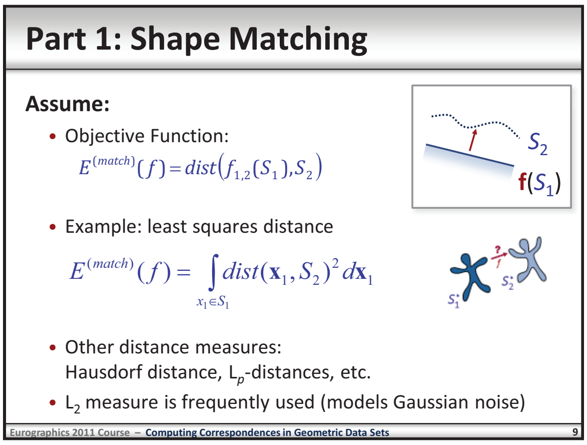


Variational Model

Variational Problem:

- Formulate as an energy minimization problem:

$$E(f) = \boxed{E^{(match)}(f)} + \boxed{E^{(regularizer)}(f)}$$



Part 1: Shape Matching

Assume:

- Objective Function:

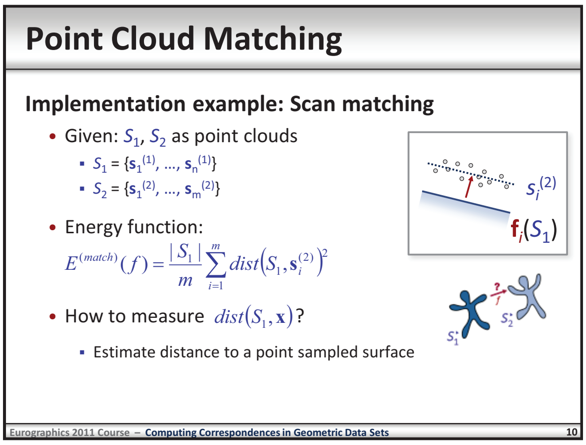
$$E^{(match)}(f) = dist\left(f_{1,2}(S_1), S_2\right)$$

- Example: least squares distance

$$E^{(match)}(f) = \int_{x_1 \in S_1} dist(\mathbf{x}_1, S_2)^2 \, d\mathbf{x}_1$$

- Other distance measures: Hausdorf distance, L$_p$-distances, etc.
- L$_2$ measure is frequently used (models Gaussian noise)

S_2
$\mathbf{f}(S_1)$
?
f
S_1
S_2



Point Cloud Matching

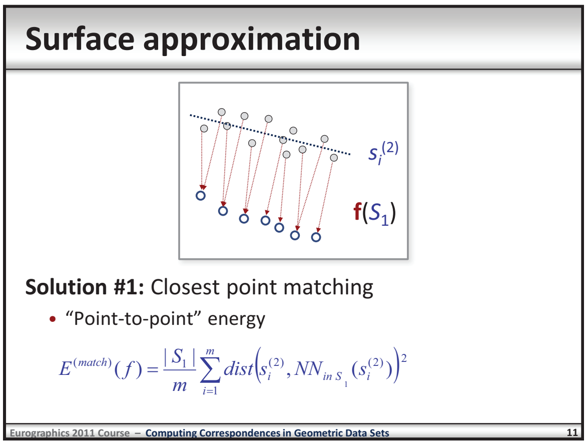
Implementation example: Scan matching

- Given: S_1, S_2 as point clouds
  - $S_1 = \{\mathbf{s}_1^{(1)}, \ldots, \mathbf{s}_n^{(1)}\}$
  - $S_2 = \{\mathbf{s}_1^{(2)}, \ldots, \mathbf{s}_m^{(2)}\}$
- Energy function:

$$E^{(match)}(f) = \frac{|S_1|}{m} \sum_{i=1}^{m} dist\left(S_1, \mathbf{s}_i^{(2)}\right)^2$$

- How to measure $dist\left(S_1, \mathbf{x}\right)$?
  - Estimate distance to a point sampled surface

$s_i^{(2)}$
$\mathbf{f}_i(S_1)$
?
f
S_1
S_2



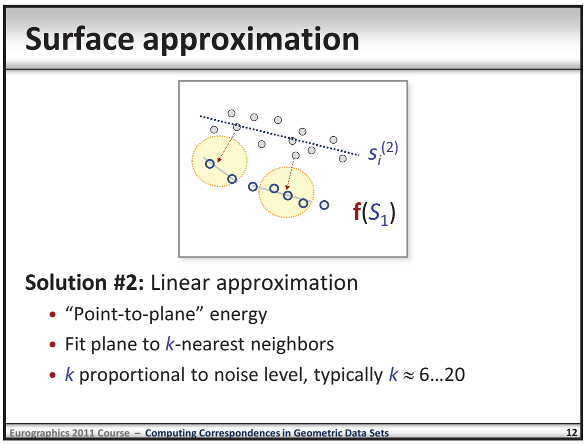
Surface approximation

$s_i^{(2)}$
$\mathbf{f}(S_1)$

Solution #1: Closest point matching

- "Point-to-point" energy

$$E^{(match)}(f) = \frac{|S_1|}{m} \sum_{i=1}^{m} dist\left(s_i^{(2)}, NN_{in\, S_1}(s_i^{(2)})\right)^2$$

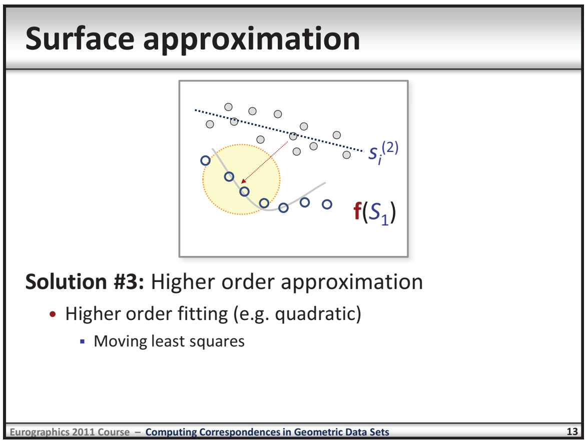


Surface approximation

$s_i^{(2)}$
$\mathbf{f}(S_1)$

Solution #2: Linear approximation

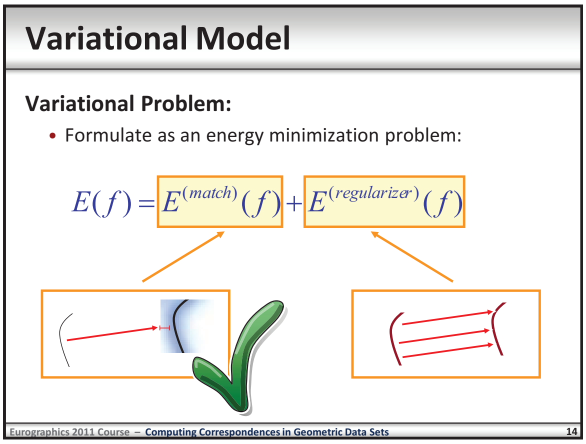
- "Point-to-plane" energy
- Fit plane to *k*-nearest neighbors
- *k* proportional to noise level, typically *k* ≈ 6…20



Surface approximation

Solution #3: Higher order approximation

- Higher order fitting (e.g. quadratic)
  - Moving least squares

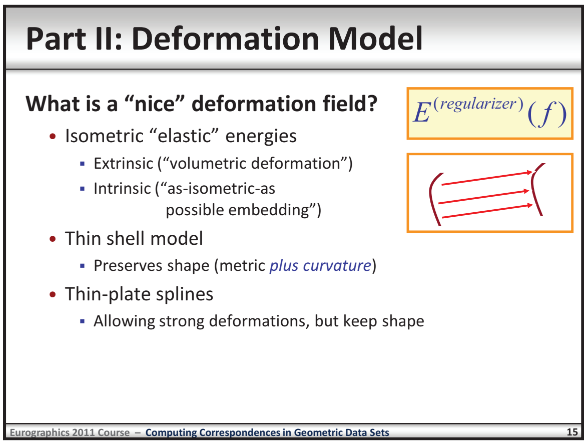


Variational Model

Variational Problem:

- Formulate as an energy minimization problem:

$$E(f) = \boxed{E^{(match)}(f)} + \boxed{E^{(regularizer)}(f)}$$

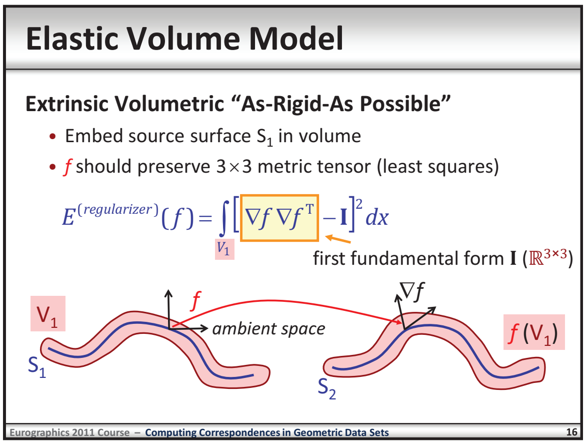


Part II: Deformation Model

$$\boxed{E^{(regularizer)}(f)}$$

What is a “nice” deformation field?

- Isometric “elastic” energies
  - Extrinsic (“volumetric deformation”)
  - Intrinsic (“as-isometric-as possible embedding”)
- Thin shell model
  - Preserves shape (metric *plus curvature*)
- Thin-plate splines
  - Allowing strong deformations, but keep shape



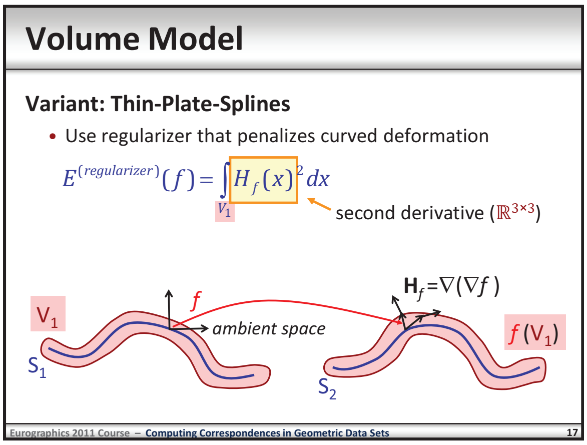
Elastic Volume Model

Extrinsic Volumetric “As-Rigid-As Possible”

- Embed source surface S_1 in volume
- f should preserve 3×3 metric tensor (least squares)

$$E^{(regularizer)}(f) = \int_{V_1} \left[\boxed{\nabla f \, \nabla f^{\mathrm{T}}} - \mathbf{I} \right]^2 dx$$

first fundamental form $\mathbf{I}$ ($\mathbb{R}^{3\times3}$)



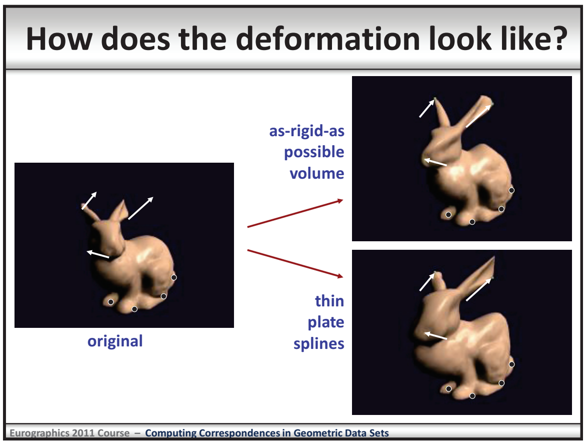
Volume Model

Variant: Thin-Plate-Splines

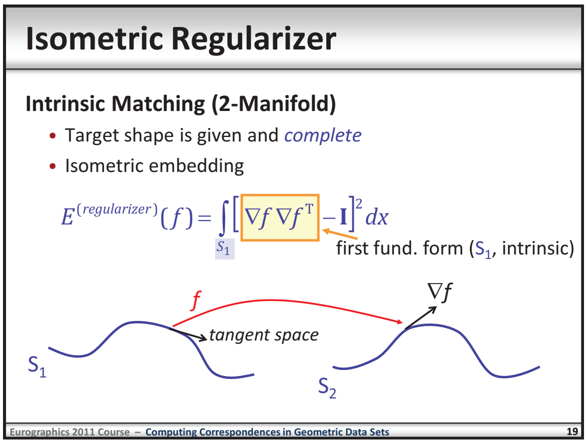
- Use regularizer that penalizes curved deformation

$$E^{(regularizer)}(f) = \int_{V_1} \boxed{H_f(x)}^2 dx$$

second derivative ($\mathbb{R}^{3\times3}$)



How does the deformation look like?



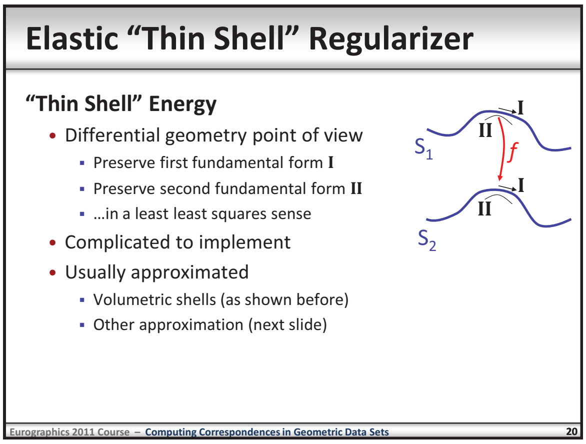
Isometric Regularizer

Intrinsic Matching (2-Manifold)

- Target shape is given and *complete*
- Isometric embedding

$$E^{(regularizer)}(f) = \int_{S_1} \left[\boxed{\nabla f \nabla f^{\mathrm{T}}} - \mathbf{I} \right]^2 dx$$

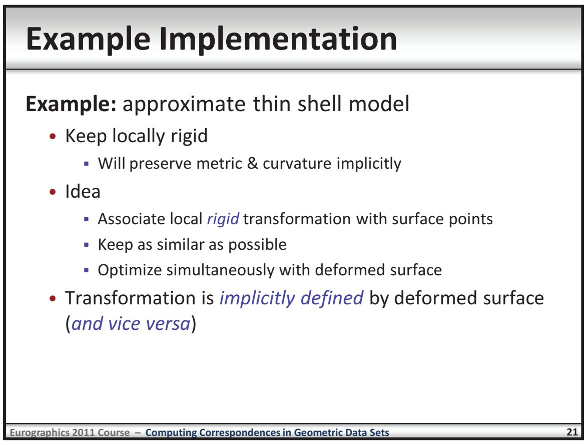
first fund. form (S_1, intrinsic)



Elastic "Thin Shell" Regularizer

"Thin Shell" Energy

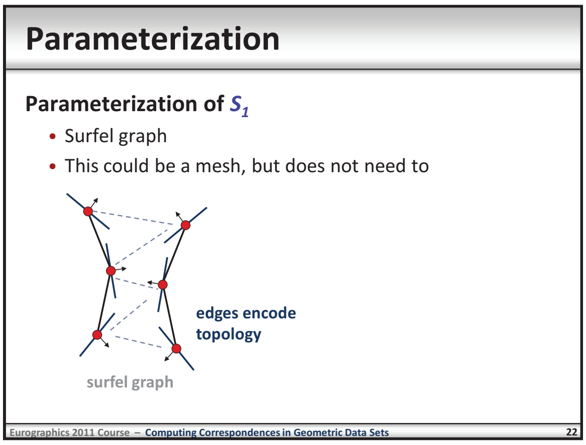
- Differential geometry point of view
  - Preserve first fundamental form **I**
  - Preserve second fundamental form **II**
  - …in a least least squares sense
- Complicated to implement
- Usually approximated
  - Volumetric shells (as shown before)
  - Other approximation (next slide)



Example Implementation

Example: approximate thin shell model

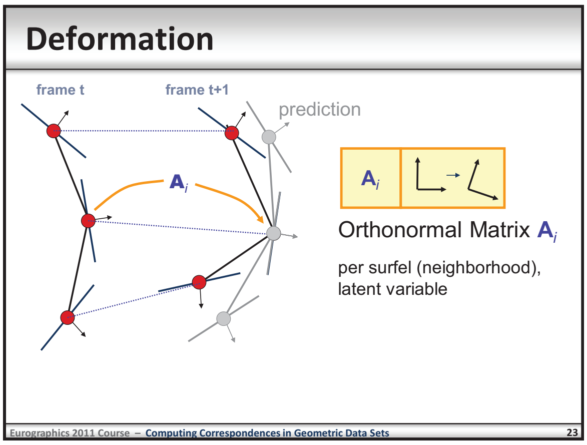
- Keep locally rigid
  - Will preserve metric & curvature implicitly
- Idea
  - Associate local *rigid* transformation with surface points
  - Keep as similar as possible
  - Optimize simultaneously with deformed surface
- Transformation is *implicitly defined* by deformed surface (*and vice versa*)



Parameterization

Parameterization of *S_1*

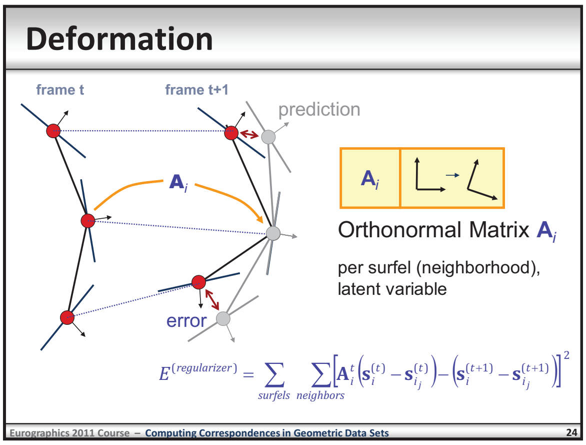
- Surfel graph
- This could be a mesh, but does not need to



Deformation

Orthonormal Matrix $\mathbf{A}_i$

per surfel (neighborhood), latent variable

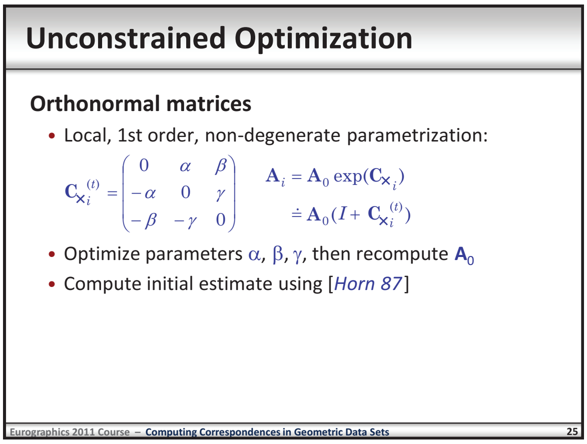


Deformation

Orthonormal Matrix $\mathbf{A}_i$

per surfel (neighborhood), latent variable

$$E^{(regularizer)} = \sum_{surfels} \sum_{neighbors} \left[\mathbf{A}_i^t \left(\mathbf{s}_i^{(t)} - \mathbf{s}_{i_j}^{(t)} \right) - \left(\mathbf{s}_i^{(t+1)} - \mathbf{s}_{i_j}^{(t+1)} \right) \right]^2$$



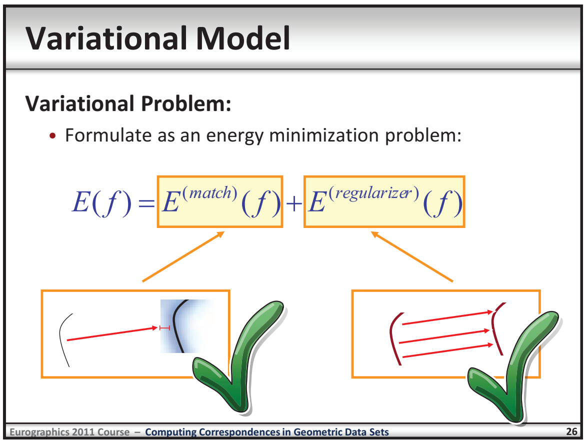
Unconstrained Optimization

Orthonormal matrices

- Local, 1st order, non-degenerate parametrization:

$$\mathbf{C}_{\times_i}^{(t)} = \begin{pmatrix} 0 & \alpha & \beta \\ -\alpha & 0 & \gamma \\ -\beta & -\gamma & 0 \end{pmatrix} \qquad \begin{aligned} \mathbf{A}_i &= \mathbf{A}_0 \exp(\mathbf{C}_{\times_i}) \\ &\doteq \mathbf{A}_0 (I + \mathbf{C}_{\times_i}^{(t)}) \end{aligned}$$

- Optimize parameters α, β, γ, then recompute $\mathbf{A}_0$
- Compute initial estimate using [*Horn 87*]

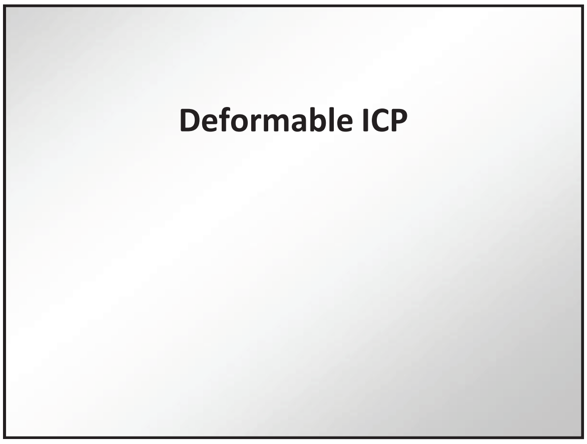


Variational Model

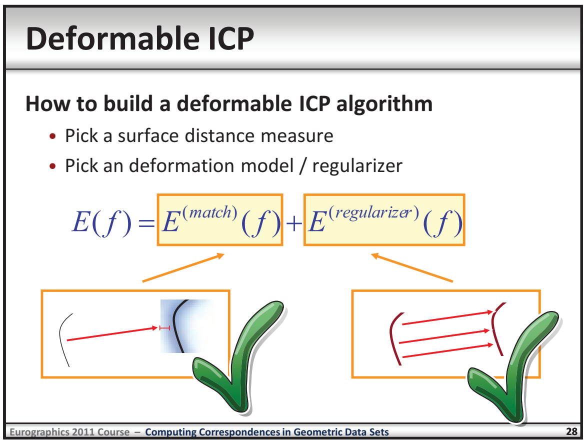
Variational Problem:

- Formulate as an energy minimization problem:

$$E(f) = \boxed{E^{(match)}(f)} + \boxed{E^{(regularizer)}(f)}$$



Deformable ICP

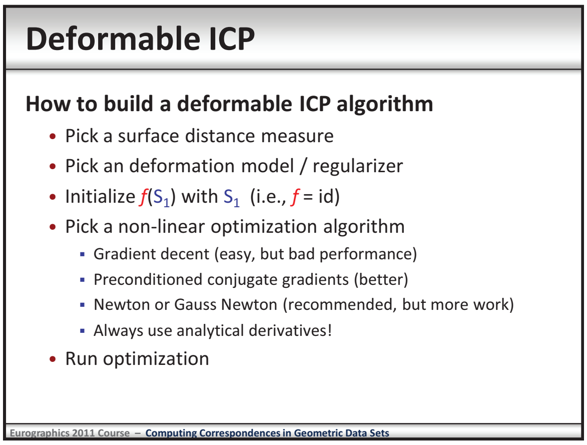


Deformable ICP

How to build a deformable ICP algorithm

- Pick a surface distance measure
- Pick an deformation model / regularizer

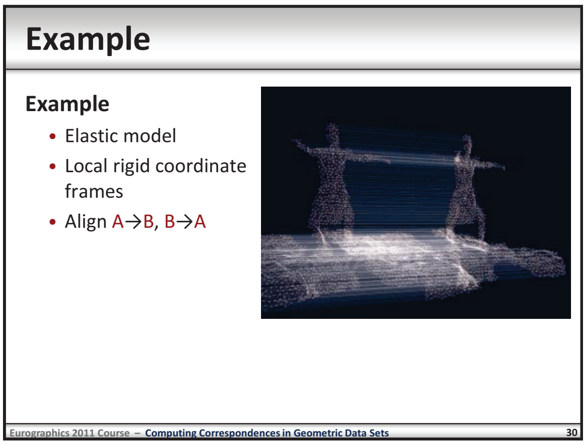
$$E(f) = \boxed{E^{(match)}(f)} + \boxed{E^{(regularizer)}(f)}$$



Deformable ICP

How to build a deformable ICP algorithm

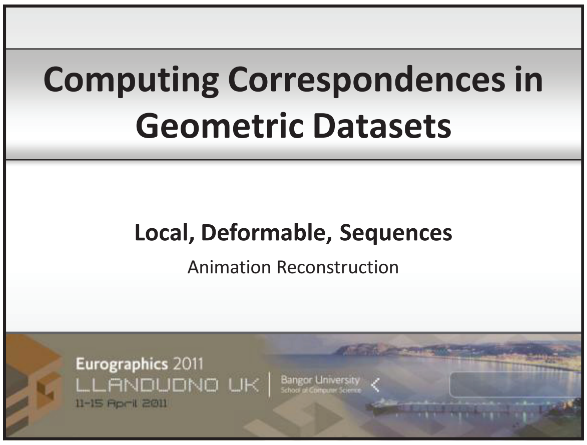
- Pick a surface distance measure
- Pick an deformation model / regularizer
- Initialize $f(S_1)$ with S_1 (i.e., f = id)
- Pick a non-linear optimization algorithm
  - Gradient decent (easy, but bad performance)
  - Preconditioned conjugate gradients (better)
  - Newton or Gauss Newton (recommended, but more work)
  - Always use analytical derivatives!
- Run optimization



Example

Example

- Elastic model
- Local rigid coordinate frames
- Align A→B, B→A

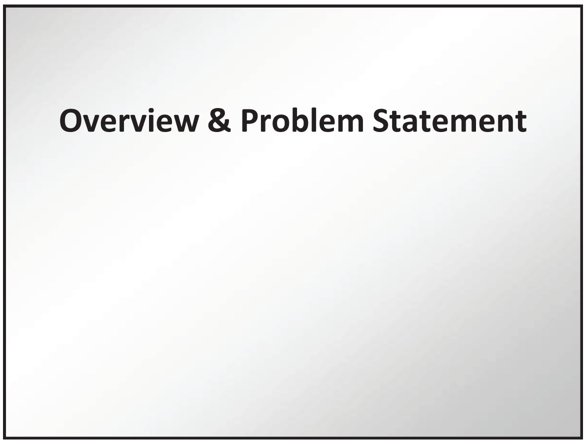


Computing Correspondences in Geometric Datasets

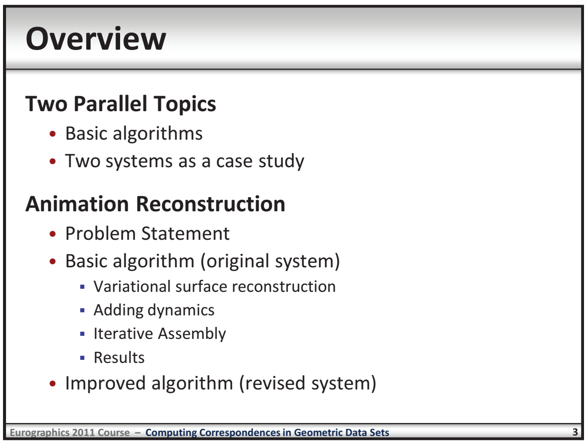
Local, Deformable, Sequences

Animation Reconstruction

Eurographics 2011
LLANDUDNO UK
11-15 April 2011
Bangor University
School of Computer Science



Overview & Problem Statement



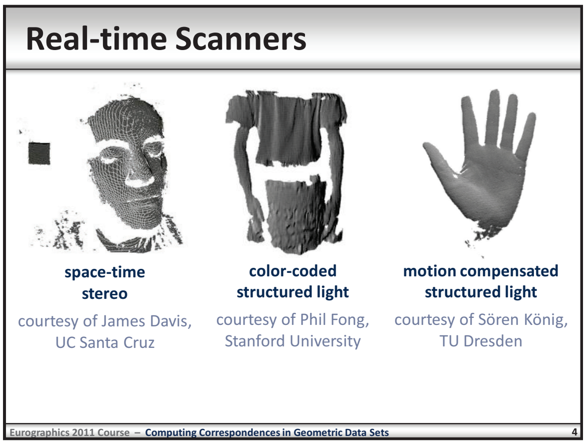
Overview

Two Parallel Topics

- Basic algorithms
- Two systems as a case study

Animation Reconstruction

- Problem Statement
- Basic algorithm (original system)
  - Variational surface reconstruction
  - Adding dynamics
  - Iterative Assembly
  - Results
- Improved algorithm (revised system)



Real-time Scanners

space-time stereo

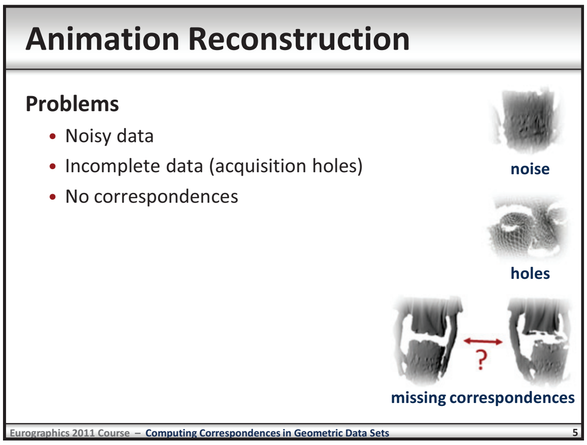
courtesy of James Davis, UC Santa Cruz

color-coded structured light

courtesy of Phil Fong, Stanford University

motion compensated structured light

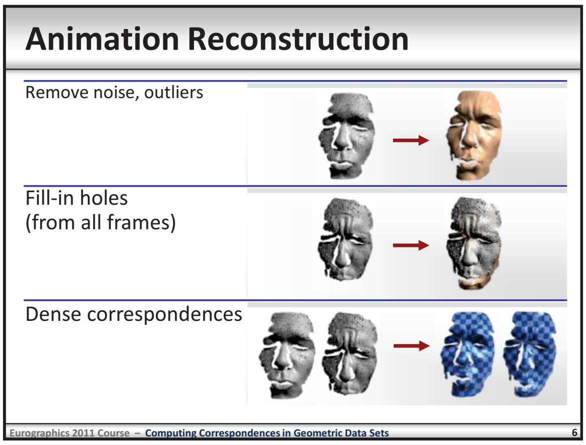
courtesy of Sören König, TU Dresden



Animation Reconstruction

Problems

- Noisy data
- Incomplete data (acquisition holes)
- No correspondences

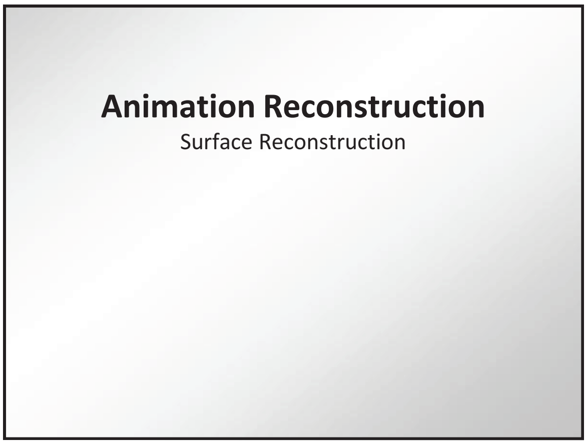


Animation Reconstruction

Remove noise, outliers

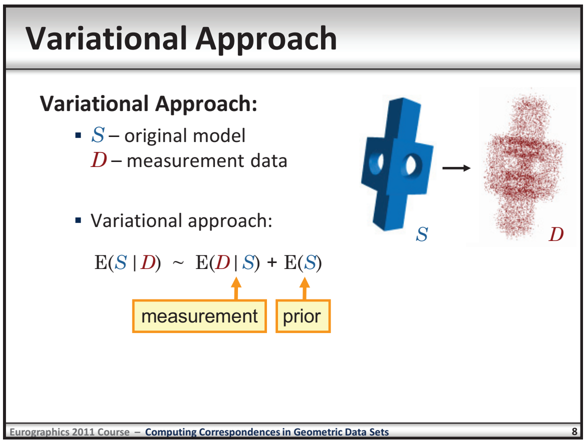
Fill-in holes
(from all frames)

Dense correspondences



Animation Reconstruction

Surface Reconstruction



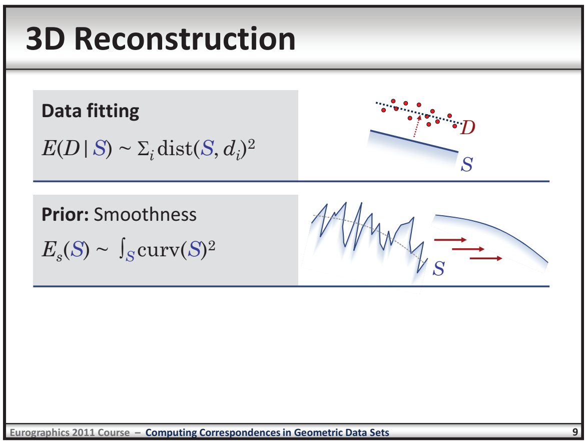
Variational Approach

Variational Approach:

- S – original model
  D – measurement data
- Variational approach:

$$\mathrm{E}(S\,|\,D) \;\sim\; \mathrm{E}(D\,|\,S) + \mathrm{E}(S)$$

measurement prior



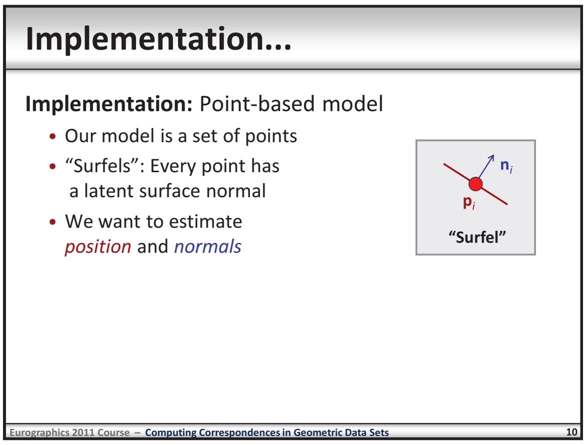
3D Reconstruction

Data fitting

$E(D\,|\,S) \sim \Sigma_i\, \mathrm{dist}(S, d_i)^2$

Prior: Smoothness

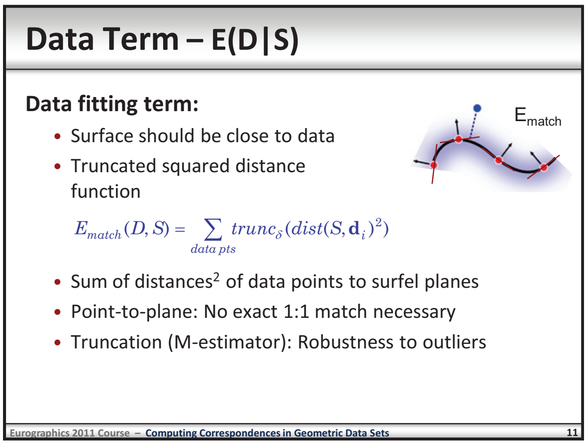
$E_s(S) \sim \int_S \mathrm{curv}(S)^2$



Implementation...

Implementation: Point-based model

- Our model is a set of points
- "Surfels": Every point has a latent surface normal
- We want to estimate *position* and *normals*



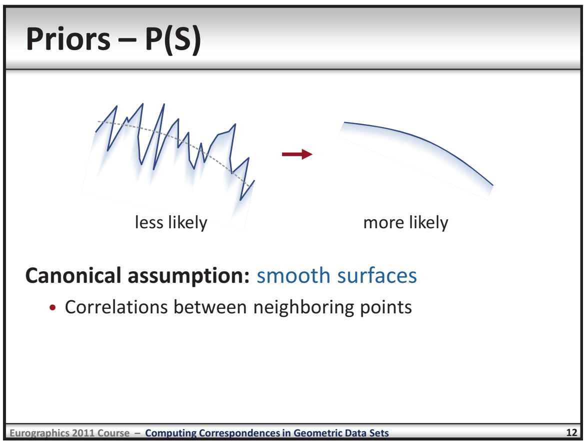
Data Term – E(D|S)

Data fitting term:

- Surface should be close to data
- Truncated squared distance function

$$E_{match}(D,S) = \sum_{data\ pts} trunc_{\delta}(dist(S,\mathbf{d}_i)^2)$$

- Sum of distances^2 of data points to surfel planes
- Point-to-plane: No exact 1:1 match necessary
- Truncation (M-estimator): Robustness to outliers



Priors – P(S)

Canonical assumption: smooth surfaces

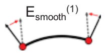
- Correlations between neighboring points

Point-based Model

Simple Smoothness Priors:

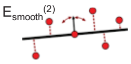
- Similar surfel normals:

$$E^{(1)}_{smooth}(S) = \sum_{surfels} \sum_{neighbors} \left(n_i - n_{i_j}\right)^2, \ \|n_i\| = 1$$



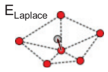
- Surfel positions – flat surface:

$$E^{(2)}_{smooth}(S) = \sum_{surfels} \sum_{neighbors} \left\langle \mathbf{s}_i - \mathbf{s}_{i_j}, \mathbf{n}(\mathbf{s}_i) \right\rangle^2$$



- Uniform density:

$$E_{Laplace}(S) = \sum_{surfels} \sum_{neighbors} \left(\mathbf{s}_i - average\right)^2$$



[c.f. Szeliski et al. 93]

Nasty Normals

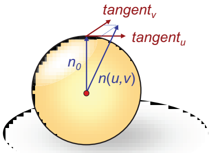
Optimizing Normals

- Problem: $E^{(1)}_{smooth}(S) = \sum_{surfels} \sum_{neighbors} \left(n_i - n_{i_j}\right)^2, \ s.t. \|n_i\| = 1$
- Need unit normals: constraint optimization
- Unconstraint: trivial solution (all zeros)

Nasty Normals

Solution: Local Parameterization

- Current normal estimate
- Tangent parameterization
- New variables *u*, *v*
- Renormalize
- Non-linear optimization
- No degeneracies



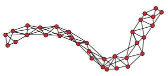
$$n(u,v) = n_0 + u \cdot tangent_u + v \cdot tangent_v$$

[Hoffer et al. 04]

Neighborhoods?

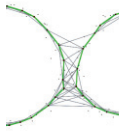
Topology estimation

- Domain of *S*, base shape (topology)
- Here, we assume this is easy to get
- In the following
  - *k*-nearest neighborhood graph
  - Typically: *k* = 6..20



Limitations

- This requires dense enough sampling
- Does not work for undersampled data



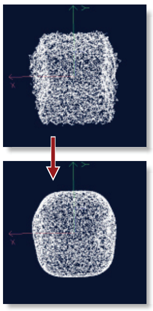
Numerical Optimization

Task:

- Compute most likely “original scene” S
- Nonlinear optimization problem

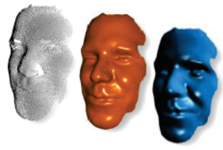
Solution:

- Create initial guess for S
  - Close to measured data
  - Use original data
- Find local optimum
  - (Conjugate) gradient descent
  - (Gauss-) Newton descent

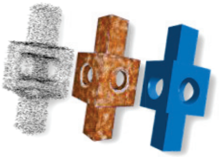


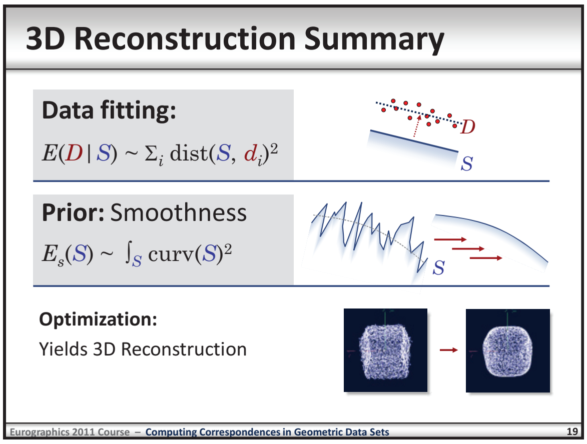
3D Examples

3D reconstruction results:



(With discontinuity lines, not used here):





3D Reconstruction Summary

Data fitting:

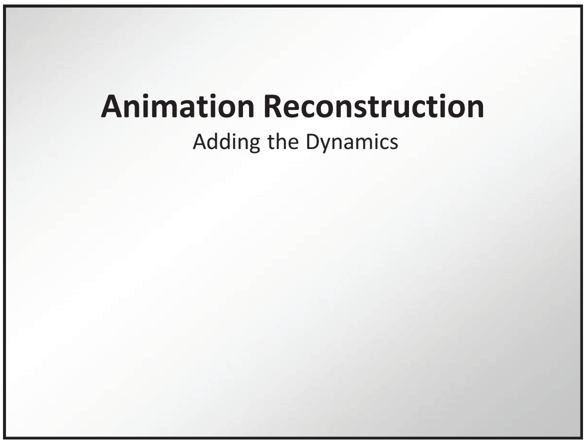
$E(D\,|\,S) \sim \Sigma_i \operatorname{dist}(S, d_i)^2$

Prior: Smoothness

$E_s(S) \sim \int_S \operatorname{curv}(S)^2$

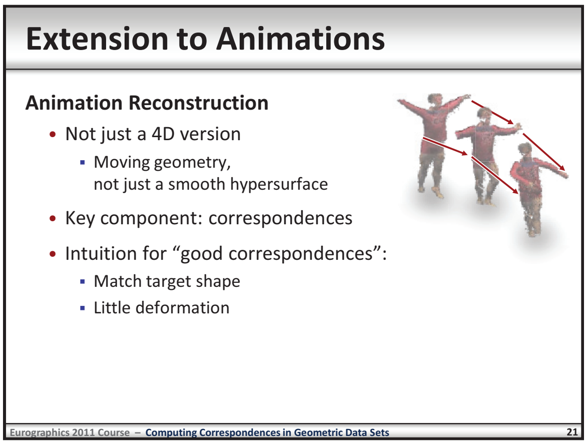
Optimization:

Yields 3D Reconstruction



Animation Reconstruction

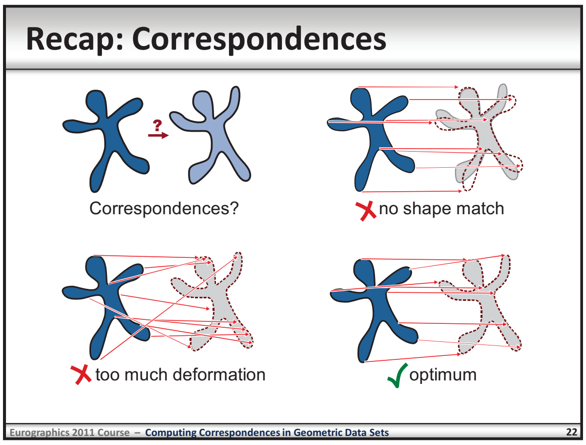
Adding the Dynamics



Extension to Animations

Animation Reconstruction

- Not just a 4D version
  - Moving geometry, not just a smooth hypersurface
- Key component: correspondences
- Intuition for “good correspondences”:
  - Match target shape
  - Little deformation



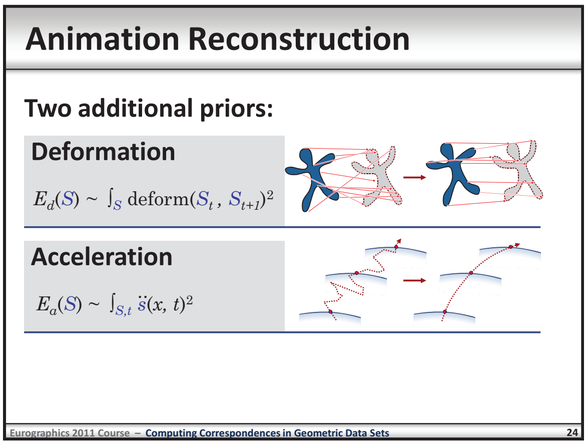
Recap: Correspondences

Correspondences?

✗ no shape match

✗ too much deformation

✓ optimum



Animation Reconstruction

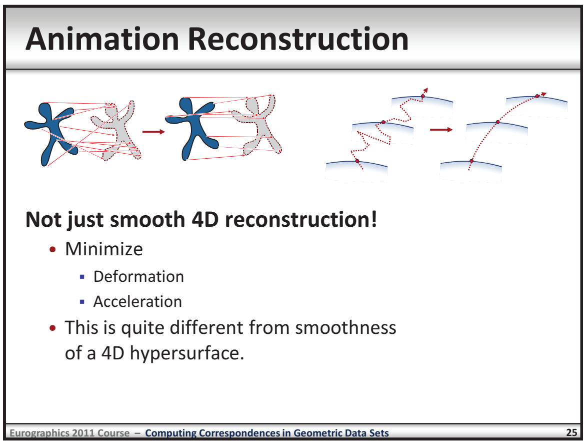
Two additional priors:

Deformation

$E_d(S) \sim \int_S \operatorname{deform}(S_t, S_{t+1})^2$

Acceleration

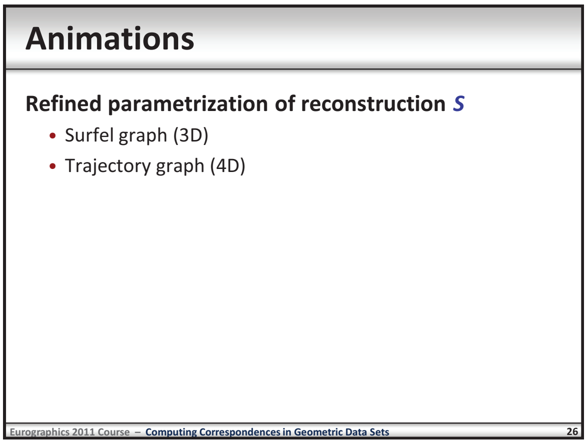
$E_a(S) \sim \int_{S,t} \ddot{s}(x, t)^2$



Animation Reconstruction

Not just smooth 4D reconstruction!

- Minimize
  - Deformation
  - Acceleration
- This is quite different from smoothness of a 4D hypersurface.

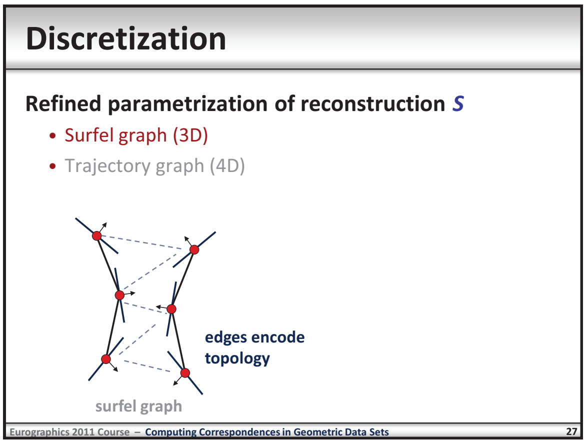


Animations

Refined parametrization of reconstruction *S*

- Surfel graph (3D)
- Trajectory graph (4D)

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 26



Discretization

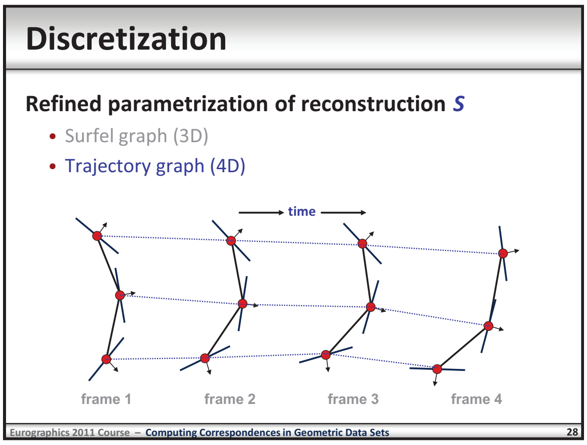
Refined parametrization of reconstruction *S*

- Surfel graph (3D)
- Trajectory graph (4D)

edges encode topology

surfel graph

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 27



Discretization

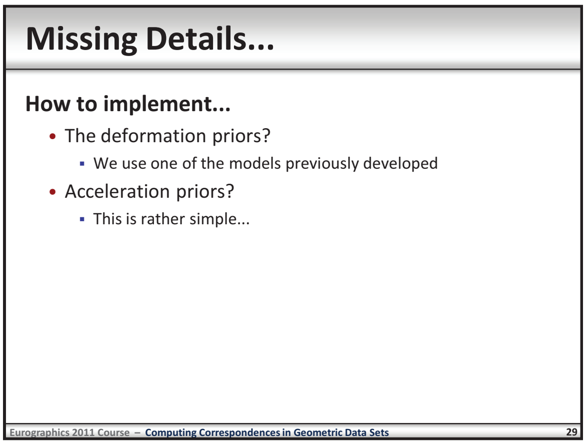
Refined parametrization of reconstruction *S*

- Surfel graph (3D)
- Trajectory graph (4D)

time

frame 1 frame 2 frame 3 frame 4

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 28

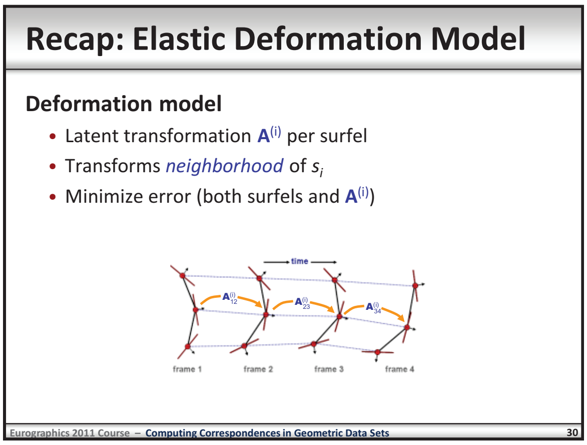


Missing Details...

How to implement...

- The deformation priors?
  - We use one of the models previously developed
- Acceleration priors?
  - This is rather simple...

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 29



Recap: Elastic Deformation Model

Deformation model

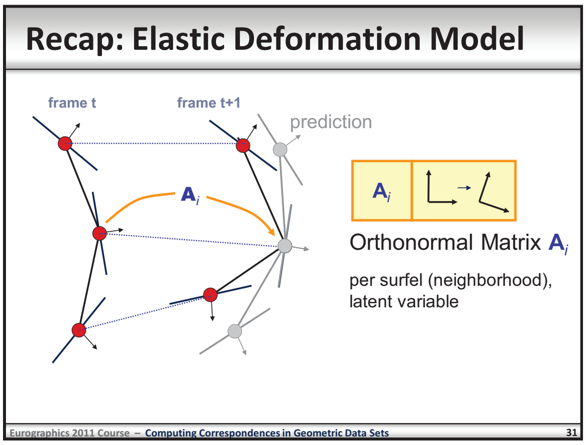
- Latent transformation $\mathbf{A}^{(i)}$ per surfel
- Transforms *neighborhood* of s_i
- Minimize error (both surfels and $\mathbf{A}^{(i)}$)

time

$\mathbf{A}^{(i)}_{12}$ $\mathbf{A}^{(i)}_{23}$ $\mathbf{A}^{(i)}_{34}$

frame 1 frame 2 frame 3 frame 4

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 30



Recap: Elastic Deformation Model

frame t frame t+1

prediction

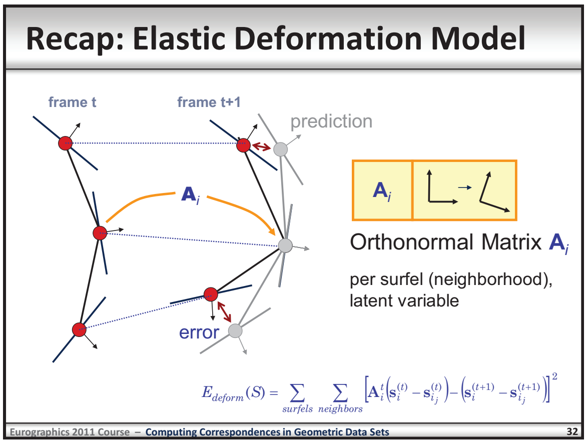
$\mathbf{A}_i$

$\mathbf{A}_i$

Orthonormal Matrix $\mathbf{A}_i$

per surfel (neighborhood), latent variable

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 31

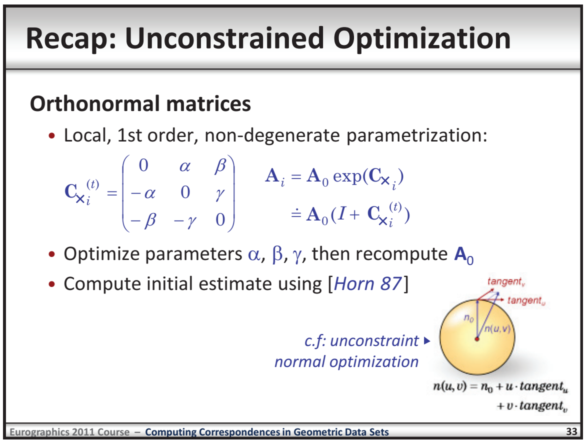


Recap: Elastic Deformation Model

Orthonormal Matrix $\mathbf{A}_i$

per surfel (neighborhood), latent variable

$$E_{deform}(S) = \sum_{surfels} \sum_{neighbors} \left[\mathbf{A}_i^t\left(\mathbf{s}_i^{(t)} - \mathbf{s}_{i_j}^{(t)}\right) - \left(\mathbf{s}_i^{(t+1)} - \mathbf{s}_{i_j}^{(t+1)}\right)\right]^2$$



Recap: Unconstrained Optimization

Orthonormal matrices

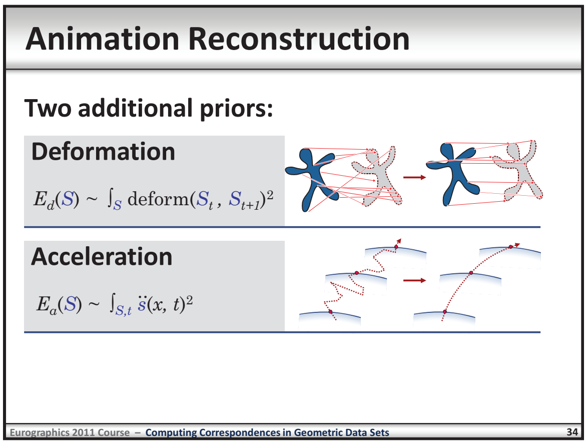
- Local, 1st order, non-degenerate parametrization:

$$\mathbf{C}_{\times_i}^{(t)} = \begin{pmatrix} 0 & \alpha & \beta \\ -\alpha & 0 & \gamma \\ -\beta & -\gamma & 0 \end{pmatrix} \qquad \begin{aligned} \mathbf{A}_i &= \mathbf{A}_0 \exp(\mathbf{C}_{\times_i}) \\ &\doteq \mathbf{A}_0 (I + \mathbf{C}_{\times_i}^{(t)}) \end{aligned}$$

- Optimize parameters α, β, γ, then recompute $\mathbf{A}_0$
- Compute initial estimate using [*Horn 87*]

c.f: unconstraint normal optimization ▸

$$n(u,v) = n_0 + u \cdot tangent_u + v \cdot tangent_v$$



Animation Reconstruction

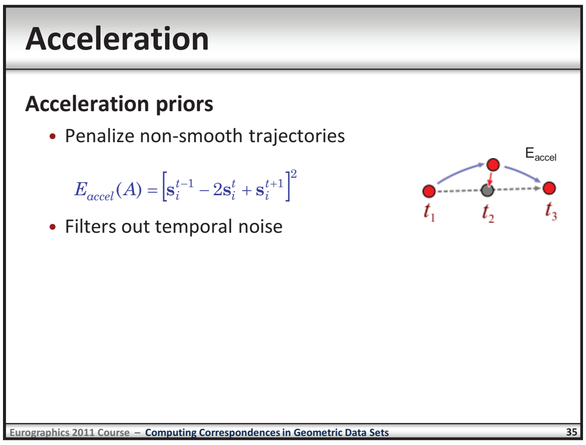
Two additional priors:

Deformation

$E_d(S) \sim \int_S \text{deform}(S_t\,,\, S_{t+1})^2$

Acceleration

$E_a(S) \sim \int_{S,t} \ddot{s}(x,\, t)^2$



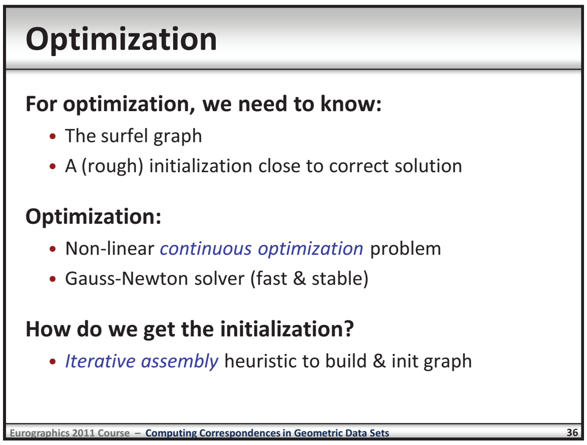
Acceleration

Acceleration priors

- Penalize non-smooth trajectories

$$E_{accel}(A) = \left[\mathbf{s}_i^{t-1} - 2\mathbf{s}_i^t + \mathbf{s}_i^{t+1}\right]^2$$

- Filters out temporal noise



Optimization

For optimization, we need to know:

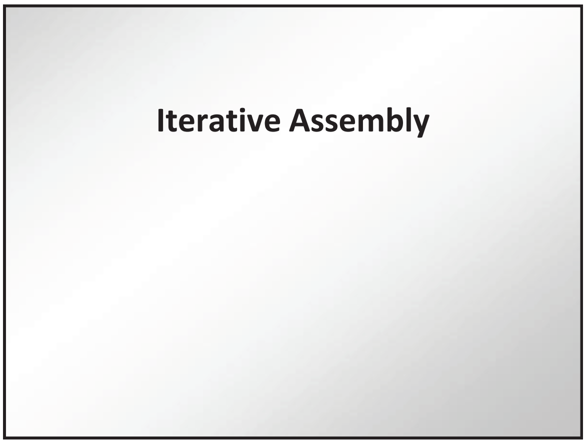
- The surfel graph
- A (rough) initialization close to correct solution

Optimization:

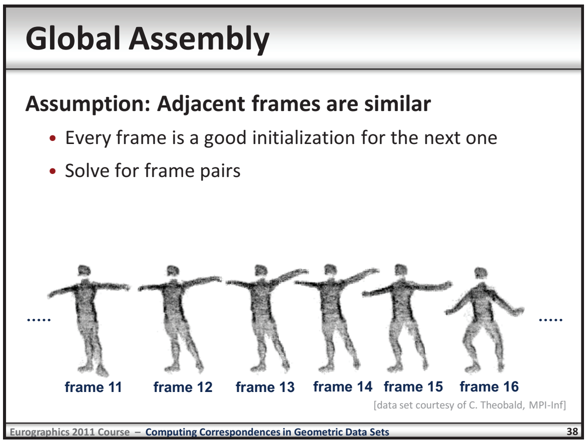
- Non-linear *continuous optimization* problem
- Gauss-Newton solver (fast & stable)

How do we get the initialization?

- *Iterative assembly* heuristic to build & init graph



Iterative Assembly

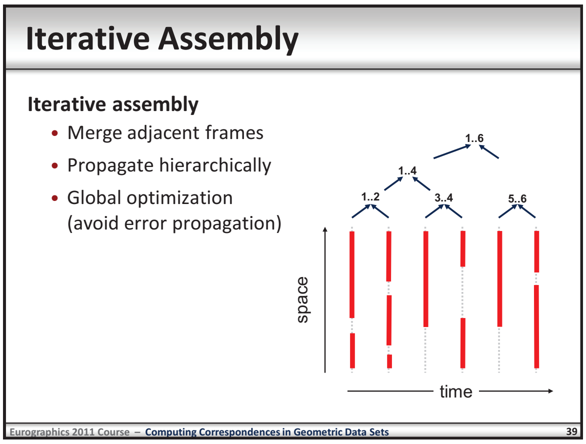


Global Assembly

Assumption: Adjacent frames are similar

- Every frame is a good initialization for the next one
- Solve for frame pairs

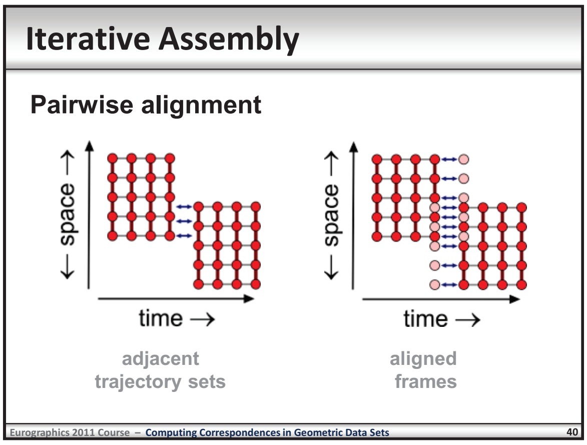
[data set courtesy of C. Theobald, MPI-Inf]



Iterative Assembly

Iterative assembly

- Merge adjacent frames
- Propagate hierarchically
- Global optimization (avoid error propagation)

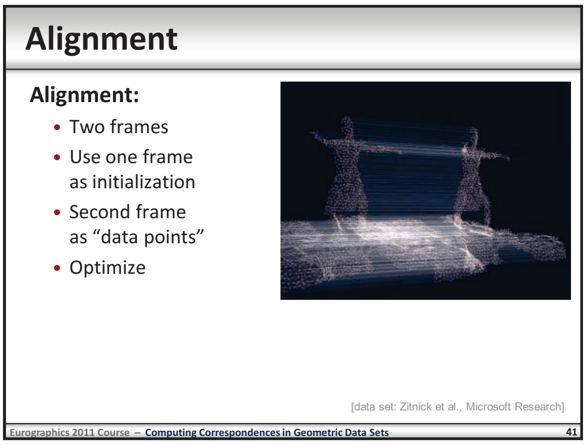


Iterative Assembly

Pairwise alignment

adjacent trajectory sets

aligned frames

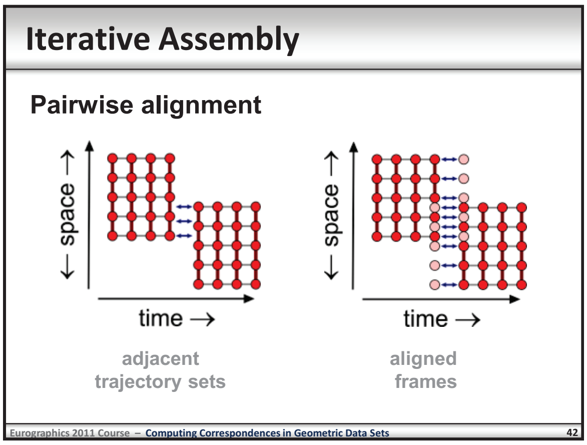


Alignment

Alignment:

- Two frames
- Use one frame as initialization
- Second frame as “data points”
- Optimize

[data set: Zitnick et al., Microsoft Research]

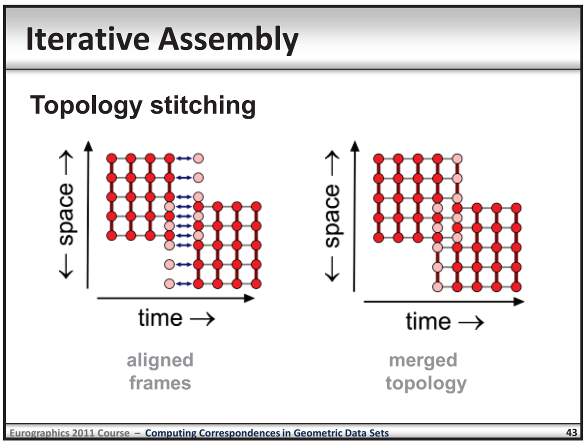


Iterative Assembly

Pairwise alignment

adjacent trajectory sets

aligned frames

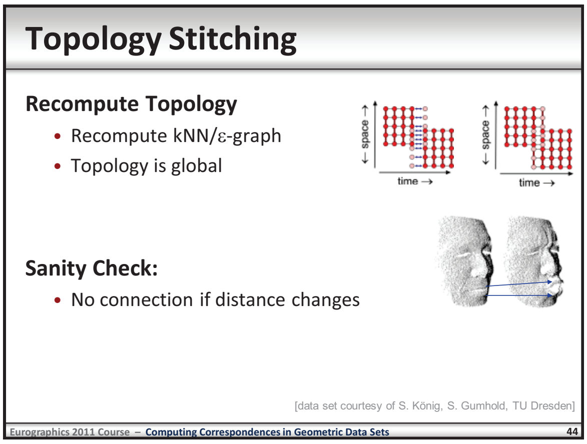


Iterative Assembly

Topology stitching

aligned frames

merged topology



Topology Stitching

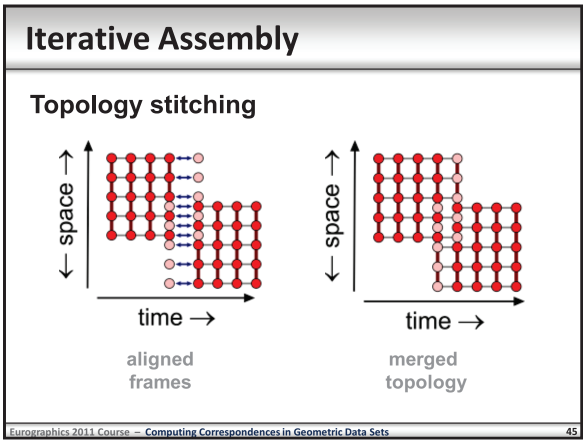
Recompute Topology

- Recompute kNN/ε-graph
- Topology is global

Sanity Check:

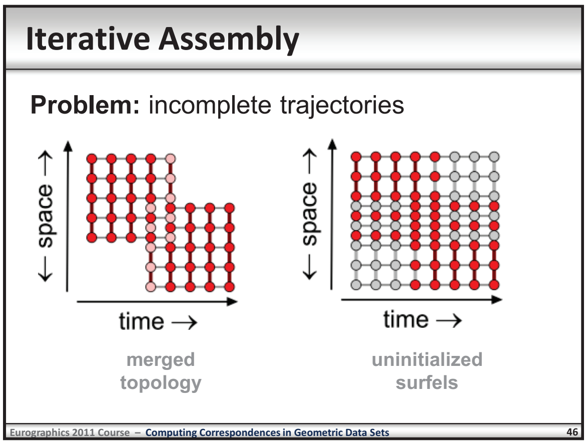
- No connection if distance changes

[data set courtesy of S. König, S. Gumhold, TU Dresden]



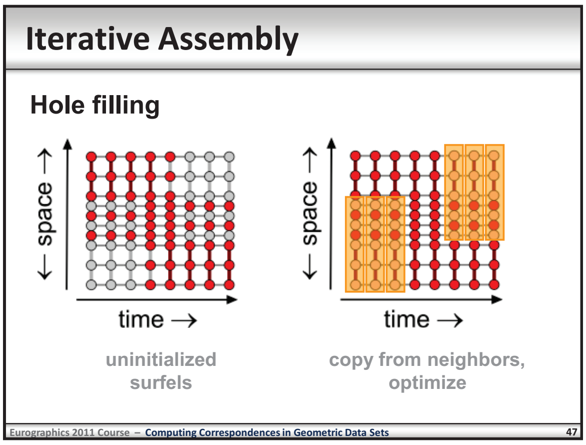
Iterative Assembly

Topology stitching



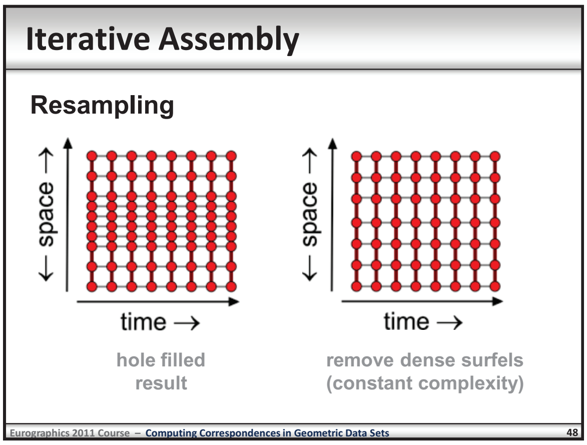
Iterative Assembly

Problem: incomplete trajectories



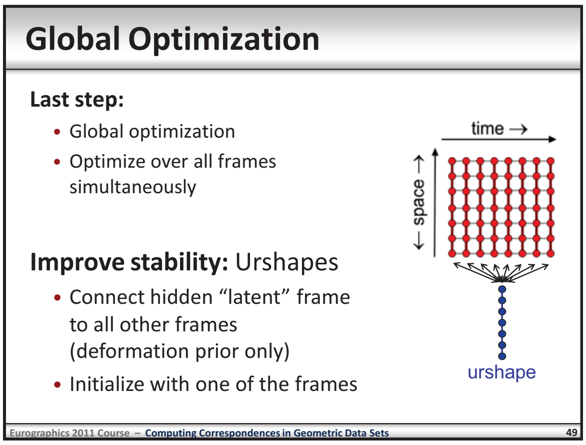
Iterative Assembly

Hole filling



Iterative Assembly

Resampling



Global Optimization

Last step:

- Global optimization
- Optimize over all frames simultaneously

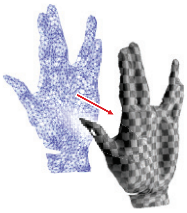
Improve stability: Urshapes

- Connect hidden "latent" frame to all other frames (deformation prior only)
- Initialize with one of the frames

Meshing

Last step: create mesh

- After complete surfel graph is reconstructed
- Pick one frame (or urshape)
- “Marching cubes” meshing [*Hoppe et al. 92*, *Shen et al. 04*]
- Morph according to trajectories (local weighted sum)



[data set courtesy of O. Schall, MPI Informatik Saarbrücken]

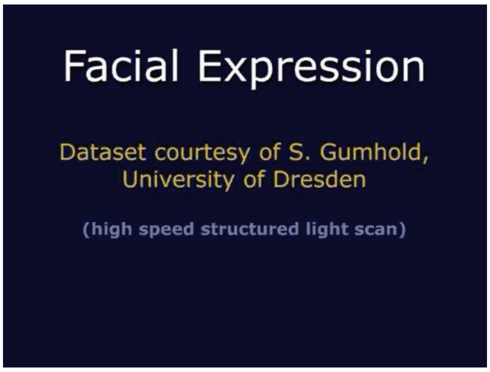
Results

Elephant

deformation & rotation, noise, outliers, large holes

(synthetic data)

frames	surfels	data pts	preprocessing	reconstruction	
20	49,500	963,671	6 min 52 sec	4 h 25 min	[Pentium-4, 3.4GHz]



Facial Expression

Dataset courtesy of S. Gumhold, University of Dresden

(high speed structured light scan)

frames	surfels	data pts	preprocessing	reconstruction	
20	32,740	400,000	6 min 59 sec(*)	7 h 31 min	[Pentium-4, 3.4GHz / (*)3.0GHz]

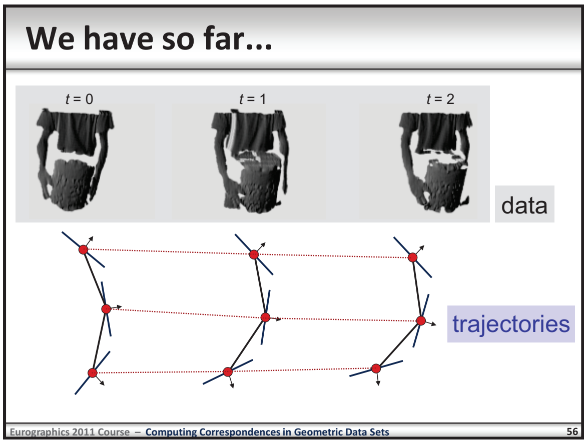
Improved Algorithm

Urshape Factorization

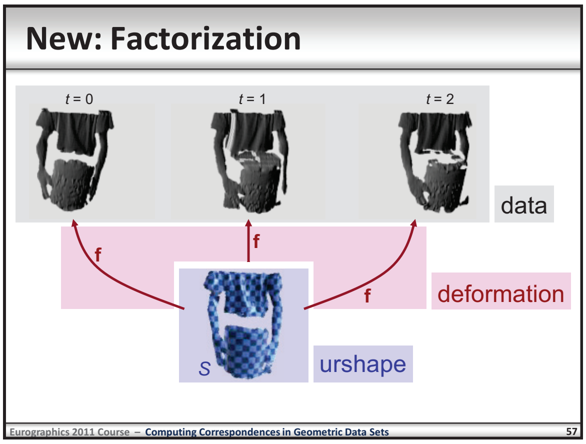
Improved Version

Factorization Model:

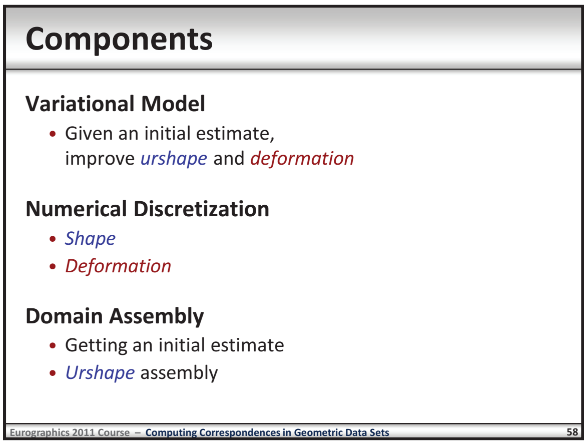
- Solving for the geometry in every frame wastes resources
- Store one urshape and a deformation field
  - High resolution geometry
  - Low resolution deformation (adaptive)
- Less memory, faster, and much more stable
- Streaming computation (constant working set)



We have so far…



New: Factorization



Components

Variational Model

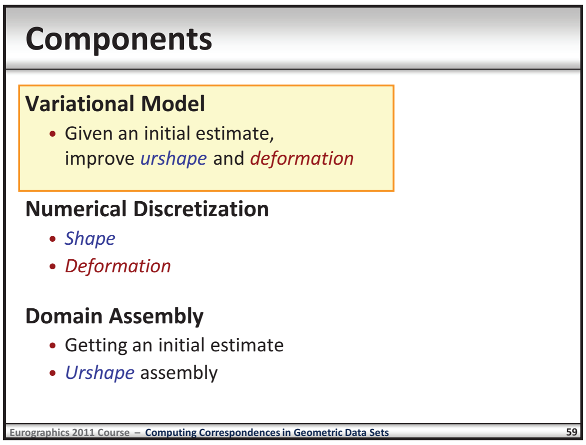
- Given an initial estimate, improve *urshape* and *deformation*

Numerical Discretization

- *Shape*
- *Deformation*

Domain Assembly

- Getting an initial estimate
- *Urshape* assembly



Components

Variational Model

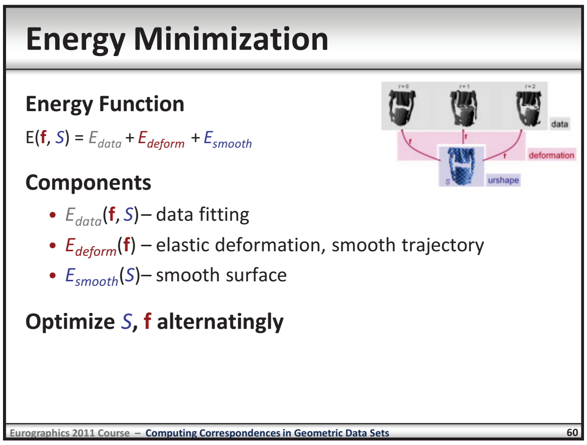
- Given an initial estimate, improve *urshape* and *deformation*

Numerical Discretization

- *Shape*
- *Deformation*

Domain Assembly

- Getting an initial estimate
- *Urshape* assembly



Energy Minimization

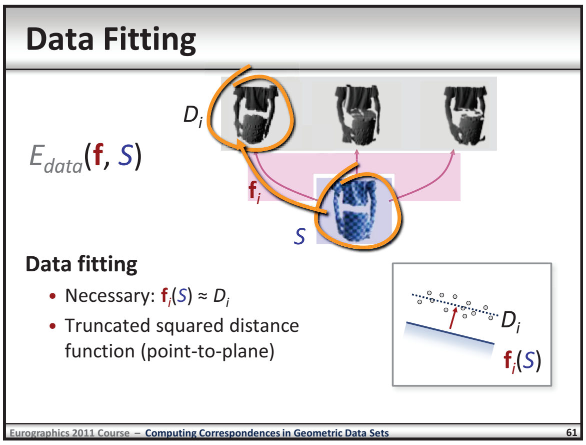
Energy Function

$E(\mathbf{f}, S) = E_{data} + E_{deform} + E_{smooth}$

Components

- $E_{data}(\mathbf{f}, S)$ – data fitting
- $E_{deform}(\mathbf{f})$ – elastic deformation, smooth trajectory
- $E_{smooth}(S)$ – smooth surface

Optimize S, $\mathbf{f}$ alternatingly

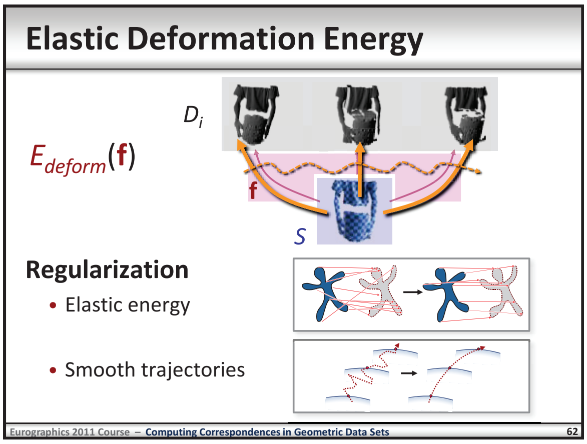


Data Fitting

$E_{data}(\mathbf{f}, S)$

Data fitting

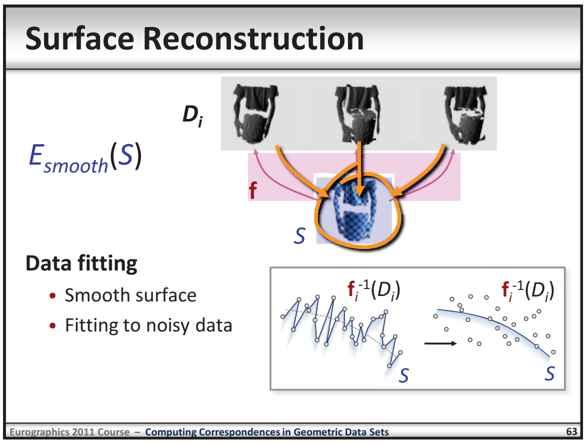
- Necessary: $\mathbf{f}_i(S) \approx D_i$
- Truncated squared distance function (point-to-plane)



Elastic Deformation Energy

Regularization

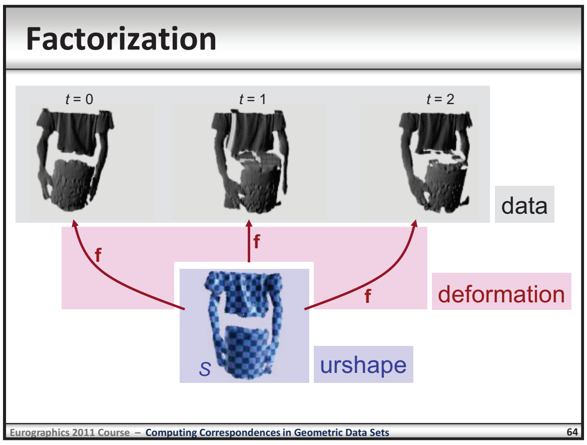
- Elastic energy
- Smooth trajectories



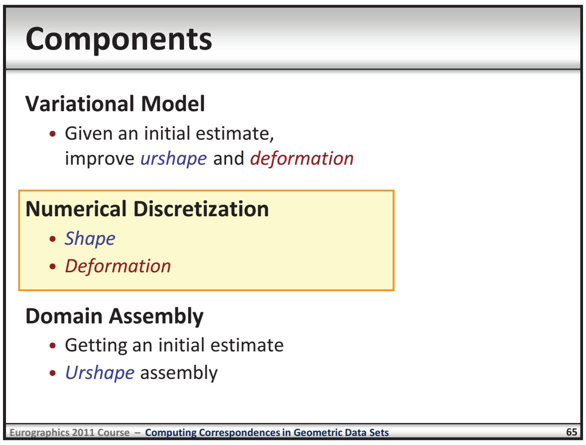
Surface Reconstruction

Data fitting

- Smooth surface
- Fitting to noisy data



Factorization



Components

Variational Model

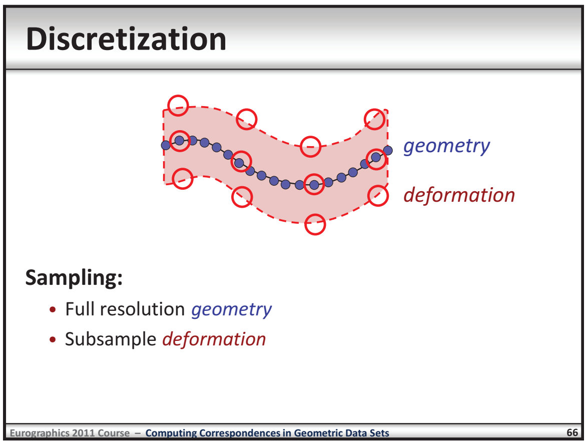
- Given an initial estimate, improve *urshape* and *deformation*

Numerical Discretization

- *Shape*
- *Deformation*

Domain Assembly

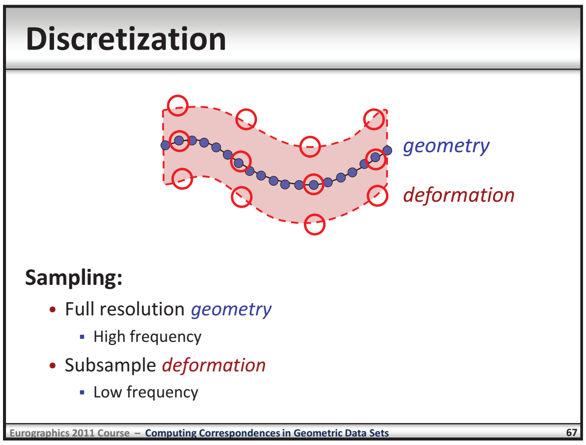
- Getting an initial estimate
- *Urshape* assembly



Discretization

Sampling:

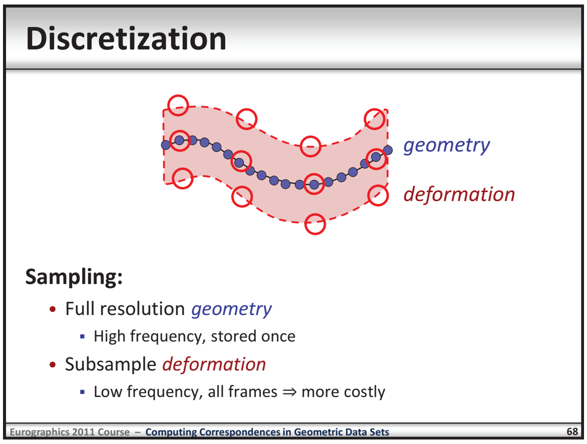
- Full resolution *geometry*
- Subsample *deformation*



Discretization

Sampling:

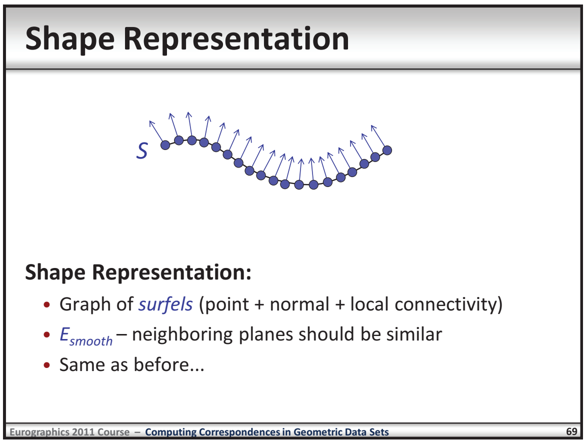
- Full resolution *geometry*
  - High frequency
- Subsample *deformation*
  - Low frequency



Discretization

Sampling:

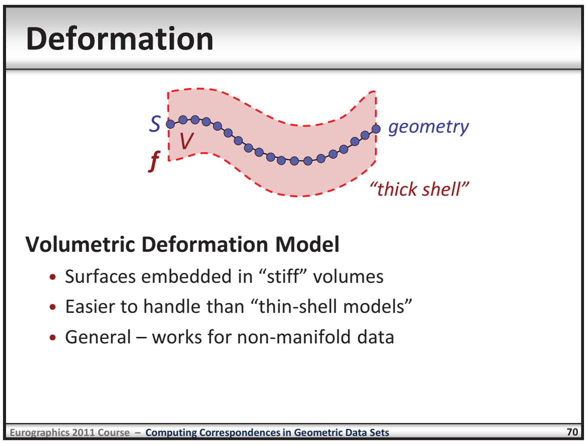
- Full resolution *geometry*
  - High frequency, stored once
- Subsample *deformation*
  - Low frequency, all frames ⇒ more costly



Shape Representation

Shape Representation:

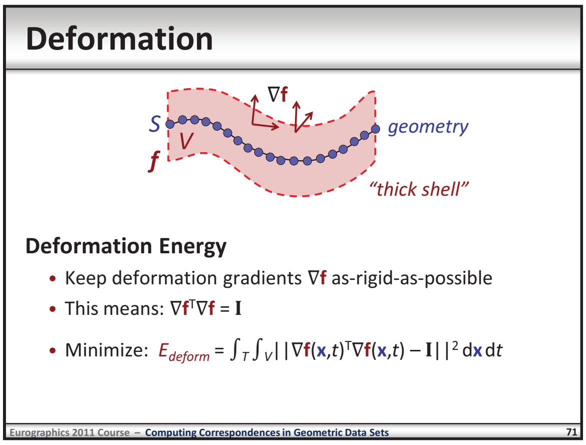
- Graph of *surfels* (point + normal + local connectivity)
- E_{smooth} – neighboring planes should be similar
- Same as before...



Deformation

Volumetric Deformation Model

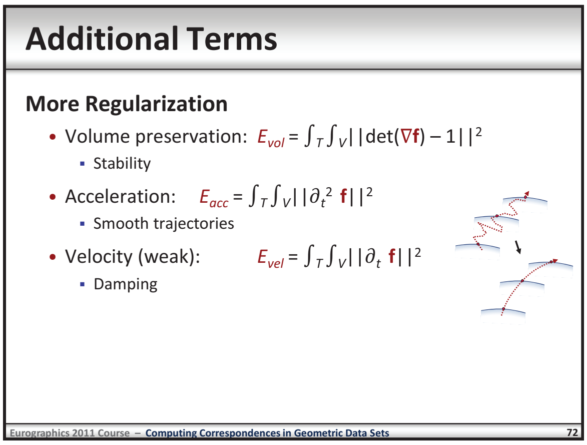
- Surfaces embedded in "stiff" volumes
- Easier to handle than "thin-shell models"
- General – works for non-manifold data



Deformation

Deformation Energy

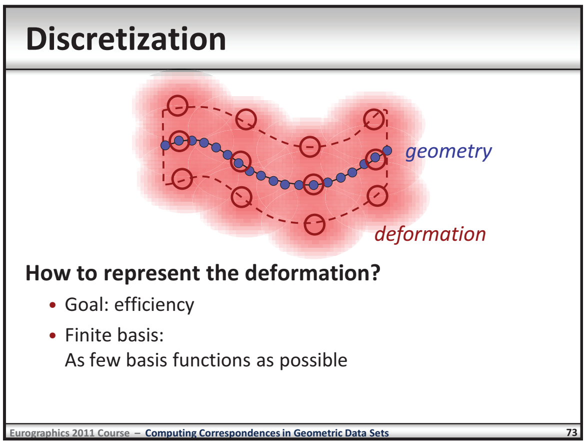
- Keep deformation gradients $\nabla \mathbf{f}$ as-rigid-as-possible
- This means: $\nabla \mathbf{f}^T \nabla \mathbf{f} = \mathbf{I}$
- Minimize: $E_{deform} = \int_T \int_V ||\nabla \mathbf{f}(\mathbf{x},t)^T \nabla \mathbf{f}(\mathbf{x},t) - \mathbf{I}||^2 \, d\mathbf{x} \, dt$



Additional Terms

More Regularization

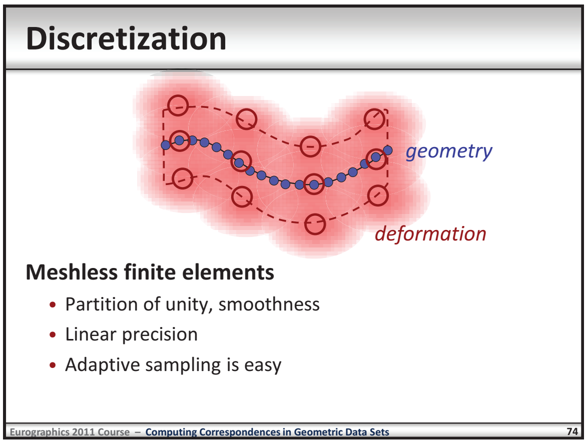
- Volume preservation: $E_{vol} = \int_T \int_V ||\det(\nabla \mathbf{f}) - 1||^2$
  - Stability
- Acceleration: $E_{acc} = \int_T \int_V ||\partial_t^2 \, \mathbf{f}||^2$
  - Smooth trajectories
- Velocity (weak): $E_{vel} = \int_T \int_V ||\partial_t \, \mathbf{f}||^2$
  - Damping



Discretization

How to represent the deformation?

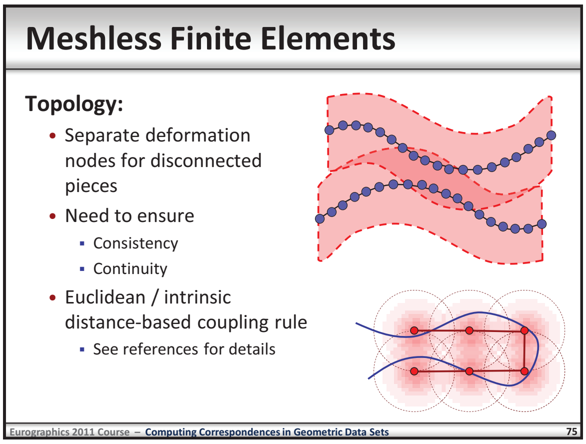
- Goal: efficiency
- Finite basis:
  As few basis functions as possible



Discretization

Meshless finite elements

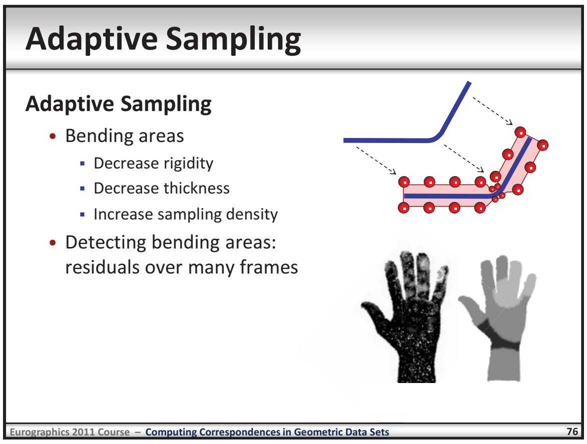
- Partition of unity, smoothness
- Linear precision
- Adaptive sampling is easy



Meshless Finite Elements

Topology:

- Separate deformation nodes for disconnected pieces
- Need to ensure
  - Consistency
  - Continuity
- Euclidean / intrinsic distance-based coupling rule
  - See references for details



Adaptive Sampling

Adaptive Sampling

- Bending areas
  - Decrease rigidity
  - Decrease thickness
  - Increase sampling density
- Detecting bending areas: residuals over many frames

Components

Variational Model

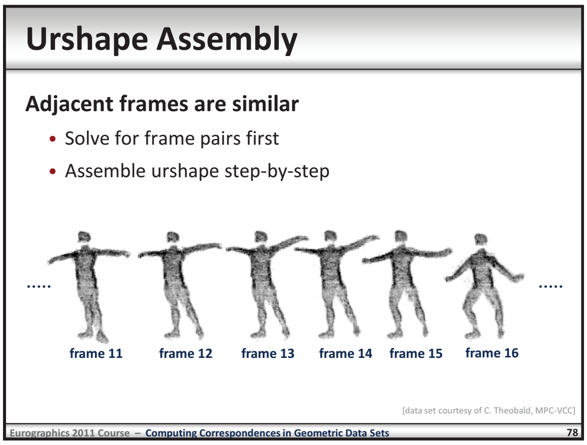
- Given an initial estimate, improve *urshape* and *deformation*

Numerical Discretization

- *Deformation*
- *Shape*

Domain Assembly

- Getting an initial estimate
- *Urshape* assembly

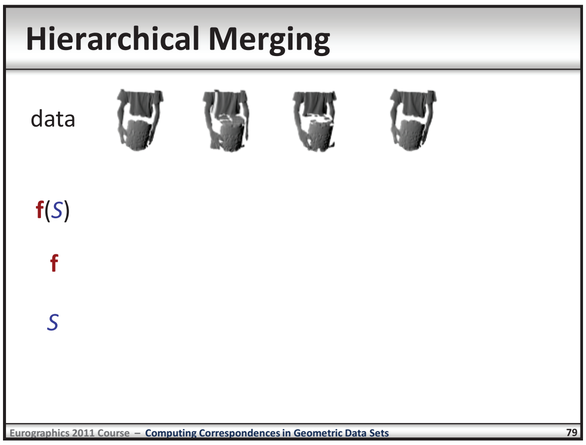


Urshape Assembly

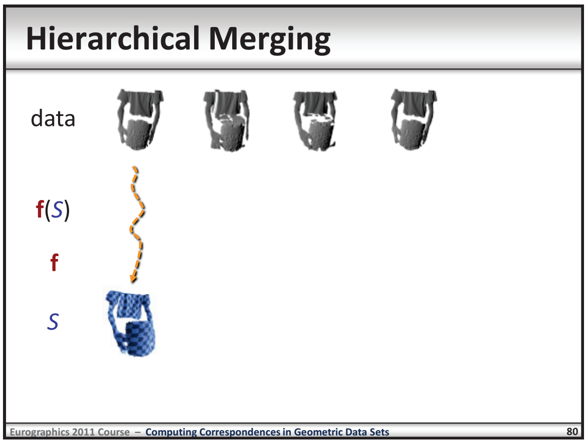
Adjacent frames are similar

- Solve for frame pairs first
- Assemble urshape step-by-step

[data set courtesy of C. Theobald, MPC-VCC]



Hierarchical Merging



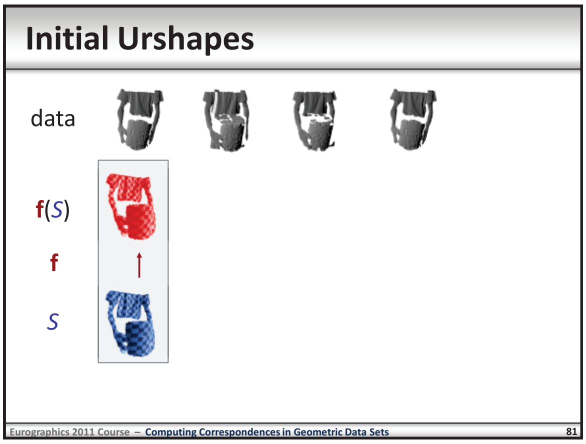
Hierarchical Merging

data

f(*S*)

f

S



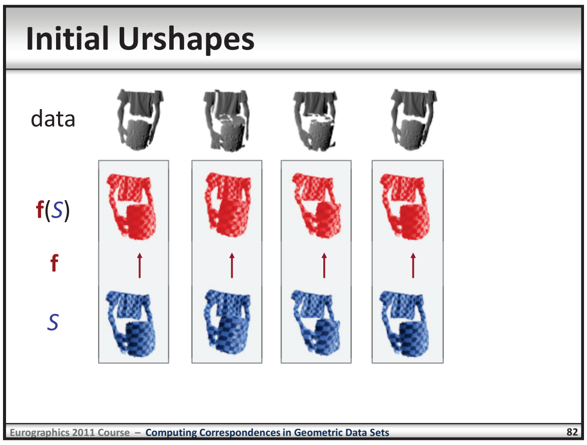
Initial Urshapes

data

f(*S*)

f

S



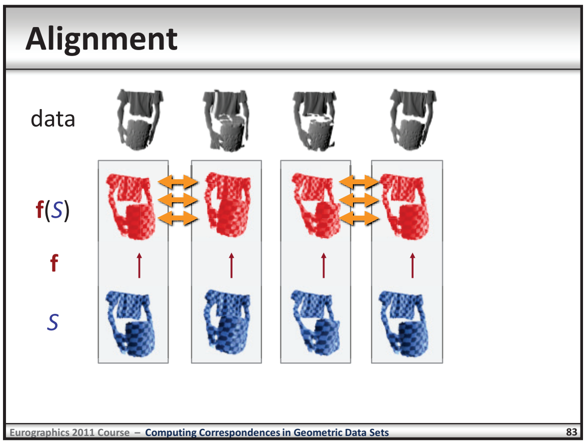
Initial Urshapes

data

f(*S*)

f

S



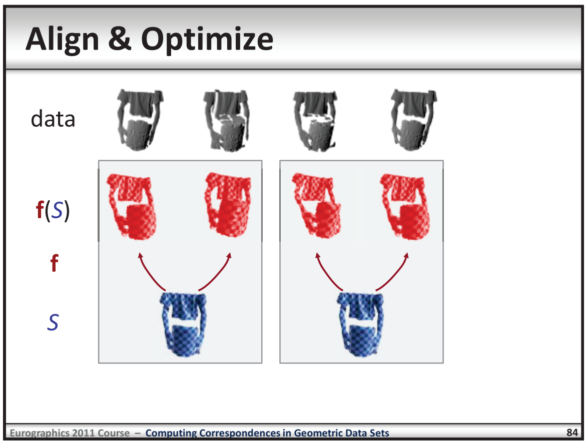
Alignment

data

f(*S*)

f

S



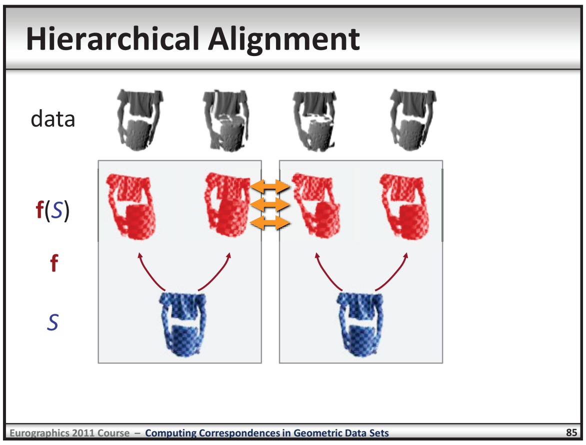
Align & Optimize

data

f(*S*)

f

S



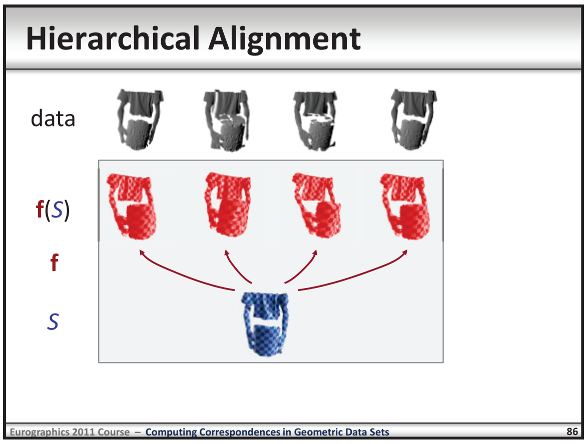
Hierarchical Alignment

data

f(*S*)

f

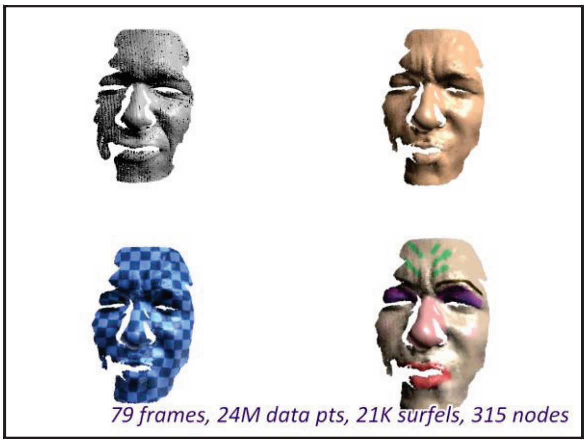
S



Hierarchical Alignment

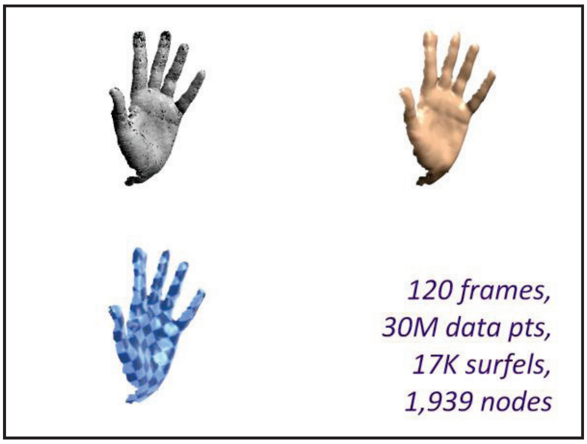
Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 86

Results

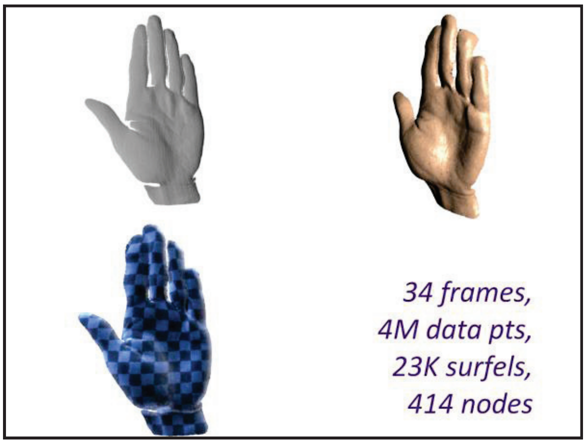


79 frames, 24M data pts, 21K surfels, 315 nodes

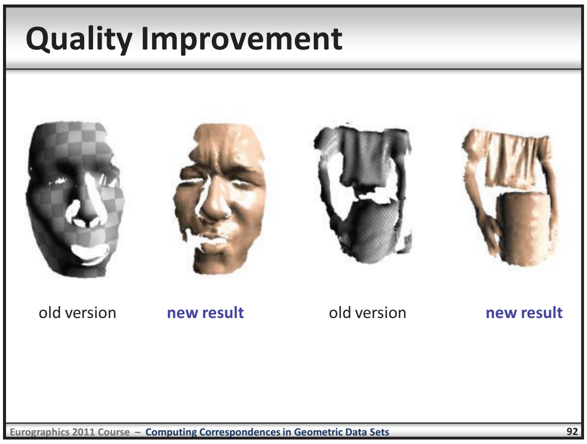
98 frames, 5M data pts, 6.4K surfels, 423 nodes



120 frames,
30M data pts,
17K surfels,
1,939 nodes



34 frames,
4M data pts,
23K surfels,
414 nodes



Quality Improvement

old version

new result

old version

new result

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 92

Summary: Variational Model

$$E(S,\mathbf{f},d) = \underbrace{E_{match}(S,\mathbf{f},d)}_{\text{data}} + \underbrace{(E_{rigid} + E_{volume} + E_{accel} + E_{velocity})}_{\text{deformatio n}}(S,\mathbf{f}) + \underbrace{E_{smooth}}_{\text{urshape}}(S)$$

$$E_{match}(S,f,d) = \sum_{t=1}^{T}\sum_{i=1}^{n_t} trunc(dist(d_i, f(S))^2)$$

$$E_{rigid}(S,\mathbf{f}) = \int_{V(S)} \omega_{rigid}(x)\left\|\nabla_{\mathbf{x}}\mathbf{f}(\mathbf{x},t)^T \nabla_{\mathbf{x}}\mathbf{f}(\mathbf{x},t) - \mathbf{I}\right\|_F^2 dx$$

$$E_{volume}(S,\mathbf{f}) = \int_{V(S)} \omega_{vol}(x)\left(\left|\nabla_{\mathbf{x}}\mathbf{f}(\mathbf{x},t)\right| - 1\right)^2 dx$$

$$E_{accel}(S,\mathbf{f}) = \int_S \omega_{acc}(x)\left(\frac{\partial^2}{\partial t^2}\mathbf{f}(\mathbf{x},t)\right)^2 dx \quad E_{velocity}(S,\mathbf{f}) = \int_S \omega_{velocity}(x)\left(\frac{\partial}{\partial t}\mathbf{f}(\mathbf{x},t)\right)^2 dx$$

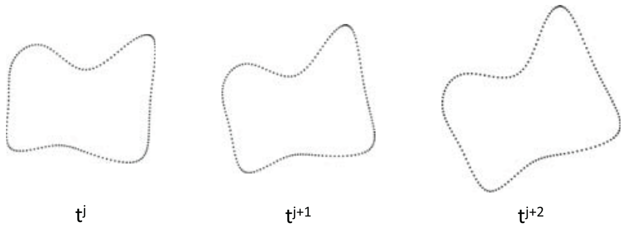
$$E_{smooth}(S) = \int_S \omega_{smooth}(x)\left(\nabla_{uv}^2 s(x)\right)^2 dx$$

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 94

Computing Correspondences in Geometric Datasets

Kinematic Surfaces

Time Ordered Scans



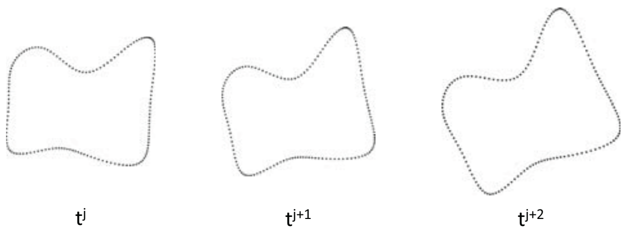
$$\widetilde{P}^j \equiv \{\widetilde{\mathbf{p}}_i^j\} := \{(\mathbf{p}_i^j, t^j), \mathbf{p}_i^j \in \mathbb{R}^d, t^j \in \mathbb{R}\}$$

Rigid Transformation

$$(R,t)?$$

$$p \to Rp + t \qquad R^T R = I$$

Time Ordered Scans

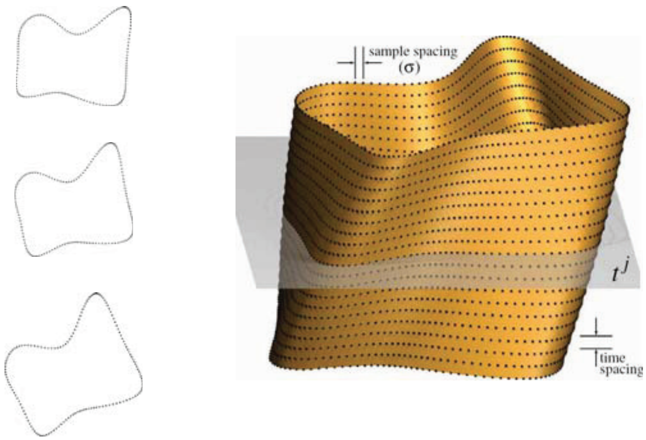


$(R,t)^j$ $(R,t)^{j+1}$

$$\widetilde{P}^j \equiv \{\widetilde{\mathbf{p}}_i^j\} := \{(\mathbf{p}_i^j, t^j), \mathbf{p}_i^j \in \mathbb{R}^d, t^j \in \mathbb{R}\}$$

Scanning (Moving) Objects

Space-time Surface



Kinematic Surfaces

Space-time registration → kinematic surface estimation

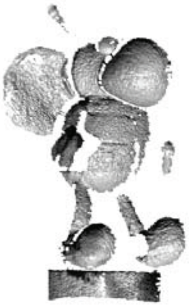
Computing Correspondences in Geometric Datasets

Dynamic Registration

Scan Registration

Solve for inter-frame motion: $\alpha_j := (\mathbf{R}_j, \mathbf{t}_j)$

Scan Registration



The Setup

Given:

A set of frames $\{P_0, P_1, \ldots P_n\}$

Goal:

Recover rigid motion $\{\alpha_1, \alpha_2, \ldots \alpha_n\}$ between adjacent frames

Scan Registration

Solve for inter-frame motion: $\alpha\ := (\mathbf{R}\ ,\mathbf{t}\ )$

The Setup

Smoothly varying object motion

Unknown correspondence between scans

Fast acquisition →
motion happens between frames

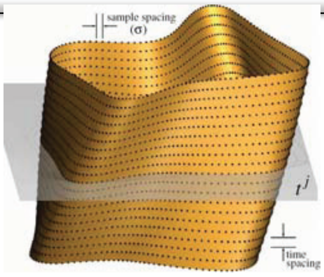
Insights

Rigid registration → kinematic property of space-time surface (locally exact)

Registration → surface normal estimation

Extension to deformable/articulated bodies

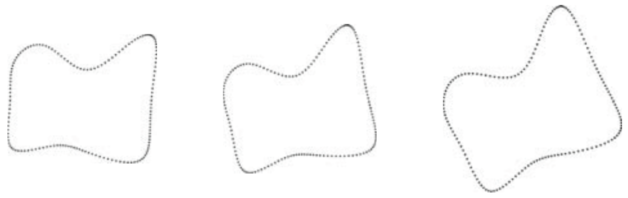
Space-time Surface



Δt^j

$$\widetilde{\mathbf{p}}_i^j \quad \to \quad \widetilde{\alpha}_j(\widetilde{\mathbf{p}}_i^j) = \left(\mathbf{R}_j\mathbf{p}_i^j + \mathbf{t}_j, \boxed{t^j + \Delta t^j}\right)$$

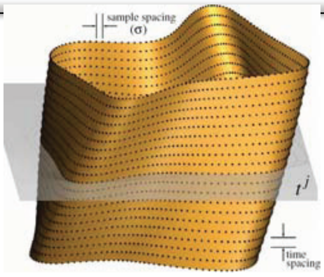
Time Ordered Scans



t^j t^{j+1} t^{j+2}

$$\widetilde{P}^j \equiv \{\widetilde{\mathbf{p}}_i^j\} := \{(\mathbf{p}_i^j, t^j), \mathbf{p}_i^j \in \mathbb{R}^d, t^j \in \mathbb{R}\}$$

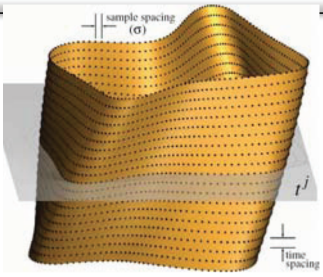
Space-time Surface



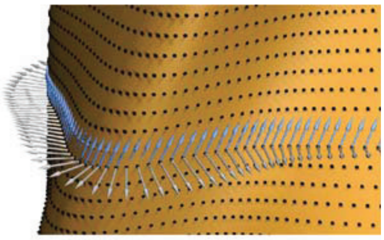
$$\widetilde{\mathbf{p}}_i^j \quad \to \quad \widetilde{\alpha}_j(\widetilde{\mathbf{p}}_i^j) = \left(\mathbf{R}_j\mathbf{p}_i^j + \mathbf{t}_j, t^j + \Delta t^j\right)$$

$$\widetilde{\alpha}_j = \operatorname{argmin} \sum_{i=1}^{|P^j|} d^2(\widetilde{\alpha}_j(\widetilde{\mathbf{p}}_i^j), S)$$

Space-time Surface



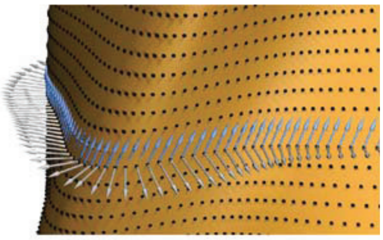
Spacetime Velocity Vectors



Tangential point movement → velocity vectors orthogonal to surface normals

$$\widetilde{\alpha}_j = \operatorname{argmin} \sum_{i=1}^{|P^j|} d^2(\widetilde{\alpha}_j(\widetilde{\mathbf{p}}_i^j), S)$$

Spacetime Velocity Vectors



Tangential point movement → velocity vectors orthogonal to surface normals

$$v(\tilde{p}_i^j).n(\tilde{p}_i^j) = 0$$

Registration Algorithm

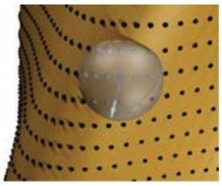
1. **Compute time coordinate spacing (σ), and form space-time surface.**
2. **Compute space time neighborhood using ANN, and locally estimate space-time surface normals.**
3. **Solve linear system to estimate (c_j,$\bar{c}_j$).**
4. **Convert velocity vectors to rotation matrix + translation vector using Plücker coordinates and quarternions.**

Final Steps

(rigid) velocity vectors → $\widetilde{\mathbf{v}}(\widetilde{\mathbf{p}}_i^j) = (\mathbf{c}_j \times \mathbf{p}_i^j + \bar{\mathbf{c}}_j, 1)$

$$\min_{\mathbf{c}_j,\bar{\mathbf{c}}_j} \sum_{i=1}^{|P^j|} w_i^j \left[(\mathbf{c}_j \times \mathbf{p}_i^j + \bar{\mathbf{c}}_j, 1) \cdot \widetilde{\mathbf{n}}_i^j \right]^2$$

Normal Estimation: PCA Based



Plane fitting using PCA using chosen neighborhood points.

Final Steps

(rigid) velocity vectors ! $\widetilde{\mathbf{v}}(\widetilde{\mathbf{p}}_i^j) = (\mathbf{c}_j \times \mathbf{p}_i^j + \bar{\mathbf{c}}_j, 1)$

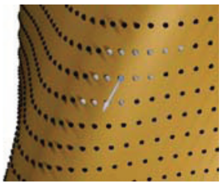
$$\min_{\mathbf{c}_j,\bar{\mathbf{c}}_j} \sum_{i=1}^{|P^j|} w_i^j \left[(\mathbf{c}_j \times \mathbf{p}_i^j + \bar{\mathbf{c}}_j, 1) \cdot \widetilde{\mathbf{n}}_i^j \right]^2$$

$$A\mathbf{x} + \mathbf{b} = 0$$

$$A = \sum_{i=1}^{|P^j|} w_i^j \begin{bmatrix} \bar{\mathbf{n}}_i^j \\ \mathbf{p}_i^j \times \bar{\mathbf{n}}_i^j \end{bmatrix} \begin{bmatrix} \bar{\mathbf{n}}_i^{j\,T} & (\mathbf{p}_i^j \times \bar{\mathbf{n}}_i^j)^T \end{bmatrix}$$

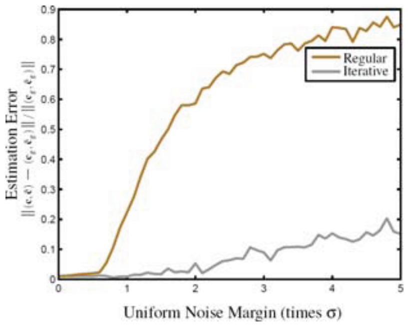
$$\mathbf{b} = \sum_{i=1}^{|P^j|} w_i^j n_i^j \begin{bmatrix} \bar{\mathbf{n}}_i^j \\ \mathbf{p}_i^j \times \bar{\mathbf{n}}_i^j \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} \bar{\mathbf{c}}_j \\ \mathbf{c}_j \end{bmatrix}$$

Normal Estimation: Iterative Refinement



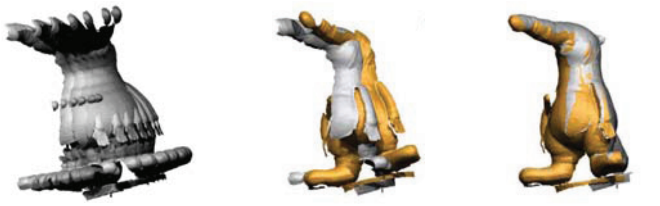
Update neighborhood with current velocity estimate.

Normal Refinement: Effect of Noise



Stable, but more expensive.

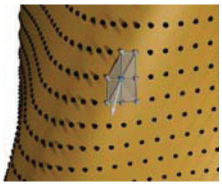
Comparison with ICP



ICP point-plane

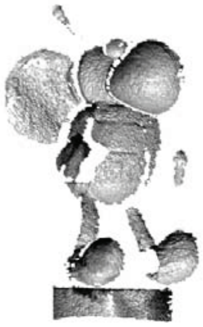
Dynamic registration

Normal Estimation: Local Triangulation

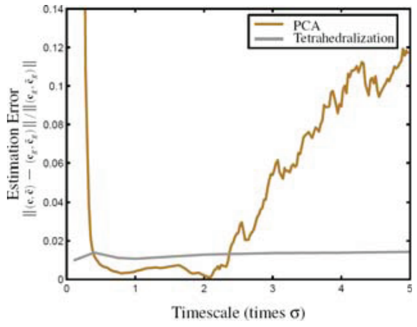


Perform local surface triangulation (tetrahedralization).

Rigid: Bee Sequence (2,200 frames)

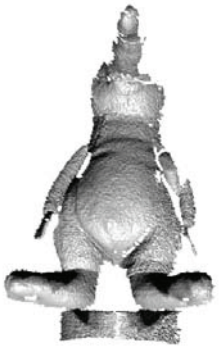


Normal Estimation

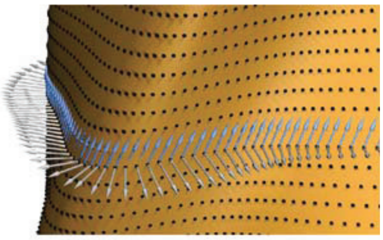


Stable, but more expensive.

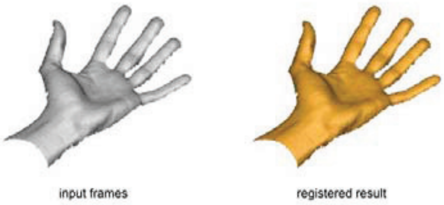
Rigid: Coati Sequence (2,200 frames)



Handling Large Number of Frames

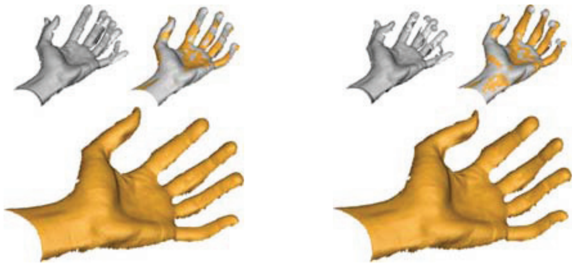


Deformable: Hand (100 frames)



Rigid/Deformable: Teapot Sequence (2,200 frames)

Deformable: Hand (100 frames)



scan #1 : scan #50

scan #1 : scan #100

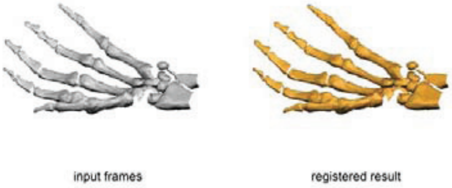
Deformable Bodies

$$\min_{\mathbf{c}_j,\bar{\mathbf{c}}_j} \sum_{i=1}^{|P^j|} w_i^j \left[(\mathbf{c}_j \times \mathbf{p}_i^j + \bar{\mathbf{c}}_j, 1)\cdot \tilde{\mathbf{n}}_i^j\right]^2$$

Cluster points, and solve smaller systems.

Propagate solutions with regularization.

Deformation + scanner motion: Skeleton (100 frames)

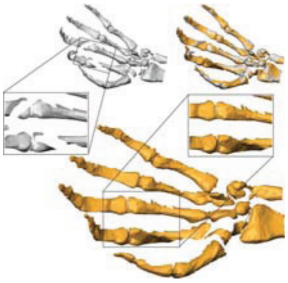


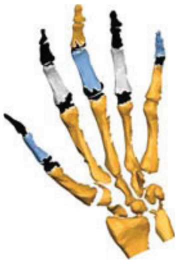
Deformation + scanner motion: Skeleton (100 frames)

scan #1 : scan #50

scan #1 : scan #100

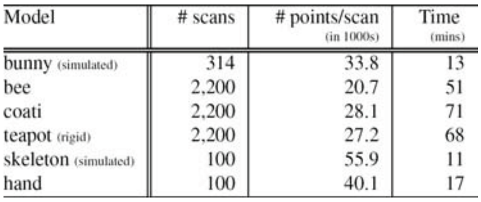
Deformation + scanner motion: Skeleton (100 frames)





rigid components

Performance (on 2.4GHz Athlon Dual Core, 2GB RAM)



Model	# scans	# points/scan (in 1000s)	Time (mins)
bunny (simulated)	314	33.8	13
bee	2,200	20.7	51
coati	2,200	28.1	71
teapot (rigid)	2,200	27.2	68
skeleton (simulated)	100	55.9	11
hand	100	40.1	17

Conclusion

Simple algorithm using kinematic properties of space-time surface.

Easy modification for deformable bodies.

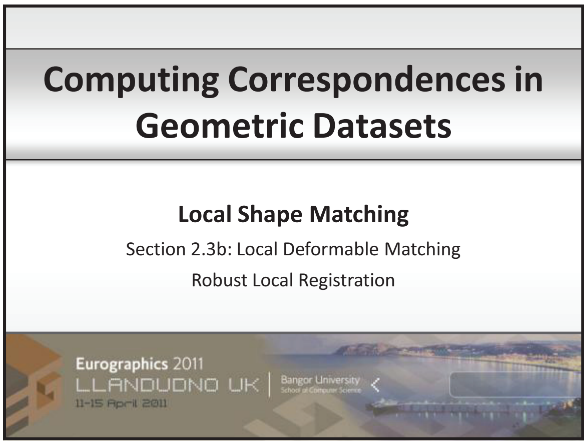
Suitable for use with fast scanners.

Limitations

Need more scans, dense scans, …

Sampling condition → time and space

thank you



Computing Correspondences in Geometric Datasets

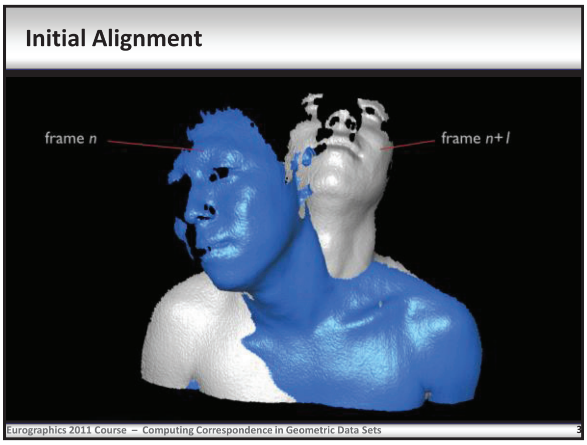
Local Shape Matching

Section 2.3b: Local Deformable Matching

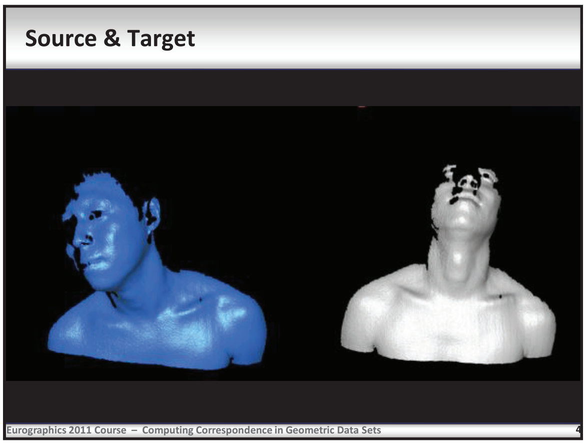
Robust Local Registration

Eurographics 2011
LLANDUDNO UK
11-15 April 2011
Bangor University
School of Computer Science

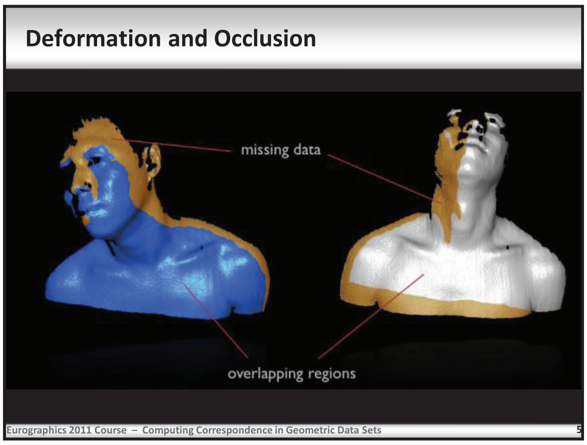
Pairwise Non-Rigid Registration



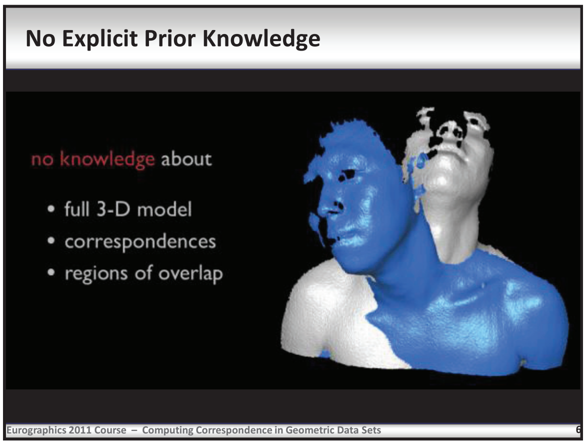
Initial Alignment



Source & Target



Deformation and Occlusion

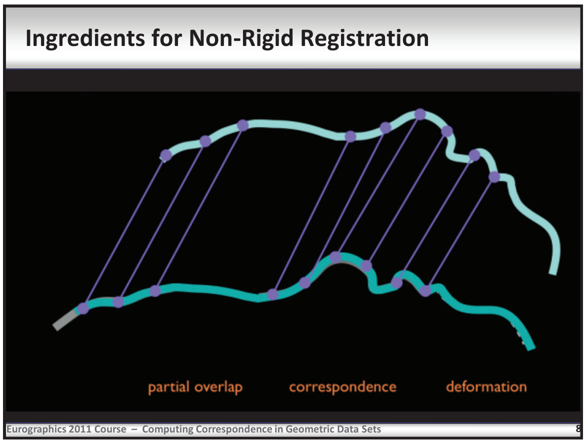


No Explicit Prior Knowledge

no knowledge about

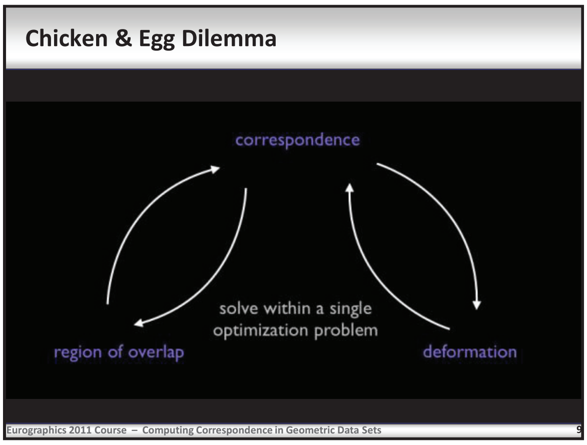
- full 3-D model
- correspondences
- regions of overlap

Goal: Automatic Local Registration



Ingredients for Non-Rigid Registration

partial overlap correspondence deformation



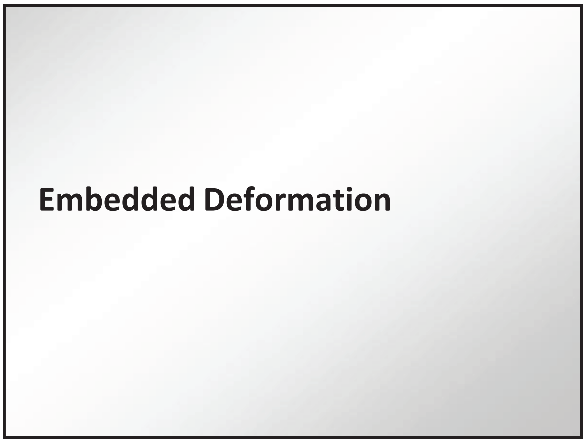
Chicken & Egg Dilemma

correspondence

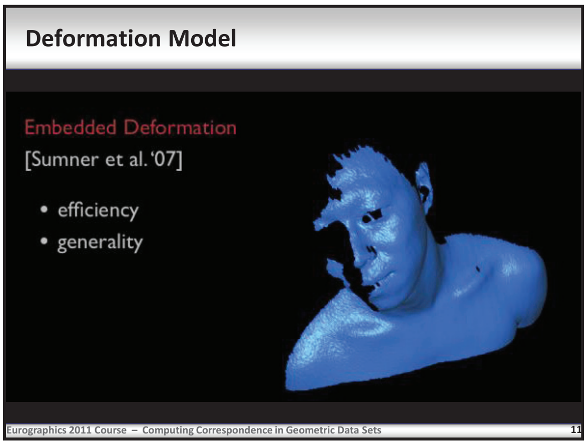
region of overlap

deformation

solve within a single optimization problem



Embedded Deformation

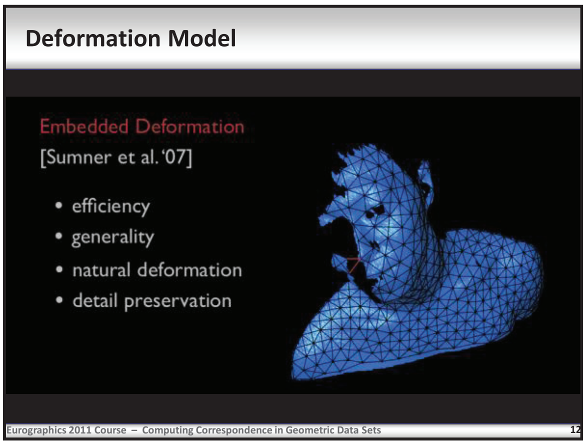


Deformation Model

Embedded Deformation

[Sumner et al. '07]

- efficiency
- generality

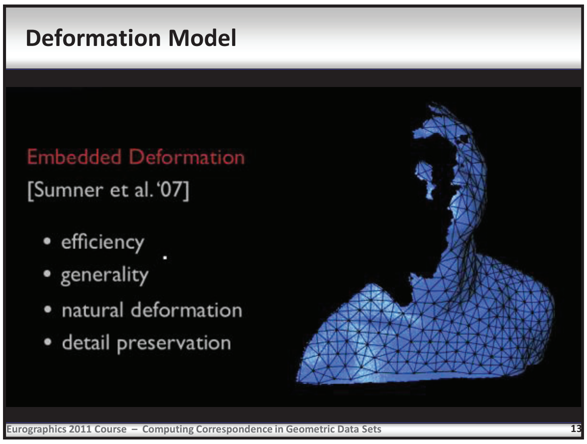


Deformation Model

Embedded Deformation

[Sumner et al. '07]

- efficiency
- generality
- natural deformation
- detail preservation

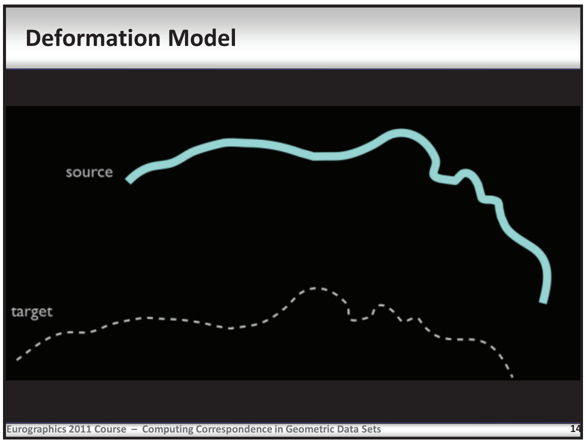


Deformation Model

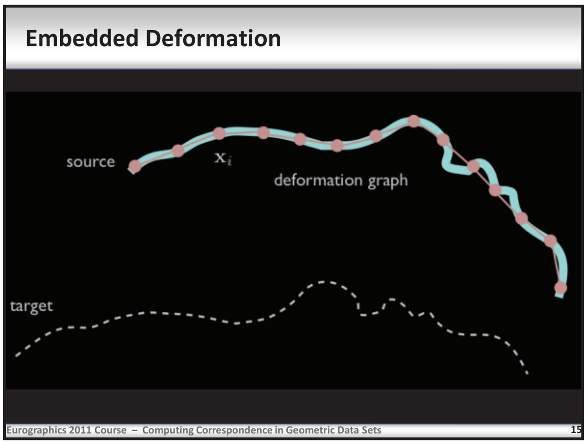
Embedded Deformation

[Sumner et al. '07]

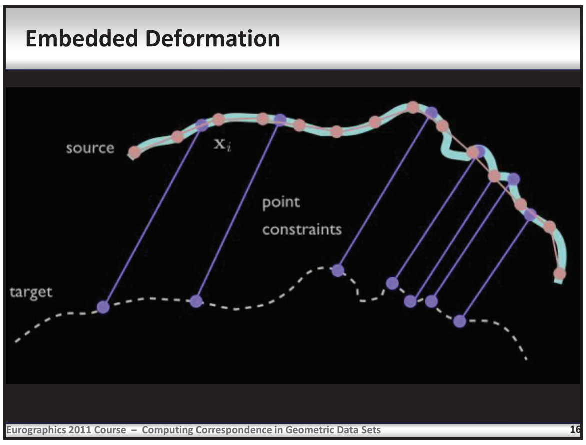
- efficiency
- generality
- natural deformation
- detail preservation



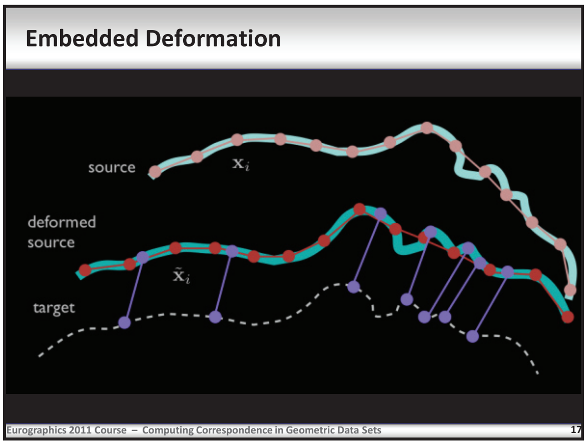
Deformation Model



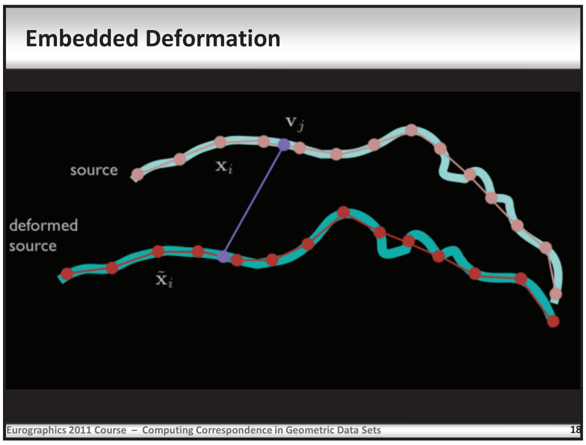
Embedded Deformation



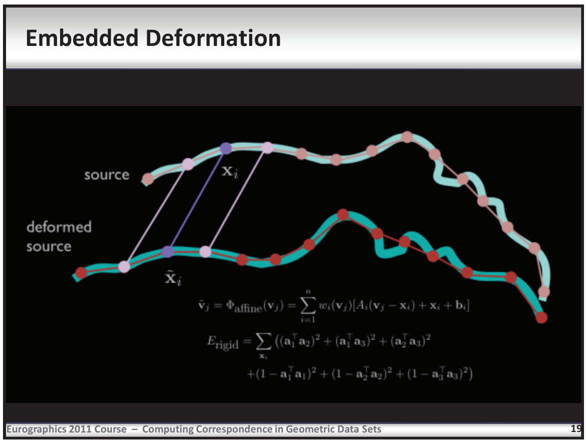
Embedded Deformation



Embedded Deformation



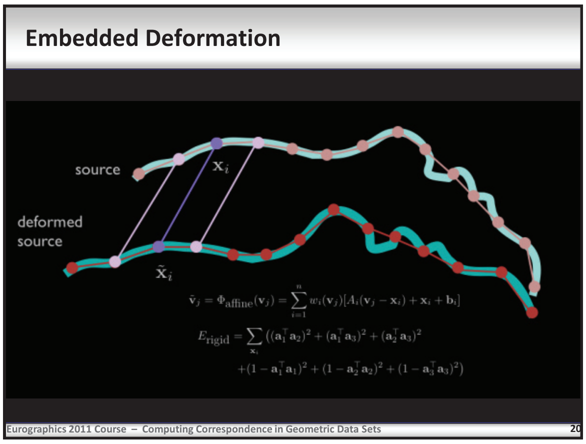
Embedded Deformation



Embedded Deformation

$$\tilde{\mathbf{v}}_j = \Phi_{\text{affine}}(\mathbf{v}_j) = \sum_{i=1}^{n} w_i(\mathbf{v}_j)[A_i(\mathbf{v}_j - \mathbf{x}_i) + \mathbf{x}_i + \mathbf{b}_i]$$

$$E_{\text{rigid}} = \sum_{\mathbf{x}_i} ((\mathbf{a}_1^\top \mathbf{a}_2)^2 + (\mathbf{a}_1^\top \mathbf{a}_3)^2 + (\mathbf{a}_2^\top \mathbf{a}_3)^2 + (1 - \mathbf{a}_1^\top \mathbf{a}_1)^2 + (1 - \mathbf{a}_2^\top \mathbf{a}_2)^2 + (1 - \mathbf{a}_3^\top \mathbf{a}_3)^2)$$

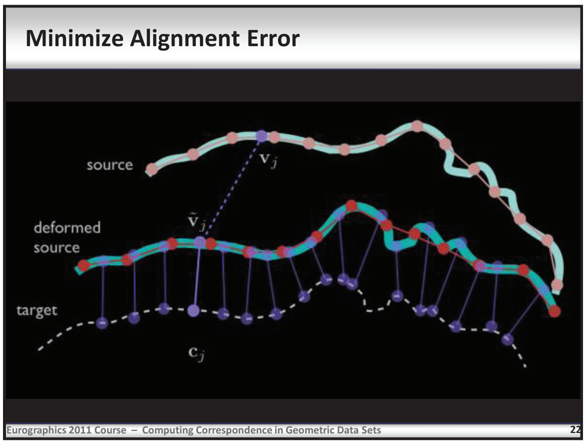


Embedded Deformation

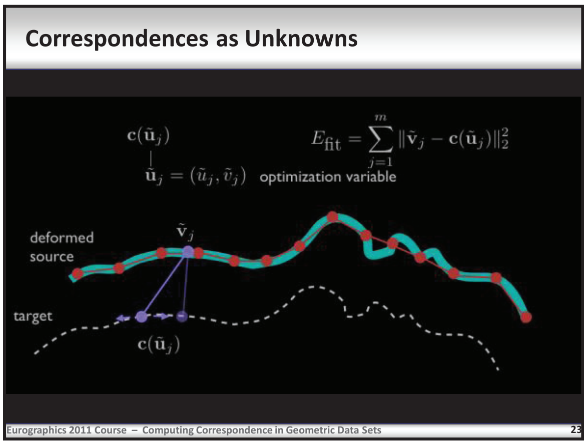
$$\tilde{\mathbf{v}}_j = \Phi_{\text{affine}}(\mathbf{v}_j) = \sum_{i=1}^{n} w_i(\mathbf{v}_j)[A_i(\mathbf{v}_j - \mathbf{x}_i) + \mathbf{x}_i + \mathbf{b}_i]$$

$$E_{\text{rigid}} = \sum_{\mathbf{x}_i} ((\mathbf{a}_1^\top \mathbf{a}_2)^2 + (\mathbf{a}_1^\top \mathbf{a}_3)^2 + (\mathbf{a}_2^\top \mathbf{a}_3)^2 + (1 - \mathbf{a}_1^\top \mathbf{a}_1)^2 + (1 - \mathbf{a}_2^\top \mathbf{a}_2)^2 + (1 - \mathbf{a}_3^\top \mathbf{a}_3)^2)$$

Global Optimal Correspondence Optimization



Minimize Alignment Error

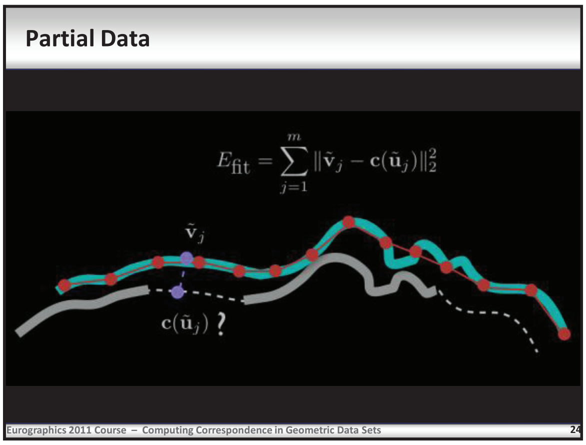


Correspondences as Unknowns

$$\mathbf{c}(\tilde{\mathbf{u}}_j)$$

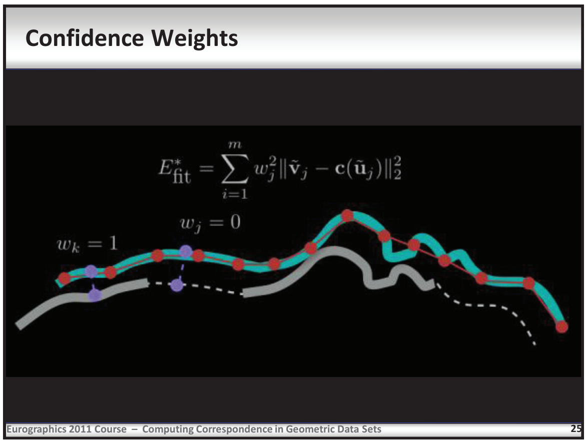
$$\tilde{\mathbf{u}}_j = (\tilde{u}_j, \tilde{v}_j)$$ optimization variable

$$E_{\text{fit}} = \sum_{j=1}^{m} \|\tilde{\mathbf{v}}_j - \mathbf{c}(\tilde{\mathbf{u}}_j)\|_2^2$$



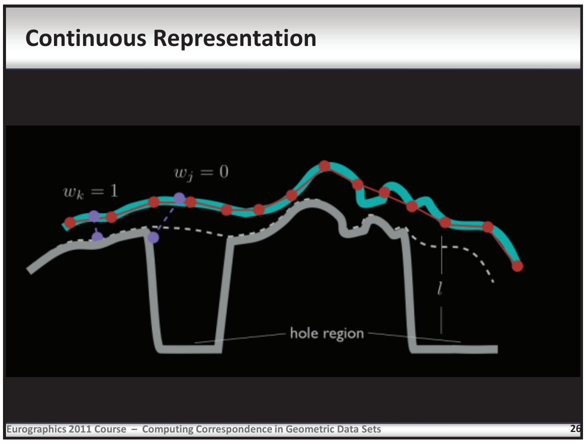
Partial Data

$$E_{\text{fit}} = \sum_{j=1}^{m} \|\tilde{\mathbf{v}}_j - \mathbf{c}(\tilde{\mathbf{u}}_j)\|_2^2$$

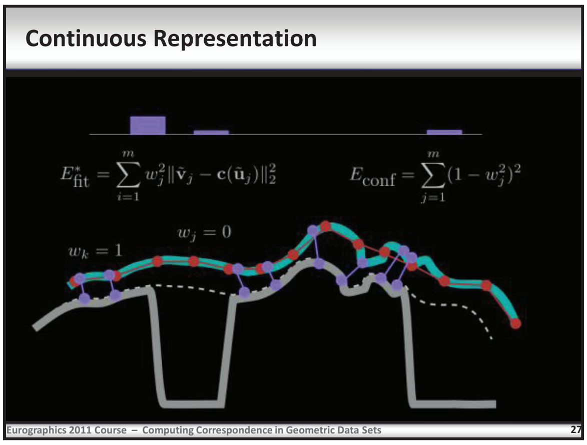


Confidence Weights

$$E^*_{\text{fit}} = \sum_{i=1}^{m} w_j^2 \|\tilde{\mathbf{v}}_j - \mathbf{c}(\tilde{\mathbf{u}}_j)\|_2^2$$

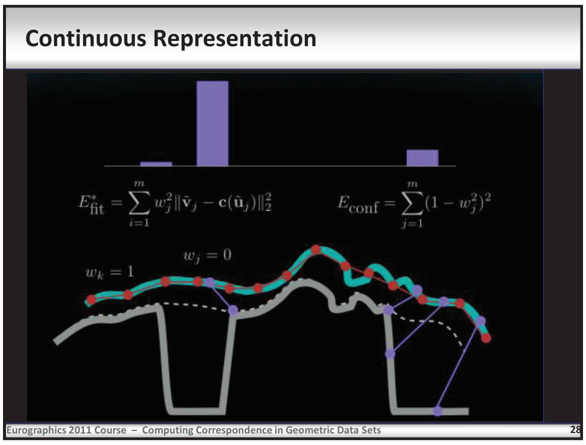


Continuous Representation



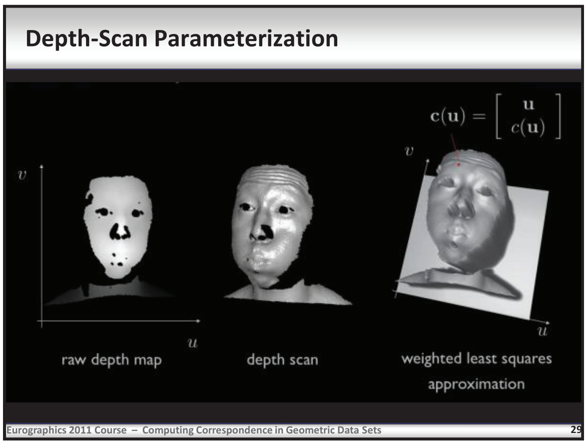
Continuous Representation

$$E^*_{\text{fit}} = \sum_{i=1}^{m} w_j^2 \|\tilde{\mathbf{v}}_j - \mathbf{c}(\tilde{\mathbf{u}}_j)\|_2^2 \qquad E_{\text{conf}} = \sum_{j=1}^{m} (1 - w_j^2)^2$$



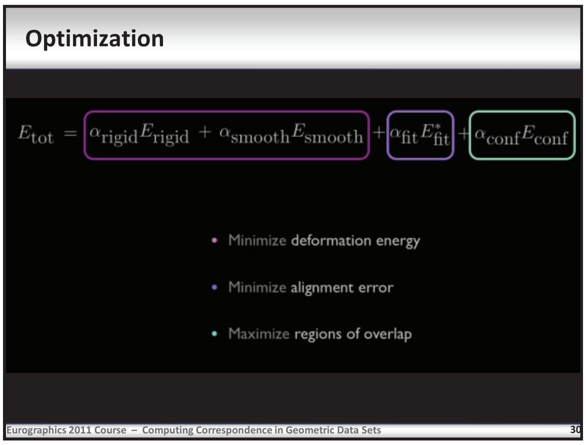
Continuous Representation

$$E^*_{\text{fit}} = \sum_{i=1}^{m} w_j^2 \|\tilde{\mathbf{v}}_j - \mathbf{c}(\tilde{\mathbf{u}}_j)\|_2^2 \qquad E_{\text{conf}} = \sum_{j=1}^{m} (1 - w_j^2)^2$$



Depth-Scan Parameterization

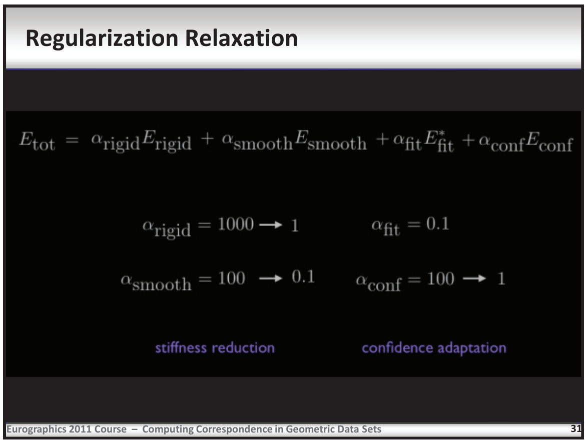
$$\mathbf{c}(\mathbf{u}) = \begin{bmatrix} \mathbf{u} \\ c(\mathbf{u}) \end{bmatrix}$$



Optimization

$$E_{\text{tot}} = \boxed{\alpha_{\text{rigid}} E_{\text{rigid}} + \alpha_{\text{smooth}} E_{\text{smooth}}} + \boxed{\alpha_{\text{fit}} E^*_{\text{fit}}} + \boxed{\alpha_{\text{conf}} E_{\text{conf}}}$$

- Minimize deformation energy
- Minimize alignment error
- Maximize regions of overlap



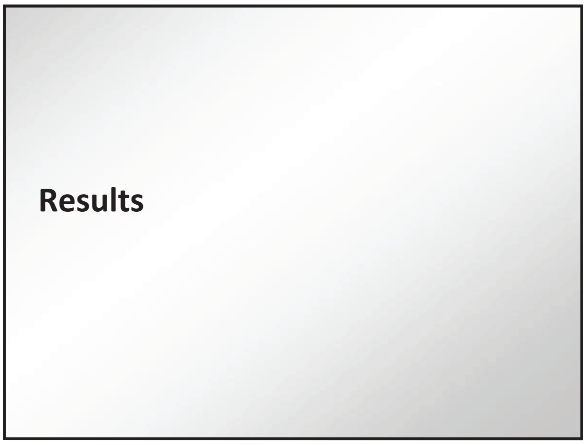
Regularization Relaxation

$$E_{\text{tot}} = \alpha_{\text{rigid}}E_{\text{rigid}} + \alpha_{\text{smooth}}E_{\text{smooth}} + \alpha_{\text{fit}}E^*_{\text{fit}} + \alpha_{\text{conf}}E_{\text{conf}}$$

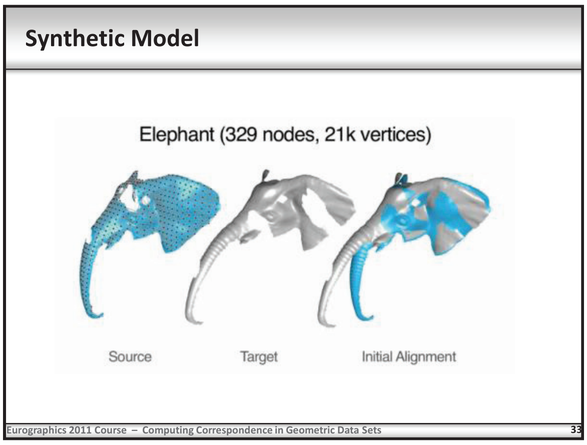
$\alpha_{\text{rigid}} = 1000 \rightarrow 1$ $\alpha_{\text{fit}} = 0.1$

$\alpha_{\text{smooth}} = 100 \rightarrow 0.1$ $\alpha_{\text{conf}} = 100 \rightarrow 1$

stiffness reduction confidence adaptation



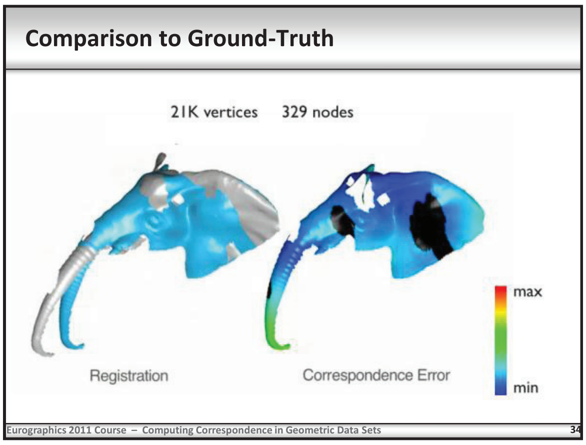
Results



Synthetic Model

Elephant (329 nodes, 21k vertices)

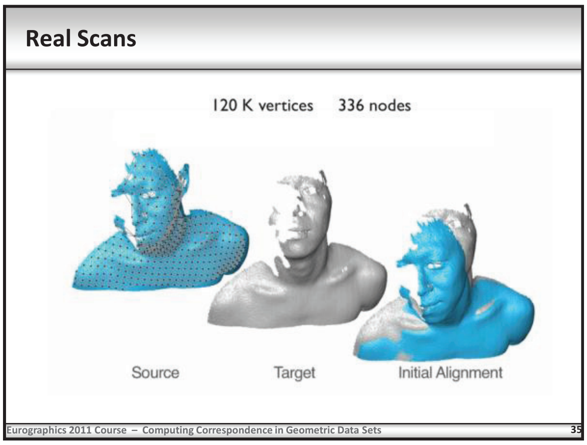
Source Target Initial Alignment



Comparison to Ground-Truth

21K vertices 329 nodes

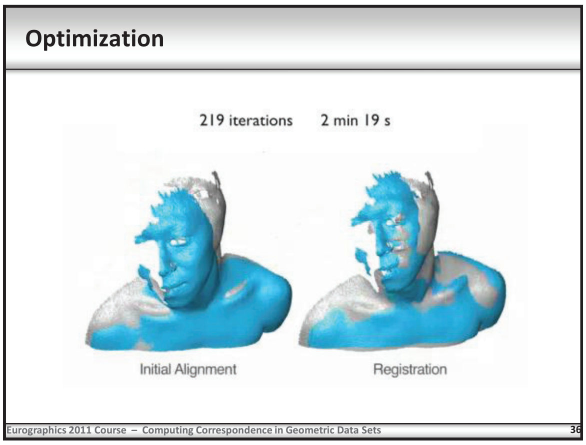
Registration Correspondence Error



Real Scans

120 K vertices 336 nodes

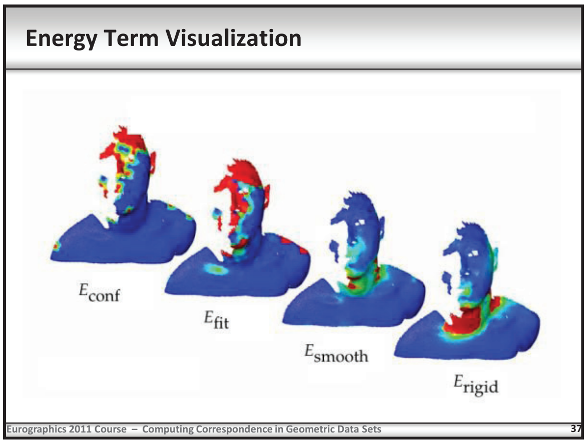
Source Target Initial Alignment



Optimization

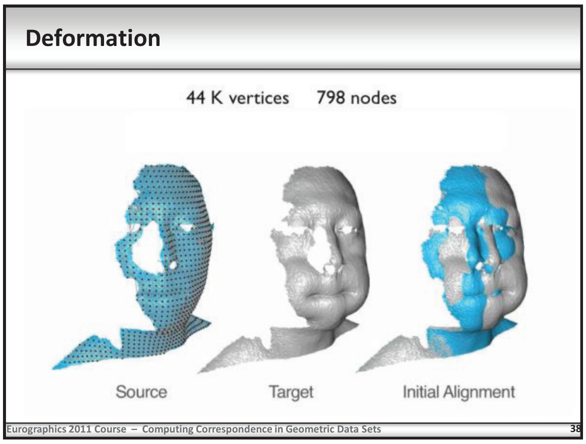
219 iterations 2 min 19 s

Initial Alignment Registration



Energy Term Visualization

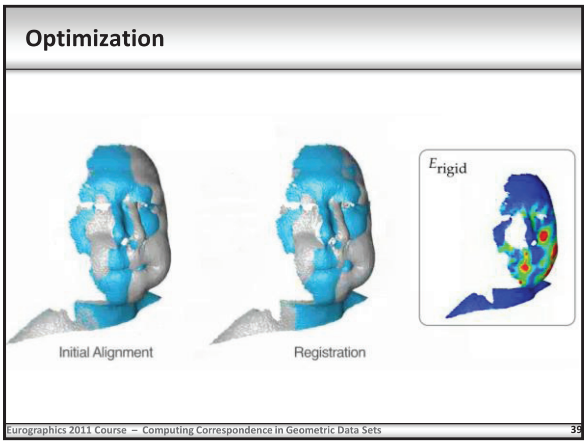
Eurographics 2011 Course – Computing Correspondence in Geometric Data Sets 37



Deformation

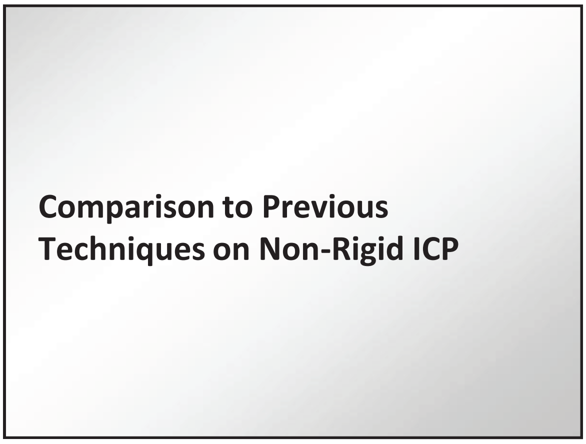
44 K vertices 798 nodes

Eurographics 2011 Course – Computing Correspondence in Geometric Data Sets 38

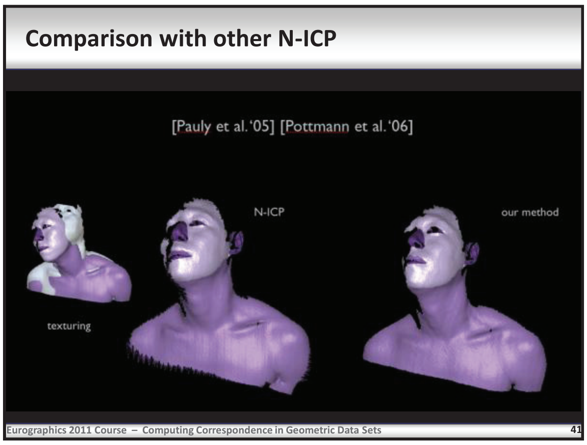


Optimization

Eurographics 2011 Course – Computing Correspondence in Geometric Data Sets 39



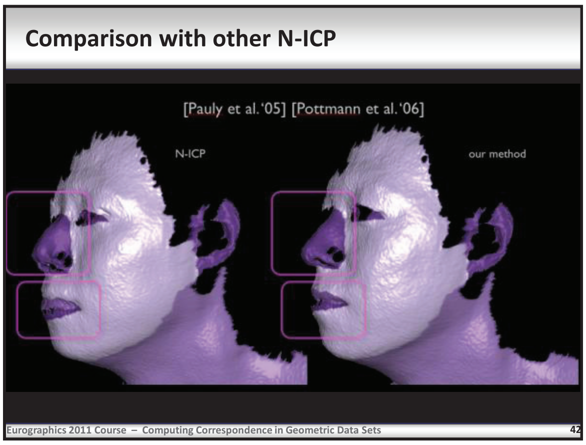
Comparison to Previous Techniques on Non-Rigid ICP



Comparison with other N-ICP

[Pauly et al.'05] [Pottmann et al.'06]

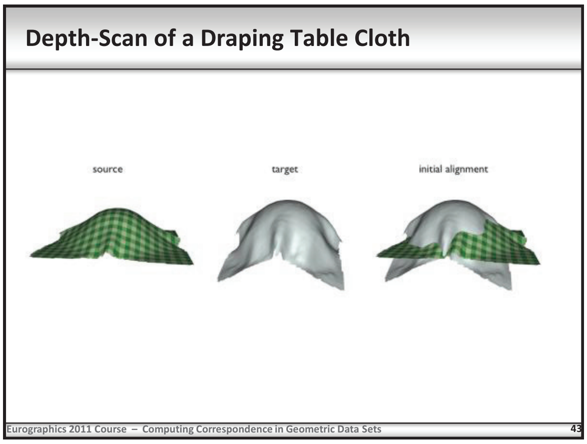
Eurographics 2011 Course – Computing Correspondence in Geometric Data Sets 41



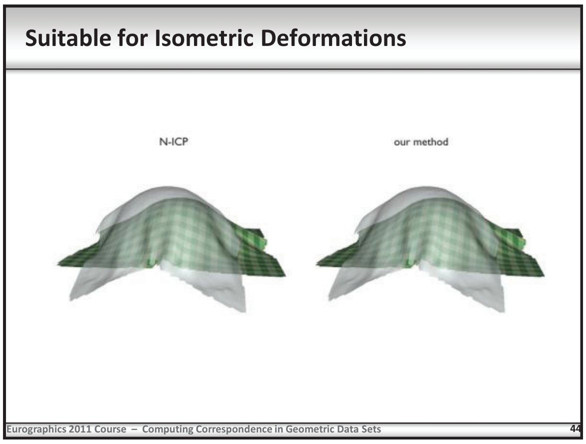
Comparison with other N-ICP

[Pauly et al.'05] [Pottmann et al.'06]

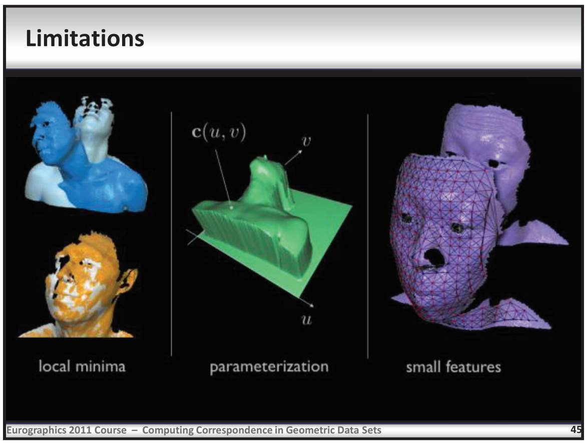
Eurographics 2011 Course – Computing Correspondence in Geometric Data Sets 42



Depth-Scan of a Draping Table Cloth



Suitable for Isometric Deformations

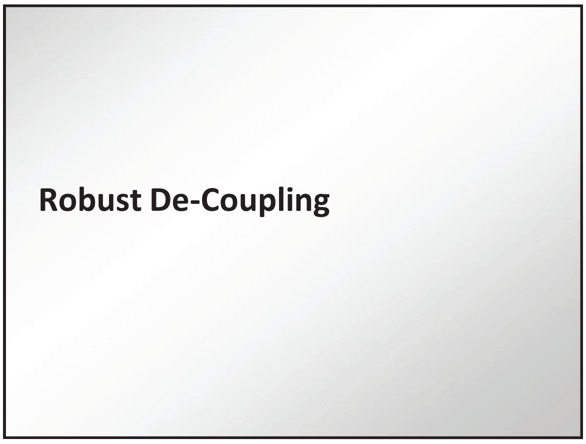


Limitations

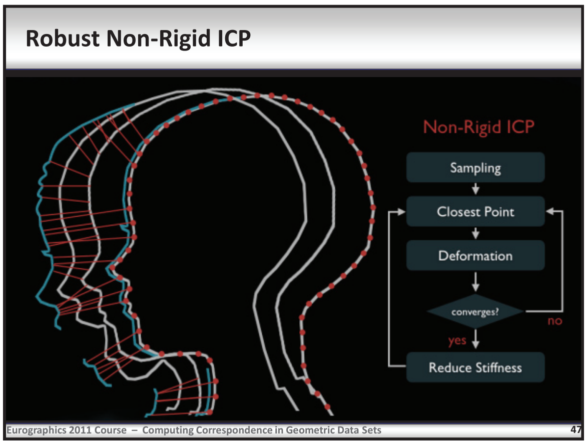
local minima

parameterization

small features



Robust De-Coupling



Robust Non-Rigid ICP

Non-Rigid ICP

Sampling

Closest Point

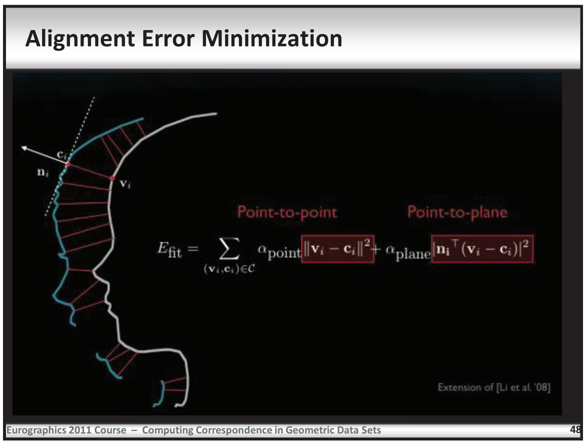
Deformation

converges?

no

yes

Reduce Stiffness



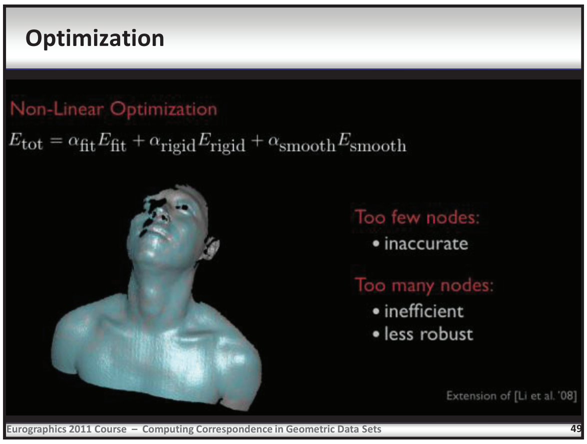
Alignment Error Minimization

Point-to-point

Point-to-plane

$$E_{\text{fit}} = \sum_{(\mathbf{v}_i,\mathbf{c}_i)\in\mathcal{C}} \alpha_{\text{point}} \boxed{\|\mathbf{v}_i - \mathbf{c}_i\|^2} + \alpha_{\text{plane}} \boxed{|\mathbf{n}_i^\top(\mathbf{v}_i - \mathbf{c}_i)|^2}$$

Extension of [Li et al. '08]



Optimization

Non-Linear Optimization

$$E_{\text{tot}} = \alpha_{\text{fit}} E_{\text{fit}} + \alpha_{\text{rigid}} E_{\text{rigid}} + \alpha_{\text{smooth}} E_{\text{smooth}}$$

Too few nodes:

- inaccurate

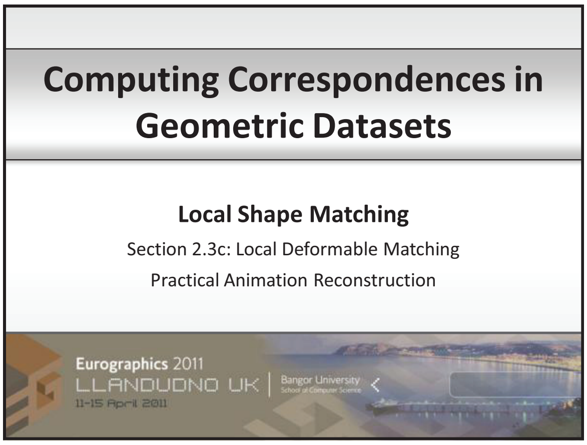
Too many nodes:

- inefficient
- less robust

Extension of [Li et al. '08]

Eurographics 2011 Course – Computing Correspondence in Geometric Data Sets 49

Talk to you later!



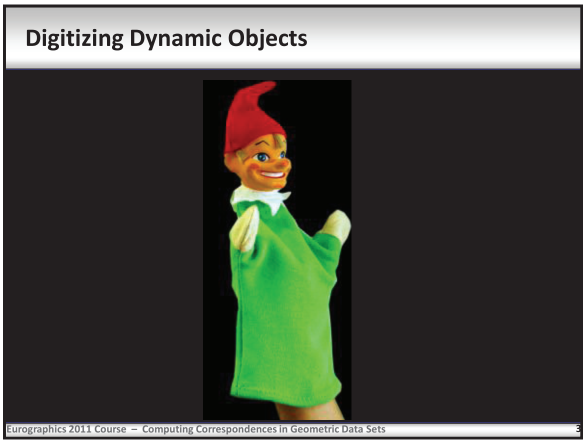
Computing Correspondences in Geometric Datasets

Local Shape Matching

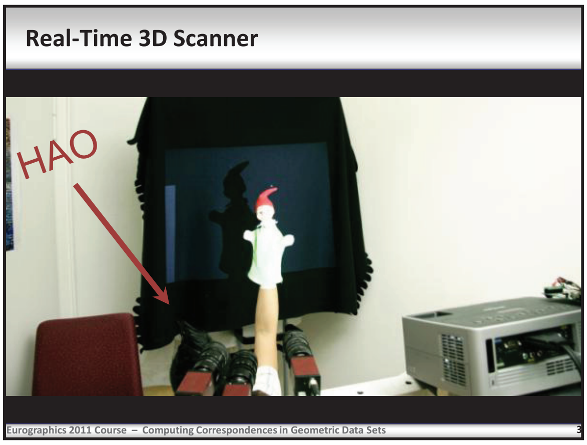
Section 2.3c: Local Deformable Matching

Practical Animation Reconstruction

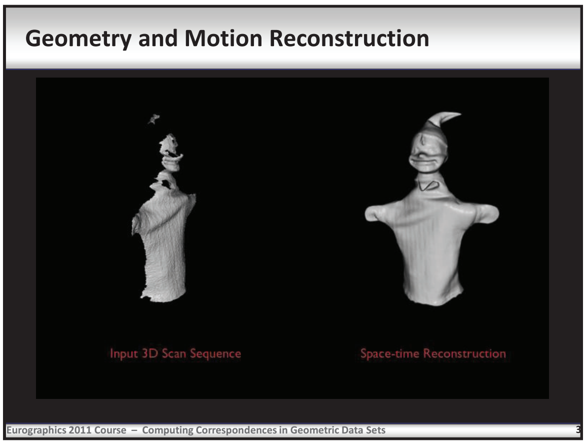
Eurographics 2011
LLANDUDNO UK
11-15 April 2011
Bangor University
School of Computer Science



Digitizing Dynamic Objects

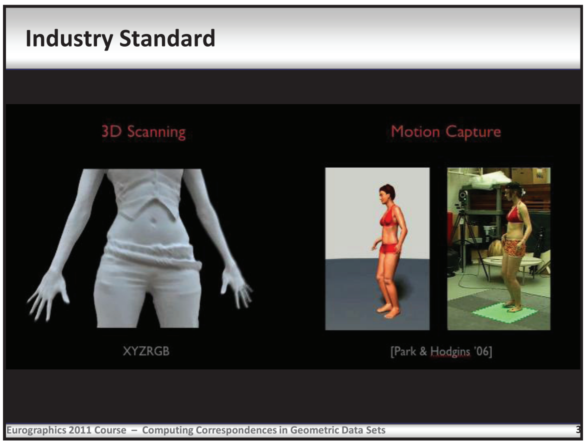


Real-Time 3D Scanner

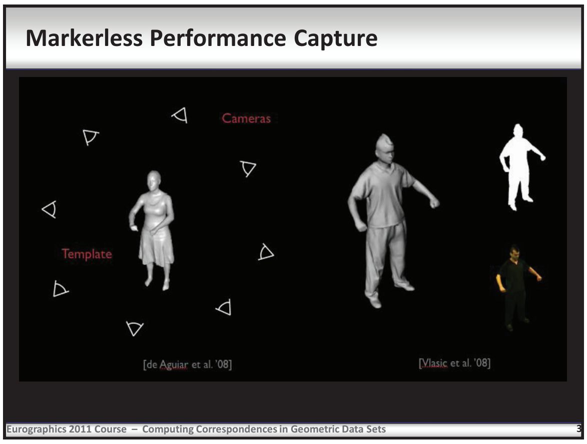


Geometry and Motion Reconstruction

State of the Art



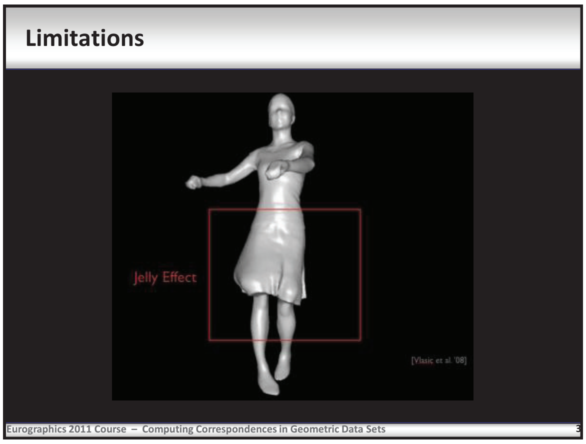
Industry Standard



Markerless Performance Capture

[de Aguiar et al. '08]

[Vlasic et al. '08]

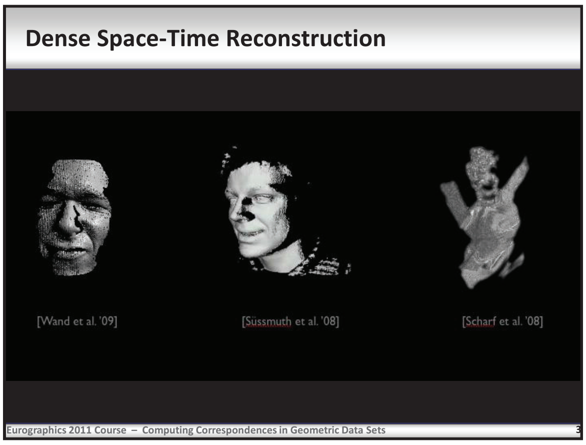


Limitations

[Vlasic et al. '08]

Real-Time 3D Scanner

[Weise et al. '07]

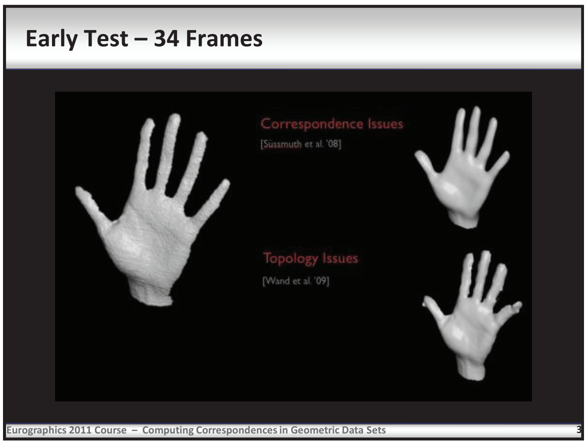


Dense Space-Time Reconstruction

[Wand et al. '09]

[Süssmuth et al. '08]

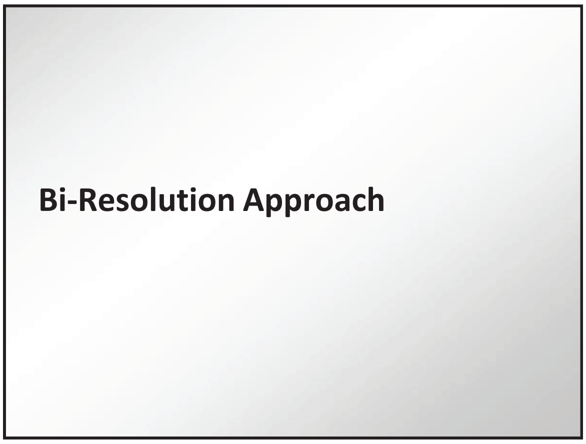
[Scharf et al. '08]



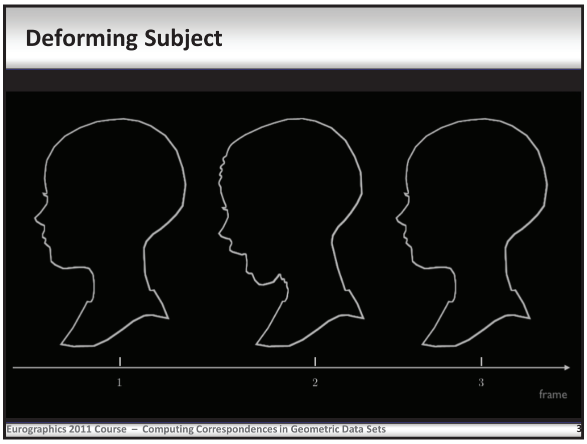
Early Test – 34 Frames

[Süssmuth et al. '08]

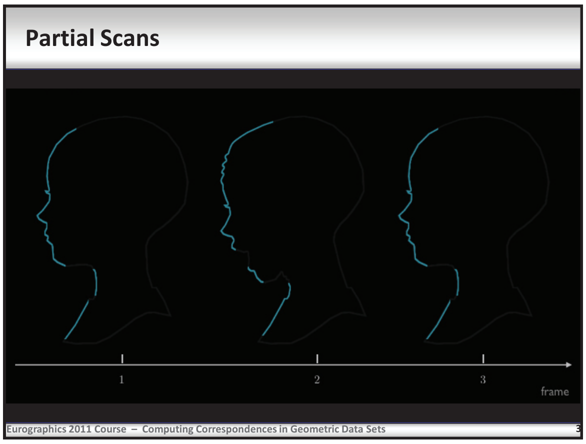
[Wand et al. '09]



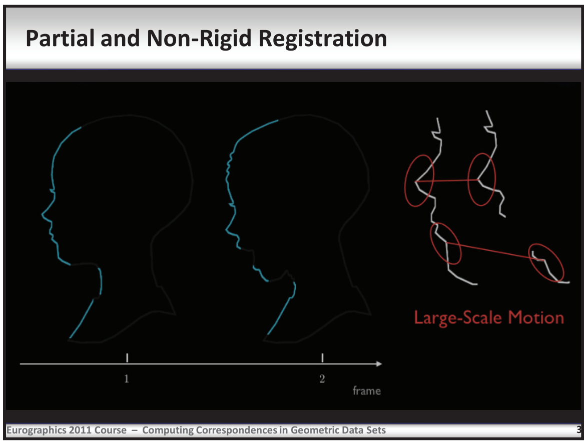
Bi-Resolution Approach



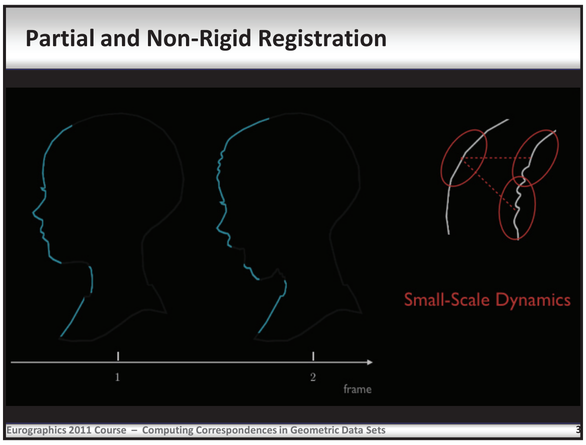
Deforming Subject



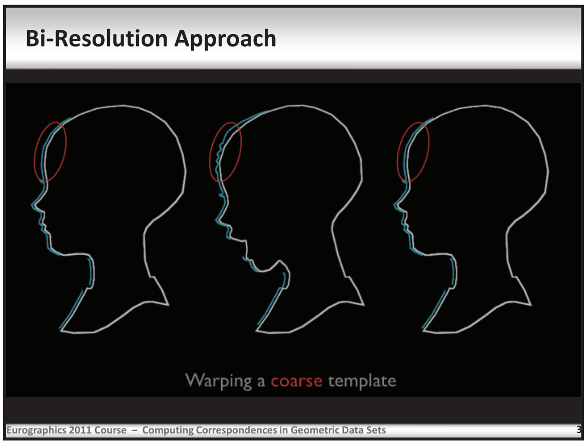
Partial Scans



Partial and Non-Rigid Registration

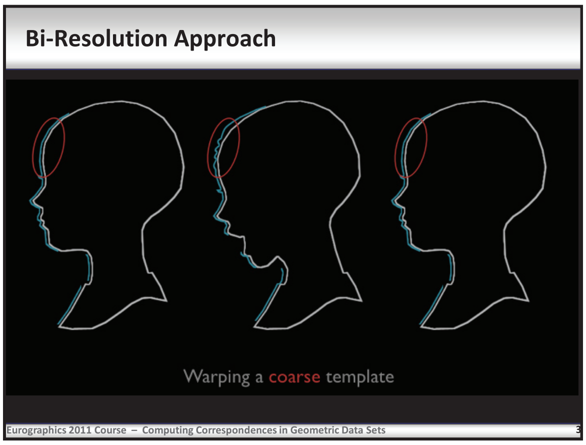


Partial and Non-Rigid Registration



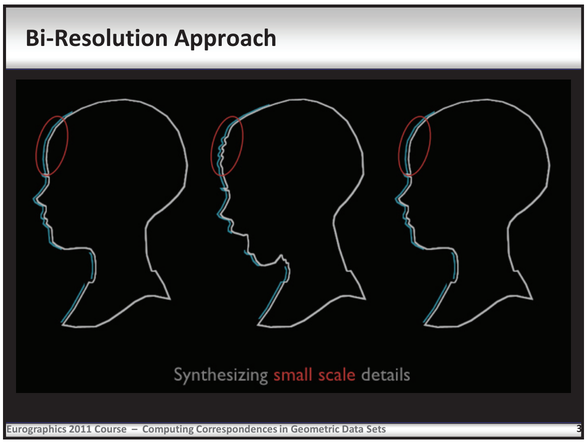
Bi-Resolution Approach

Warping a coarse template



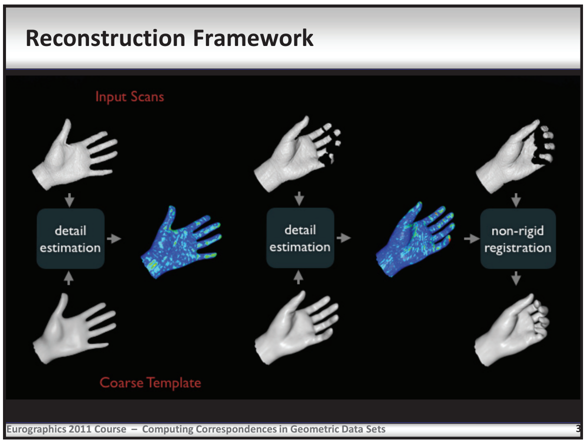
Bi-Resolution Approach

Warping a coarse template



Bi-Resolution Approach

Synthesizing small scale details



Reconstruction Framework

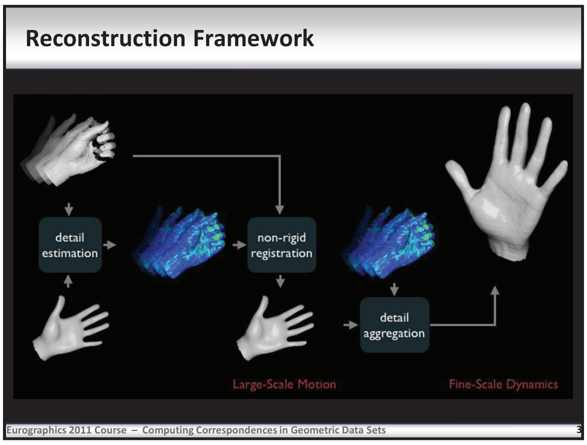
Input Scans

detail estimation

detail estimation

non-rigid registration

Coarse Template



Reconstruction Framework

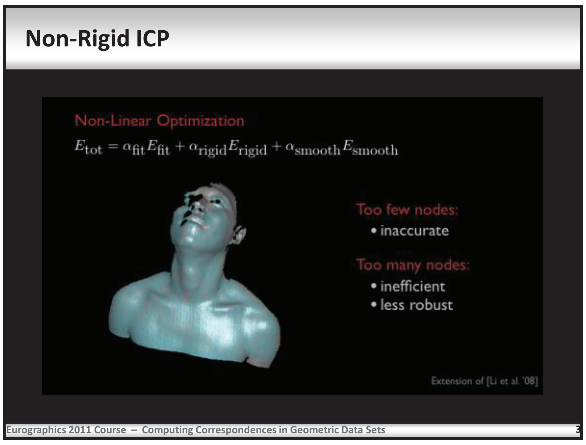
detail estimation

non-rigid registration

detail aggregation

Large-Scale Motion

Fine-Scale Dynamics



Non-Rigid ICP

Non-Linear Optimization

$$E_{\text{tot}} = \alpha_{\text{fit}} E_{\text{fit}} + \alpha_{\text{rigid}} E_{\text{rigid}} + \alpha_{\text{smooth}} E_{\text{smooth}}$$

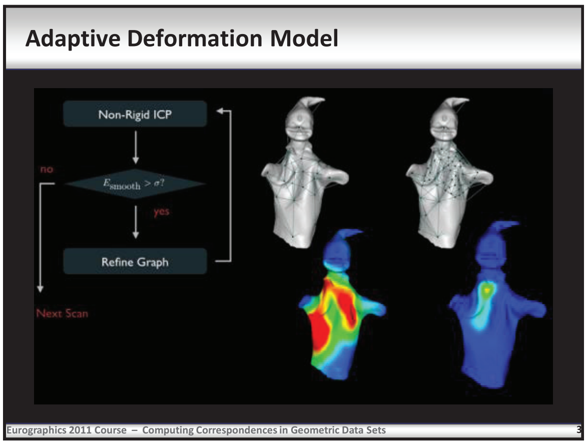
Too few nodes:

- inaccurate

Too many nodes:

- inefficient
- less robust

Extension of [Li et al. '08]



Adaptive Deformation Model

Non-Rigid ICP

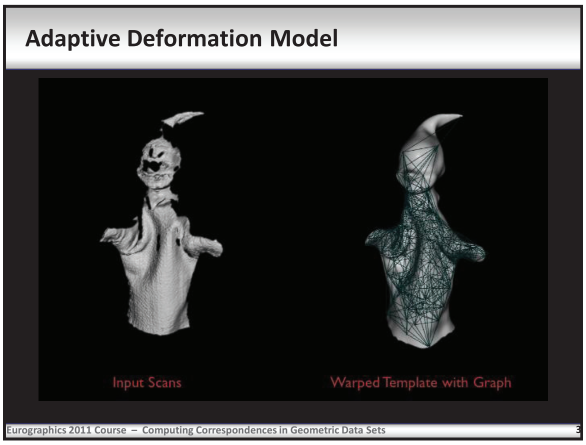
$E_{\text{smooth}} > \sigma?$

no

yes

Refine Graph

Next Scan

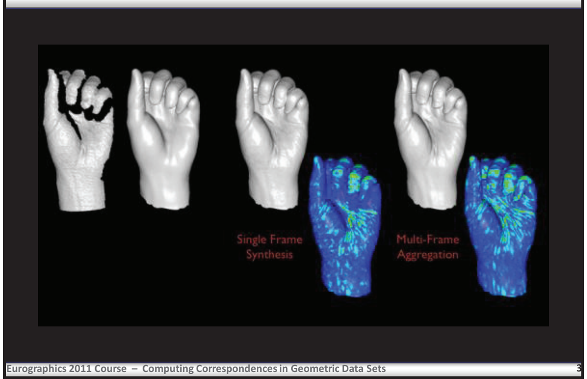


Adaptive Deformation Model

Input Scans

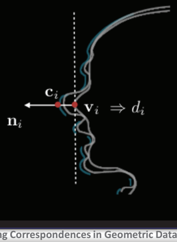
Warped Template with Graph

Detail Aggregation

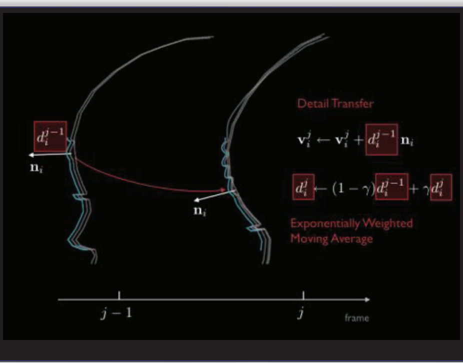


Detail Estimation

$$E_{\text{detail}} = \sum_{i \in \mathcal{V}} \underbrace{\boxed{\|\mathbf{v}_i + d_i \mathbf{n}_i - \mathbf{c}_i\|^2}}_{\text{Point Constraint}} + \beta \sum_{(i,j)\in\mathcal{E}} \underbrace{\boxed{|d_i - d_j|^2}}_{\text{Regularization}}$$



Detail Estimation



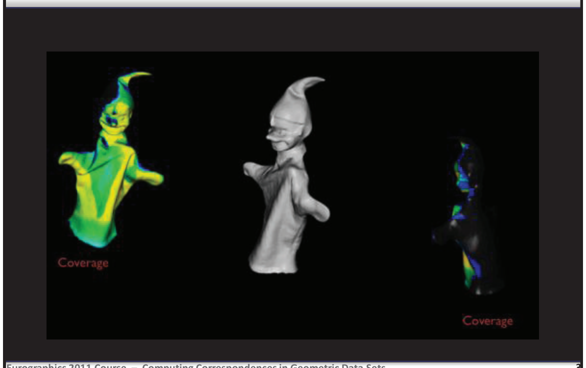
Detail Transfer

$$\mathbf{v}_i^j \leftarrow \mathbf{v}_i^j + \boxed{d_i^{j-1}}\, \mathbf{n}_i$$

$$\boxed{d_i^j} \leftarrow (1-\gamma)\boxed{d_i^{j-1}} + \gamma \boxed{d_i^j}$$

Exponentially Weighted Moving Average

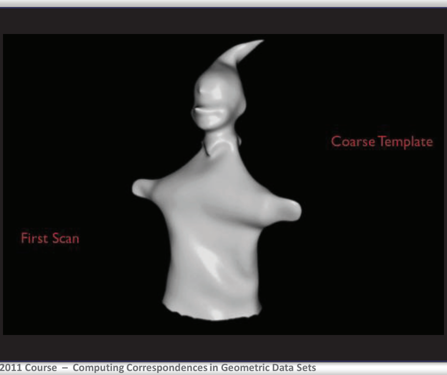
Forward-Backward Propagation



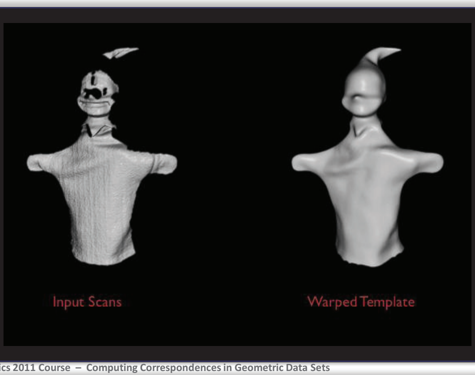
Reconstruction Process
The Puppet

3D Acquisition – 100 Frames

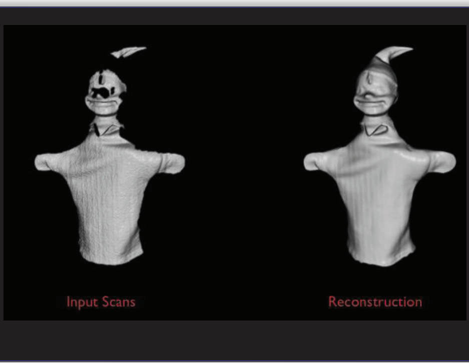
Initial Registration



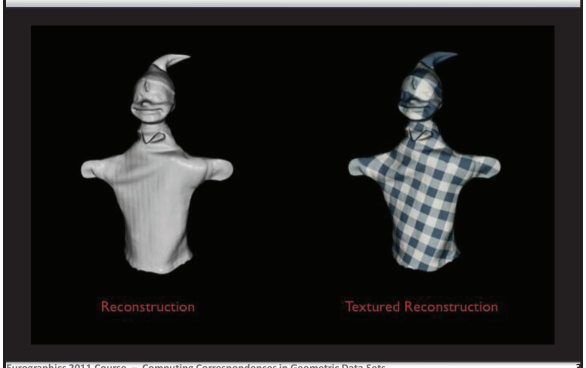
Template Warping



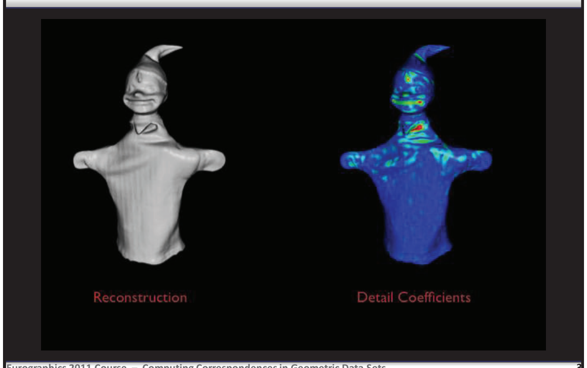
Final Reconstruction – 100 Frames



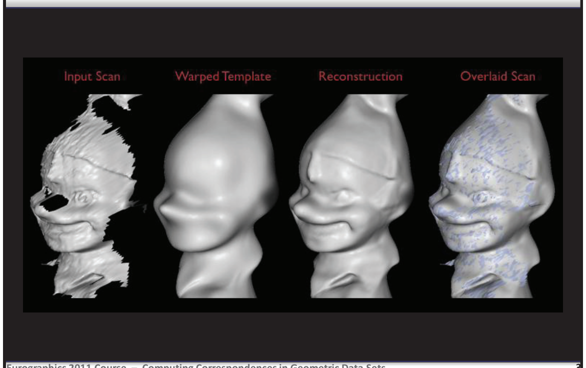
Correspondence Visualization

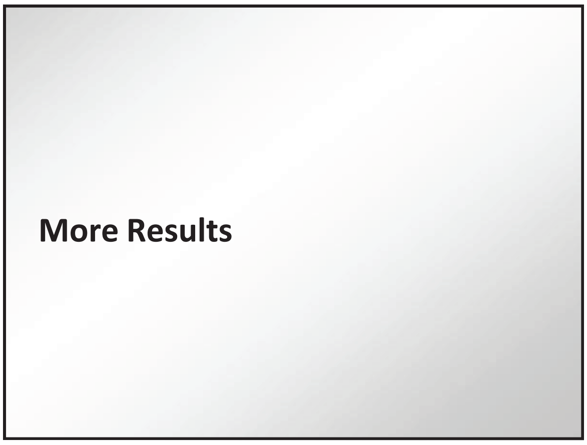


Detail-Coefficient Stability

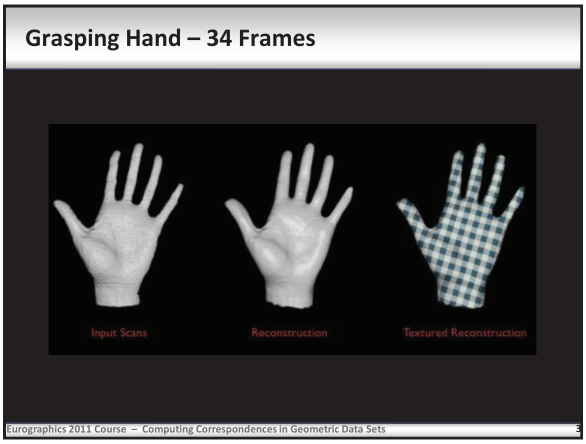


Close-up Comparison



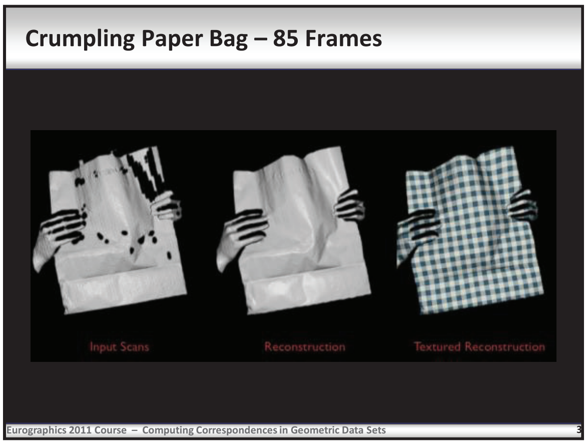


More Results



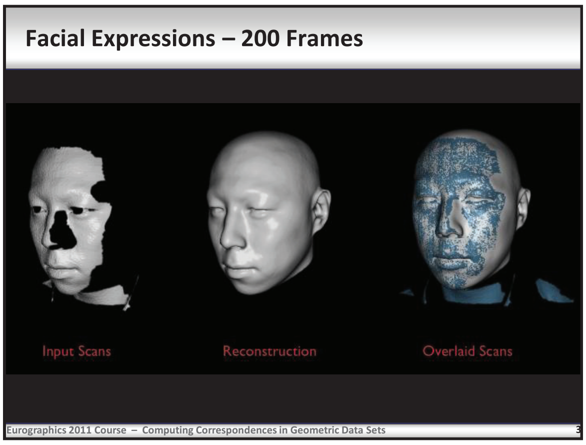
Grasping Hand – 34 Frames

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets



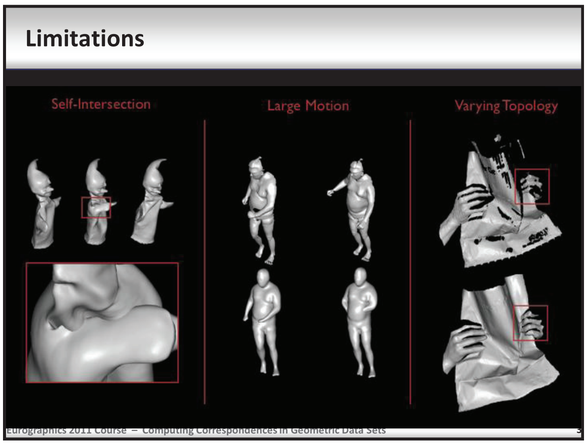
Crumpling Paper Bag – 85 Frames

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets



Facial Expressions – 200 Frames

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets



Limitations

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets

What's Next?

[Vlasic et al. '09], Princeton, ICT

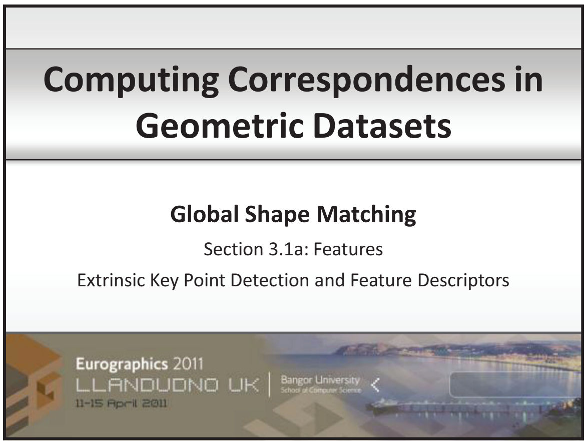
National Geographic

WordPress

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets

www.hao-li.com

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 3



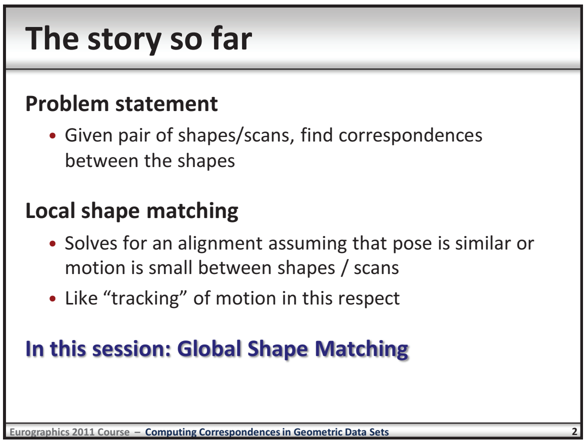
Computing Correspondences in Geometric Datasets

Global Shape Matching

Section 3.1a: Features

Extrinsic Key Point Detection and Feature Descriptors

Eurographics 2011
LLANDUDNO UK
11-15 April 2011
Bangor University
School of Computer Science



The story so far

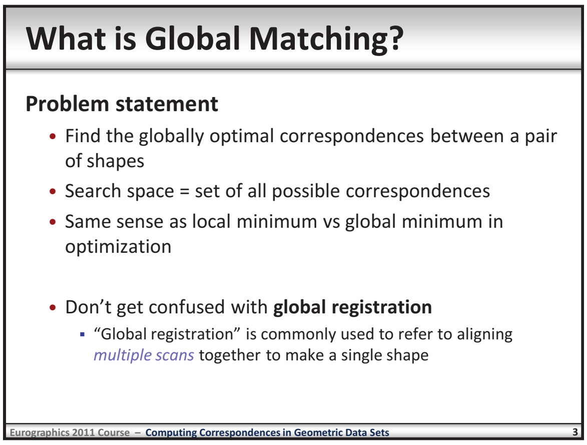
Problem statement

- Given pair of shapes/scans, find correspondences between the shapes

Local shape matching

- Solves for an alignment assuming that pose is similar or motion is small between shapes / scans
- Like "tracking" of motion in this respect

In this session: Global Shape Matching

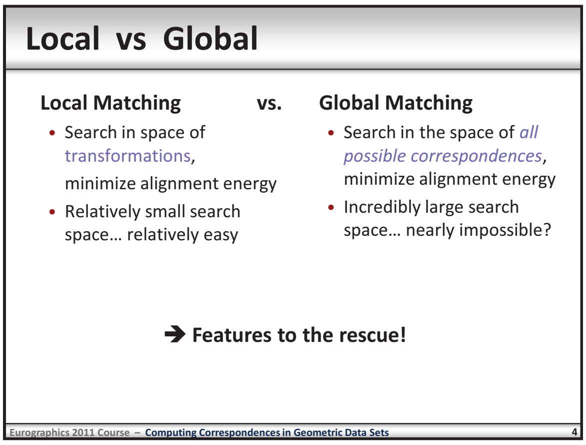


What is Global Matching?

Problem statement

- Find the globally optimal correspondences between a pair of shapes
- Search space = set of all possible correspondences
- Same sense as local minimum vs global minimum in optimization

- Don't get confused with **global registration**
  - "Global registration" is commonly used to refer to aligning *multiple scans* together to make a single shape



Local vs Global

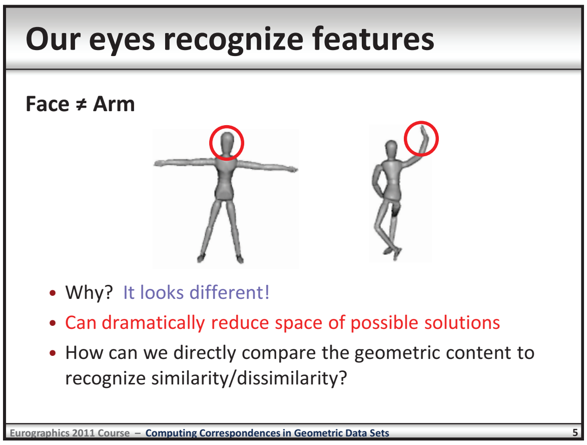
Local Matching vs.

- Search in space of transformations, minimize alignment energy
- Relatively small search space… relatively easy

Global Matching

- Search in the space of *all possible correspondences*, minimize alignment energy
- Incredibly large search space… nearly impossible?

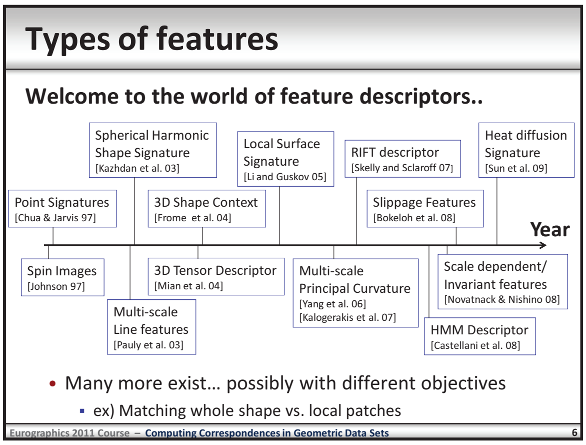
➔ **Features to the rescue!**



Our eyes recognize features

Face ≠ Arm

- Why? It looks different!
- Can dramatically reduce space of possible solutions
- How can we directly compare the geometric content to recognize similarity/dissimilarity?



Types of features

Welcome to the world of feature descriptors..

Point Signatures [Chua & Jarvis 97]

Spin Images [Johnson 97]

Spherical Harmonic Shape Signature [Kazhdan et al. 03]

Multi-scale Line features [Pauly et al. 03]

3D Shape Context [Frome et al. 04]

3D Tensor Descriptor [Mian et al. 04]

Local Surface Signature [Li and Guskov 05]

Multi-scale Principal Curvature [Yang et al. 06] [Kalogerakis et al. 07]

RIFT descriptor [Skelly and Sclaroff 07]

HMM Descriptor [Castellani et al. 08]

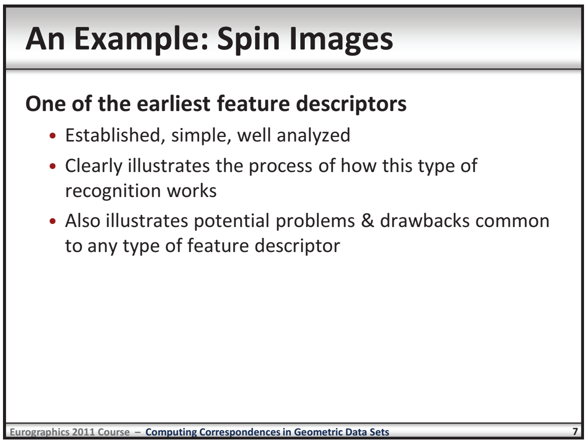
Scale dependent/ Invariant features [Novatnack & Nishino 08]

Slippage Features [Bokeloh et al. 08]

Heat diffusion Signature [Sun et al. 09]

Year

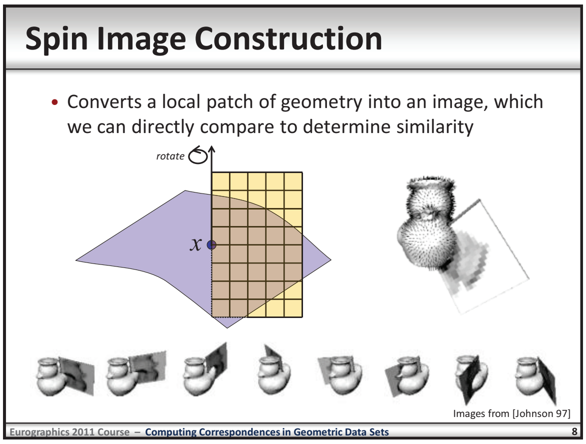
- Many more exist… possibly with different objectives
  - ex) Matching whole shape vs. local patches



An Example: Spin Images

One of the earliest feature descriptors

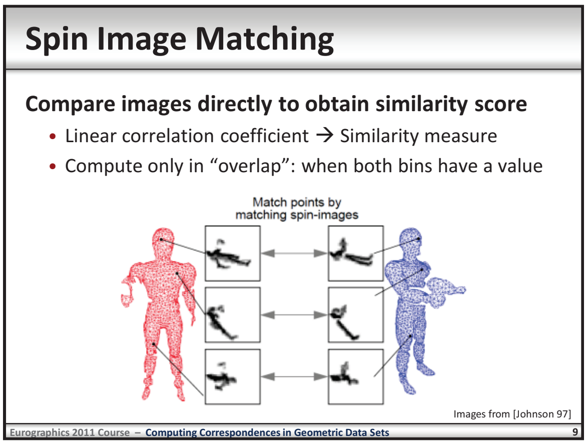
- Established, simple, well analyzed
- Clearly illustrates the process of how this type of recognition works
- Also illustrates potential problems & drawbacks common to any type of feature descriptor



Spin Image Construction

- Converts a local patch of geometry into an image, which we can directly compare to determine similarity

Images from [Johnson 97]

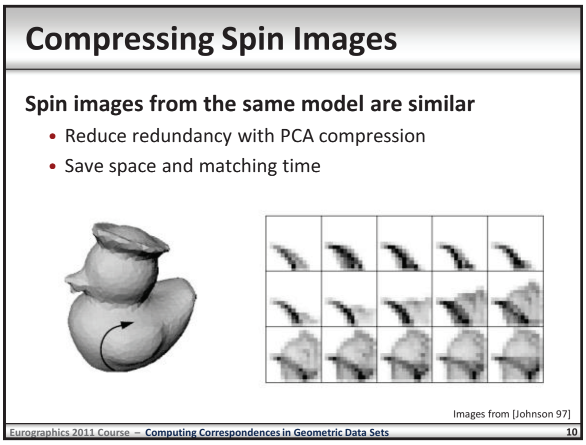


Spin Image Matching

Compare images directly to obtain similarity score

- Linear correlation coefficient → Similarity measure
- Compute only in "overlap": when both bins have a value

Images from [Johnson 97]

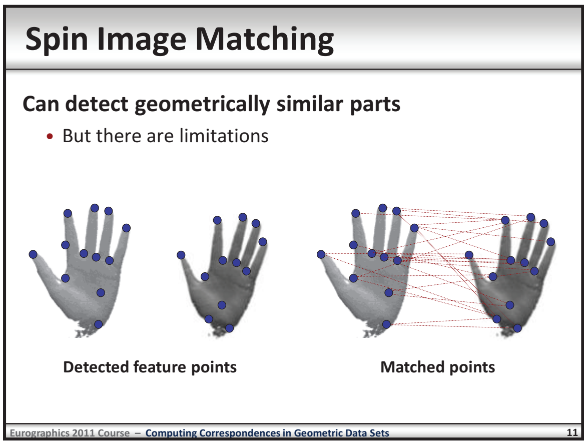


Compressing Spin Images

Spin images from the same model are similar

- Reduce redundancy with PCA compression
- Save space and matching time

Images from [Johnson 97]



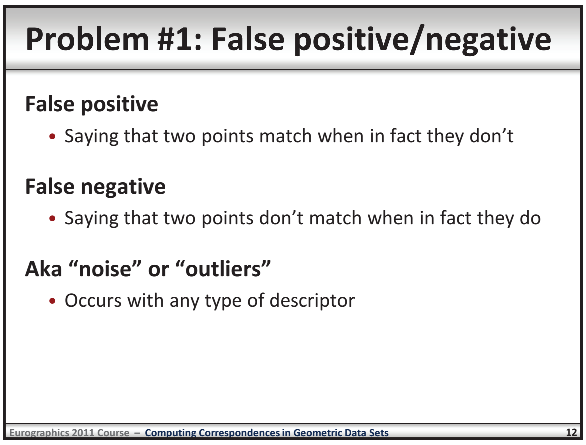
Spin Image Matching

Can detect geometrically similar parts

- But there are limitations

Detected feature points

Matched points



Problem #1: False positive/negative

False positive

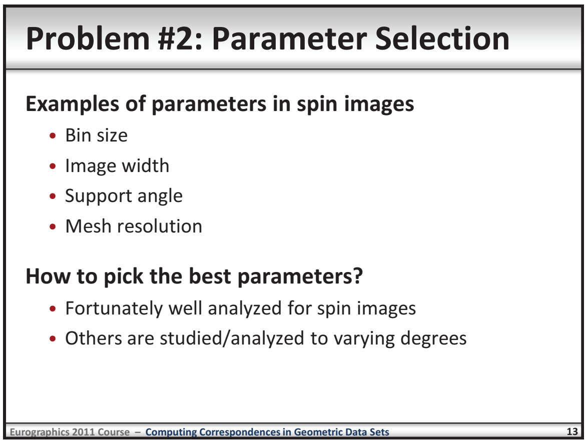
- Saying that two points match when in fact they don't

False negative

- Saying that two points don't match when in fact they do

Aka "noise" or "outliers"

- Occurs with any type of descriptor



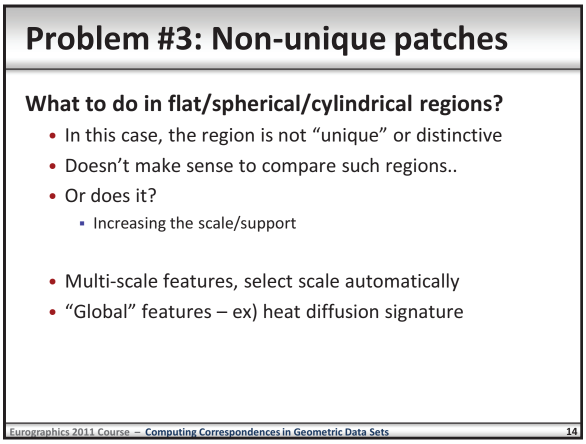
Problem #2: Parameter Selection

Examples of parameters in spin images

- Bin size
- Image width
- Support angle
- Mesh resolution

How to pick the best parameters?

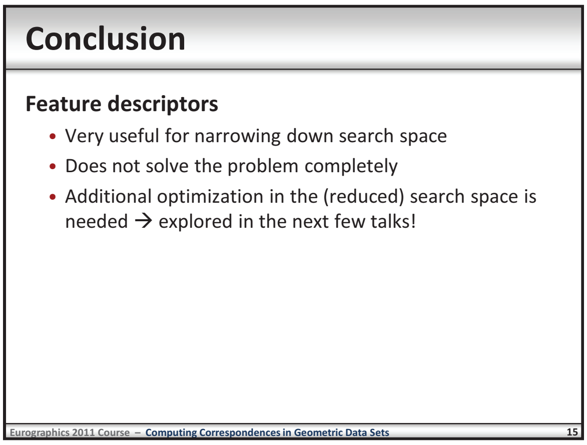
- Fortunately well analyzed for spin images
- Others are studied/analyzed to varying degrees



Problem #3: Non-unique patches

What to do in flat/spherical/cylindrical regions?

- In this case, the region is not “unique” or distinctive
- Doesn’t make sense to compare such regions..
- Or does it?
  - Increasing the scale/support
- Multi-scale features, select scale automatically
- “Global” features – ex) heat diffusion signature



Conclusion

Feature descriptors

- Very useful for narrowing down search space
- Does not solve the problem completely
- Additional optimization in the (reduced) search space is needed → explored in the next few talks!

Computing Correspondences in Geometric Datasets

Global Shape Matching

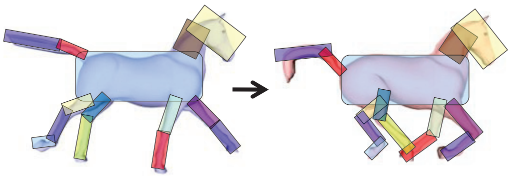
Section 3.3: Global, Articulated, Pairwise

Graph Cut Based Matching

Articulated Shape Matching

Movement consists of few parts

- Material so far focused on matching individual corresp
- **Now: point groups move together**
  - Each group according to a single rigid transformation



How can we simplify the problem?

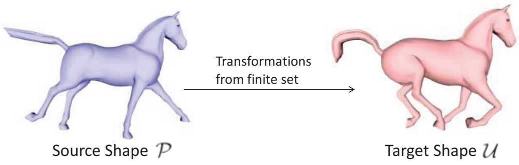
- **Before:** Optimizing correspondences of individual points
- **Articulated:** Optimizing correspondence of groups of points

- Q) What are the groups?
  - Generally: don't know in advance.
  - *If we know in advance: [PG08]*

- Q) What is the motion for each group?
  - We can guess well
  - *ICP based search, feature based search*

Basic idea

- If we know the articulated movement (small set of transformations ***{T}***)
- **Reformulate optimization**
  - Find an assignment of transformations to the points that "minimizes registration error"

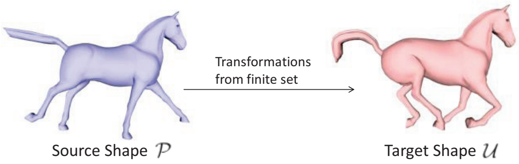

Transformations from finite set
Source Shape $\mathcal{P}$
Target Shape $\mathcal{U}$

Basic idea

Find the assignment of transformations in *{T}* to points in P, that maximizes:

$$P^{(match)}(x_1,\ldots,x_n)=\prod_{i=1}^{n}P_i^{(single)}\prod_{i,j=1}^{n}P_{i,j}^{(compatible)},x_i\in\{T\}$$

"Data" and "Smoothness" terms evaluate quality of assignment


Transformations from finite set
Source Shape $\mathcal{P}$
Target Shape $\mathcal{U}$

How to find transformations?

Global search / feature matching strategy [CZ08]

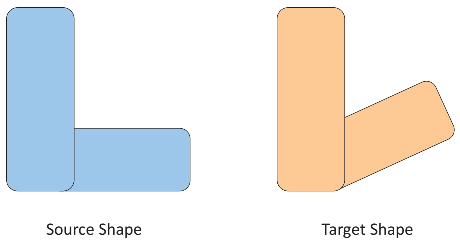
- Sample transformations in advance by feature matching
- Inspired by partial symmetry detection [MGP06]
  - Covered later in the course!

Local search / refinement strategy [CZ09]

- Start with initial part labeling, keep refining transformations of each part via ICP
- Refine part labels using transformations, repeat alternation

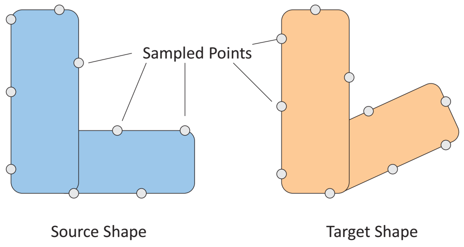
Motion Sampling Illustration

Find transformations that move parts of the source to parts of the target



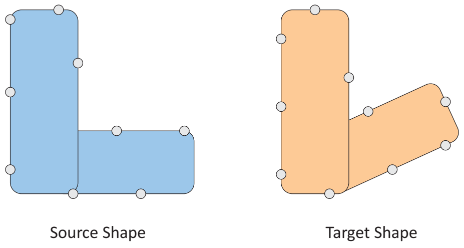
Motion Sampling Illustration

Find transformations that move parts of the source to parts of the target



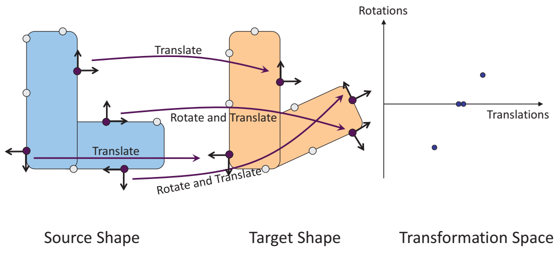
Motion Sampling Illustration

Find transformations that move parts of the source to parts of the target



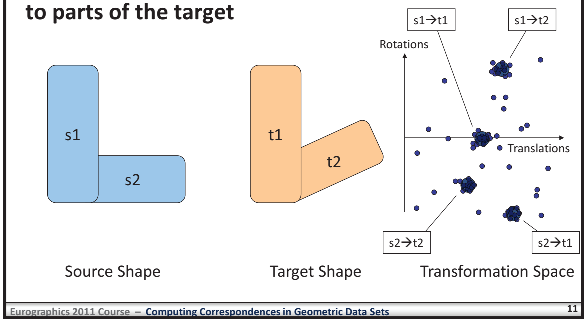
Motion Sampling Illustration

Find transformations that move parts of the source to parts of the target



Motion Sampling Illustration

Find transformations that move parts of the source to parts of the target



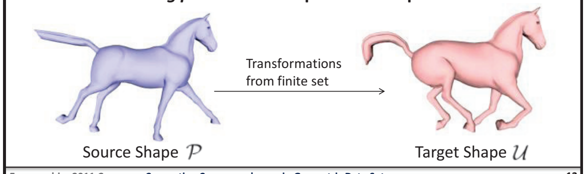
Basic idea

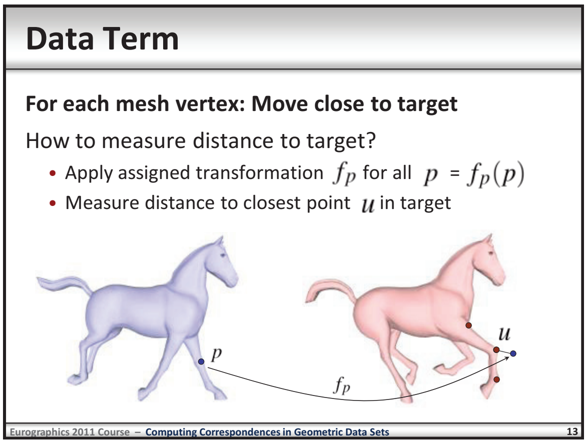
Find the assignment of transformations in *{T}* to points in P, that maximizes:

$$P^{(match)}(x_1,\ldots,x_n)=\prod_{i=1}^{n}P_i^{(single)}\prod_{i,j=1}^{n}P_{i,j}^{(compatible)},x_i\in\{T\}$$

"Data" and "Smoothness" terms evaluate quality of assignment

A *discrete labelling problem* → Graph Cuts for optimization


Transformations from finite set
Source Shape $\mathcal{P}$
Target Shape $\mathcal{U}$

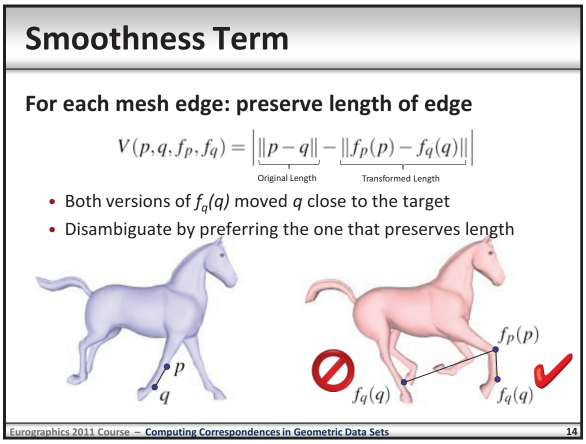


Data Term

For each mesh vertex: Move close to target

How to measure distance to target?

- Apply assigned transformation f_p for all $p = f_p(p)$
- Measure distance to closest point u in target

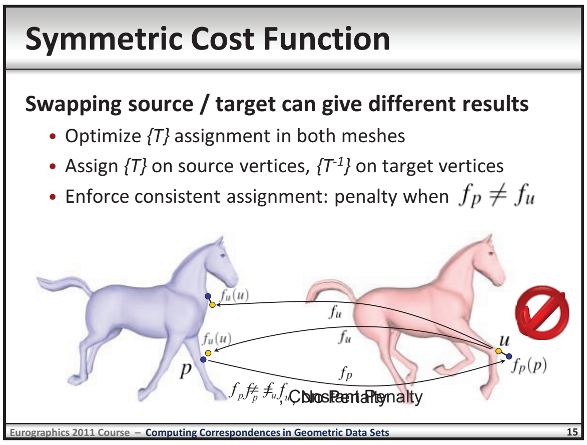


Smoothness Term

For each mesh edge: preserve length of edge

$$V(p,q,f_p,f_q) = \Big| \underbrace{\|p-q\|}_{\text{Original Length}} - \underbrace{\|f_p(p)-f_q(q)\|}_{\text{Transformed Length}} \Big|$$

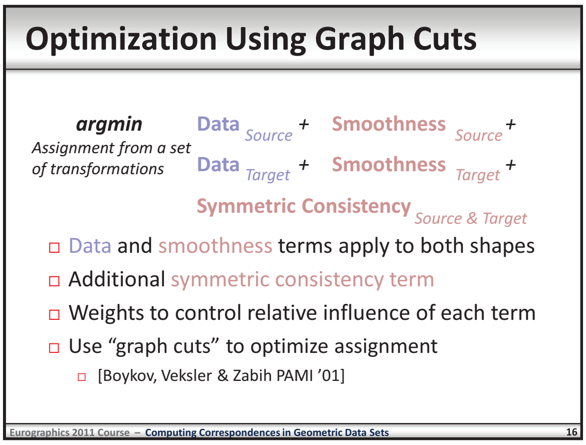
- Both versions of $f_q(q)$ moved q close to the target
- Disambiguate by preferring the one that preserves length



Symmetric Cost Function

Swapping source / target can give different results

- Optimize *{T}* assignment in both meshes
- Assign *{T}* on source vertices, *{T^{-1}}* on target vertices
- Enforce consistent assignment: penalty when $f_p \neq f_u$

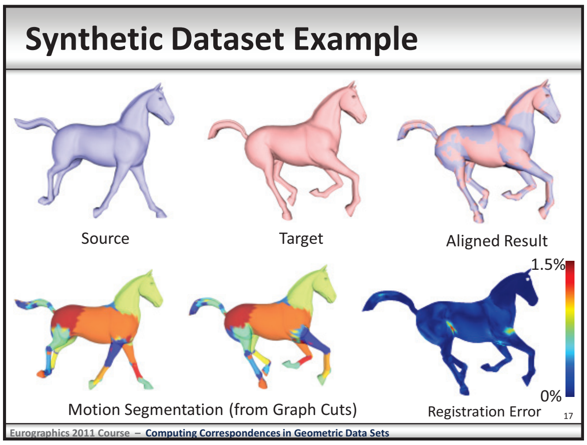


Optimization Using Graph Cuts

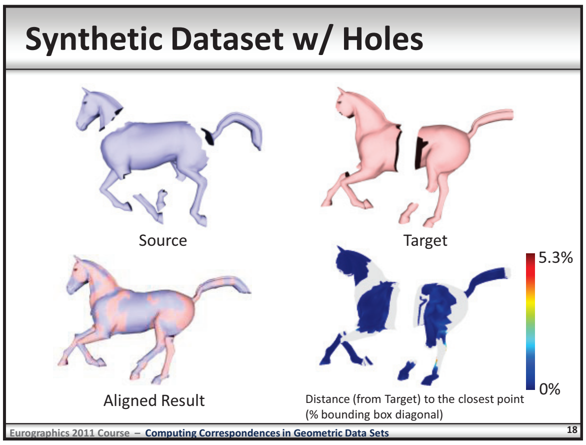
argmin
Assignment from a set of transformations

Data *Source* + **Smoothness** *Source* +
Data *Target* + **Smoothness** *Target* +
Symmetric Consistency *Source & Target*

- □ Data and smoothness terms apply to both shapes
- □ Additional symmetric consistency term
- □ Weights to control relative influence of each term
- □ Use “graph cuts” to optimize assignment
  - □ [Boykov, Veksler & Zabih PAMI ’01]

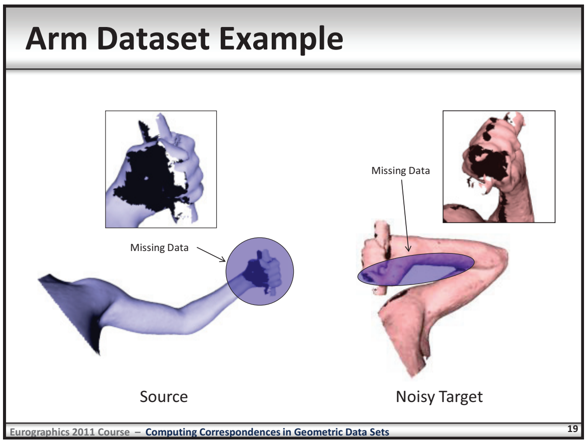


Synthetic Dataset Example

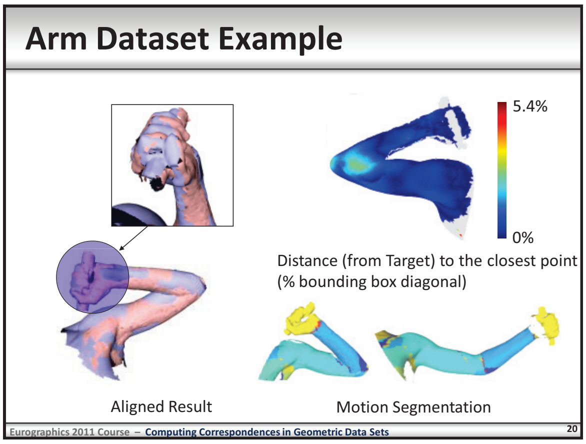


Synthetic Dataset w/ Holes

Source
Target
Aligned Result
5.3%
0%
Distance (from Target) to the closest point
(% bounding box diagonal)

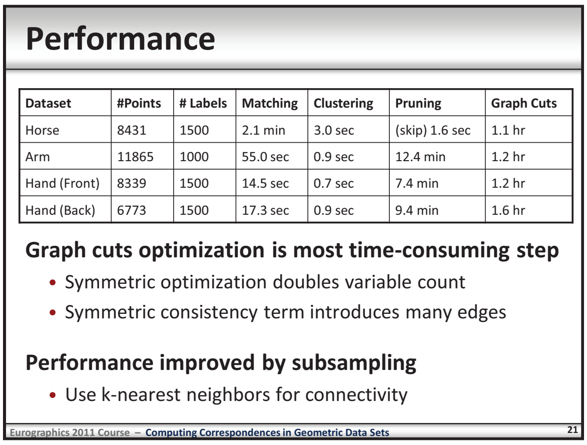


Arm Dataset Example



Arm Dataset Example

5.4%
0%
Distance (from Target) to the closest point
(% bounding box diagonal)
Aligned Result
Motion Segmentation



Performance

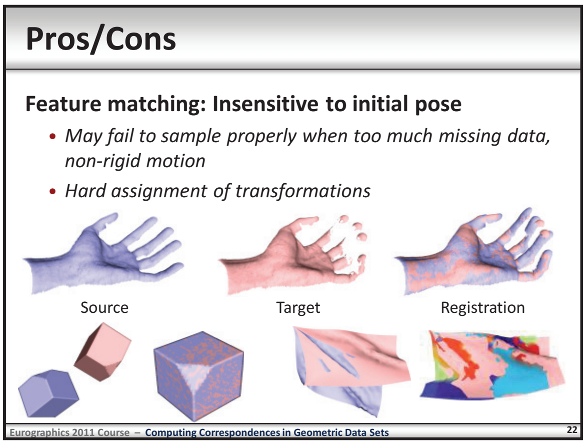
Dataset	#Points	# Labels	Matching	Clustering	Pruning	Graph Cuts
Horse	8431	1500	2.1 min	3.0 sec	(skip) 1.6 sec	1.1 hr
Arm	11865	1000	55.0 sec	0.9 sec	12.4 min	1.2 hr
Hand (Front)	8339	1500	14.5 sec	0.7 sec	7.4 min	1.2 hr
Hand (Back)	6773	1500	17.3 sec	0.9 sec	9.4 min	1.6 hr

Graph cuts optimization is most time-consuming step

- Symmetric optimization doubles variable count
- Symmetric consistency term introduces many edges

Performance improved by subsampling

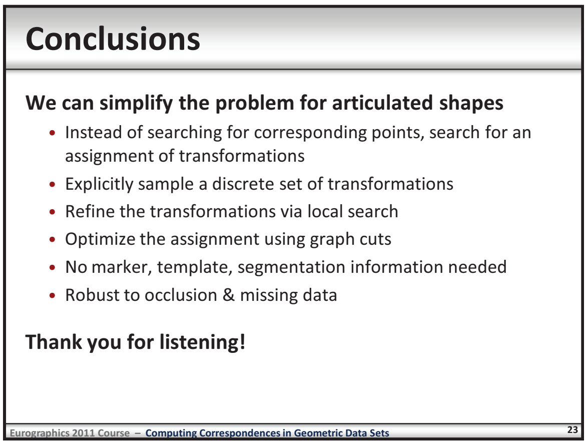
- Use k-nearest neighbors for connectivity



Pros/Cons

Feature matching: Insensitive to initial pose

- *May fail to sample properly when too much missing data, non-rigid motion*
- *Hard assignment of transformations*

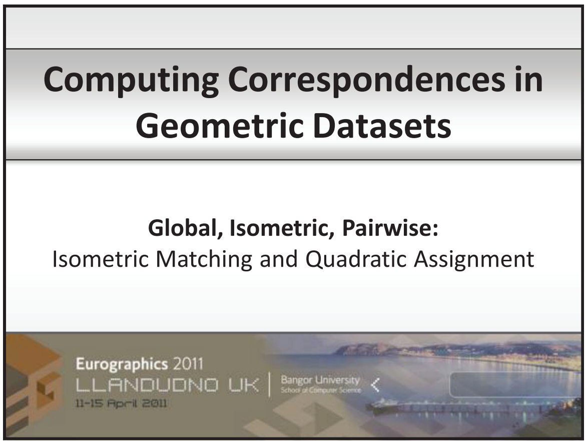


Conclusions

We can simplify the problem for articulated shapes

- Instead of searching for corresponding points, search for an assignment of transformations
- Explicitly sample a discrete set of transformations
- Refine the transformations via local search
- Optimize the assignment using graph cuts
- No marker, template, segmentation information needed
- Robust to occlusion & missing data

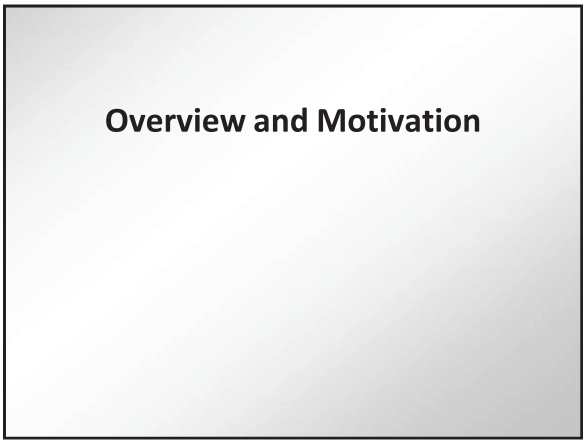
Thank you for listening!



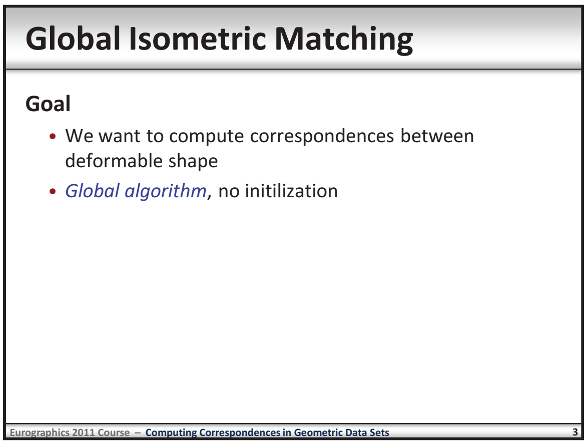
Computing Correspondences in Geometric Datasets

Global, Isometric, Pairwise:
Isometric Matching and Quadratic Assignment

Eurographics 2011
LLANDUDNO UK
Bangor University
School of Computer Science
11-15 April 2011



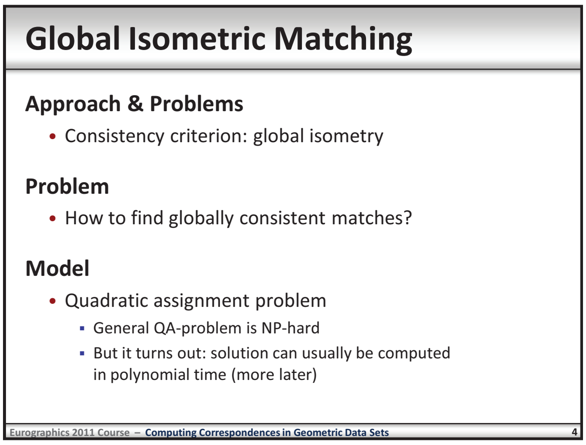
Overview and Motivation



Global Isometric Matching

Goal

- We want to compute correspondences between deformable shape
- *Global algorithm*, no initilization



Global Isometric Matching

Approach & Problems

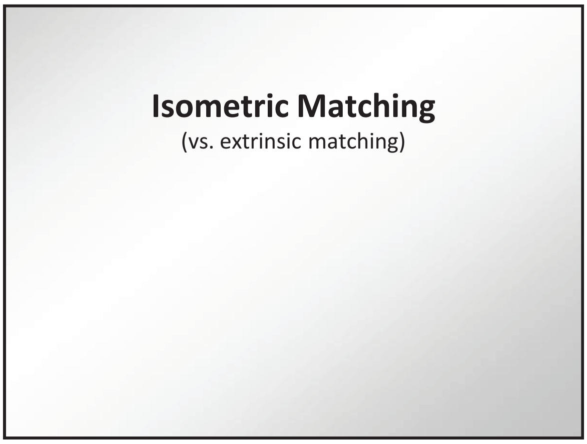
- Consistency criterion: global isometry

Problem

- How to find globally consistent matches?

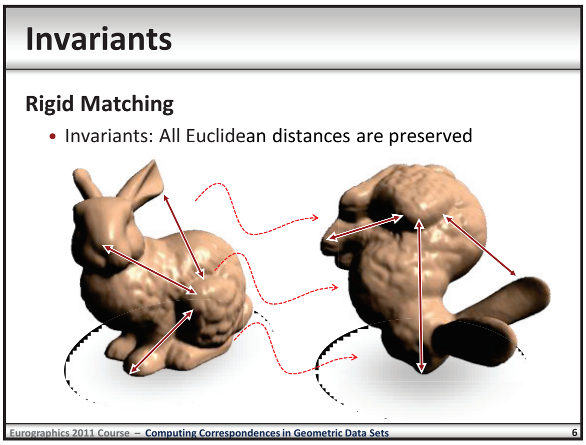
Model

- Quadratic assignment problem
  - General QA-problem is NP-hard
  - But it turns out: solution can usually be computed in polynomial time (more later)



Isometric Matching

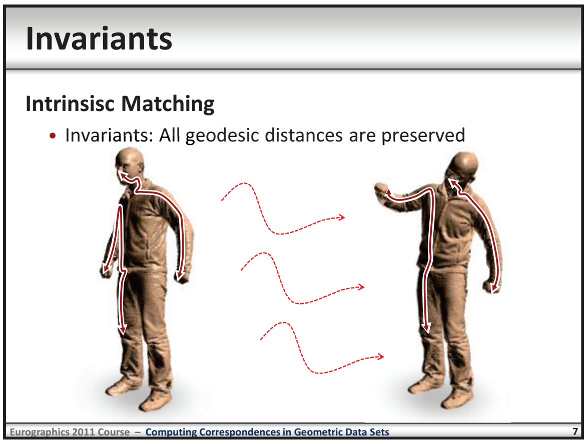
(vs. extrinsic matching)



Invariants

Rigid Matching

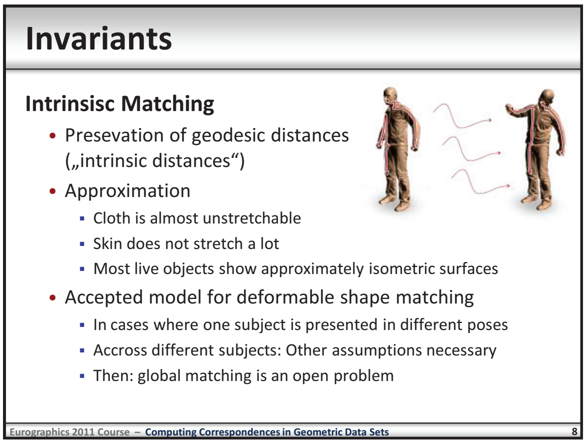
- Invariants: All Euclidean distances are preserved



Invariants

Intrinsisc Matching

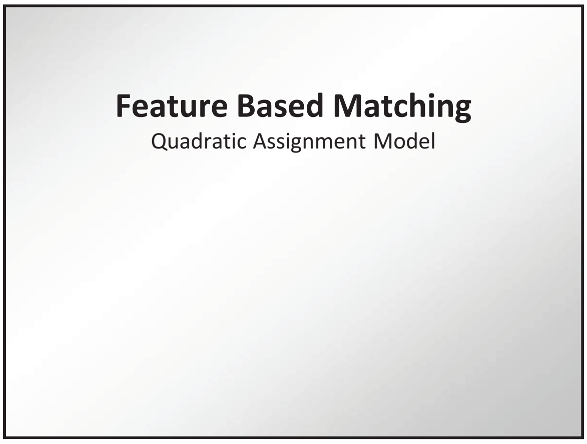
- Invariants: All geodesic distances are preserved



Invariants

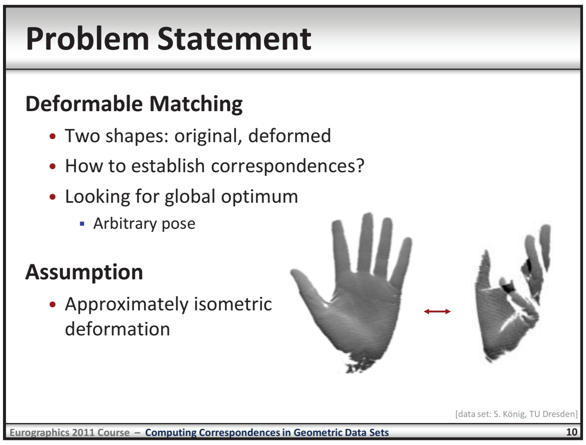
Intrinsisc Matching

- Presevation of geodesic distances („intrinsic distances“)
- Approximation
  - Cloth is almost unstretchable
  - Skin does not stretch a lot
  - Most live objects show approximately isometric surfaces
- Accepted model for deformable shape matching
  - In cases where one subject is presented in different poses
  - Accross different subjects: Other assumptions necessary
  - Then: global matching is an open problem



Feature Based Matching

Quadratic Assignment Model



Problem Statement

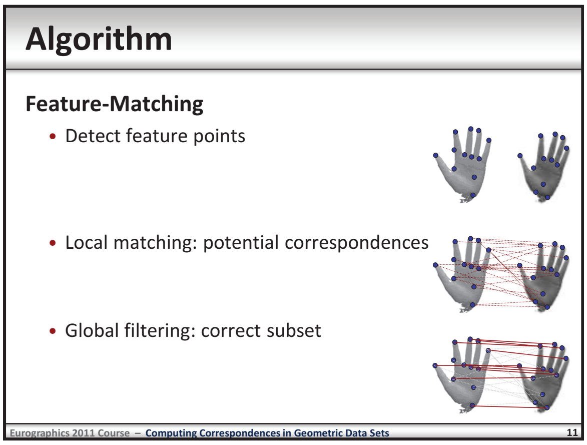
Deformable Matching

- Two shapes: original, deformed
- How to establish correspondences?
- Looking for global optimum
  - Arbitrary pose

Assumption

- Approximately isometric deformation

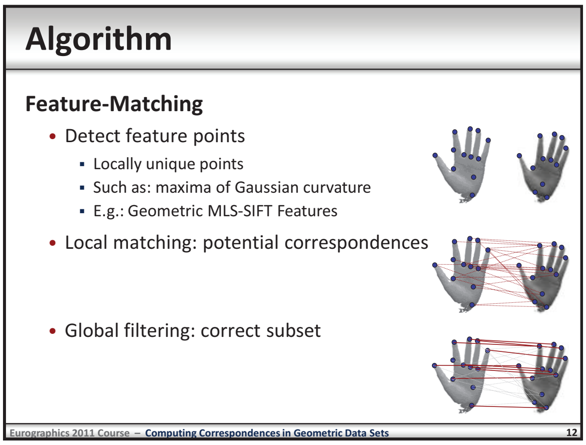
[data set: S. König, TU Dresden]



Algorithm

Feature-Matching

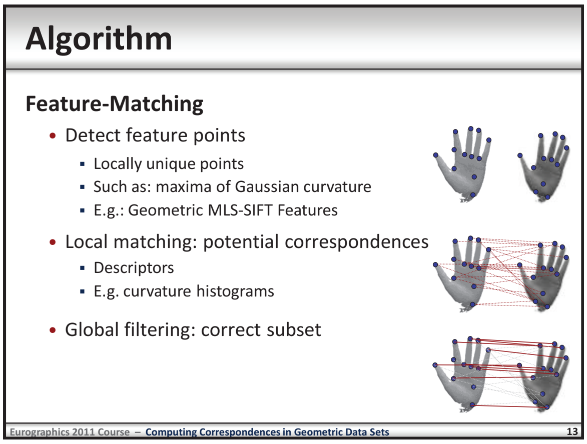
- Detect feature points
- Local matching: potential correspondences
- Global filtering: correct subset



Algorithm

Feature-Matching

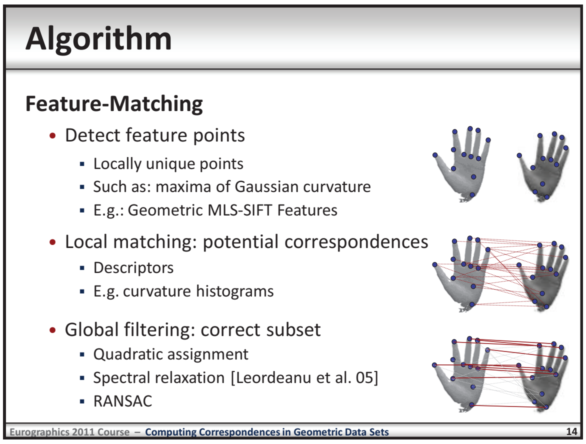
- Detect feature points
  - Locally unique points
  - Such as: maxima of Gaussian curvature
  - E.g.: Geometric MLS-SIFT Features
- Local matching: potential correspondences
- Global filtering: correct subset



Algorithm

Feature-Matching

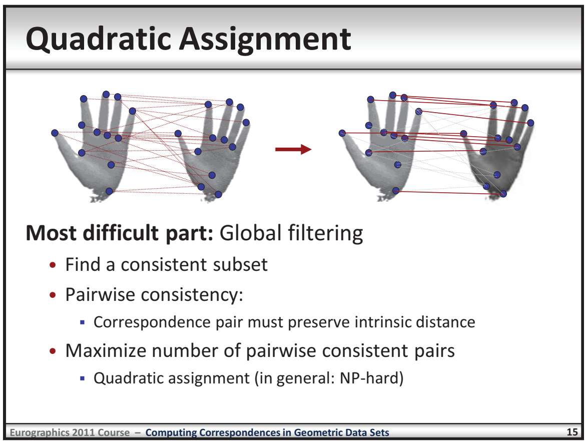
- Detect feature points
  - Locally unique points
  - Such as: maxima of Gaussian curvature
  - E.g.: Geometric MLS-SIFT Features
- Local matching: potential correspondences
  - Descriptors
  - E.g. curvature histograms
- Global filtering: correct subset



Algorithm

Feature-Matching

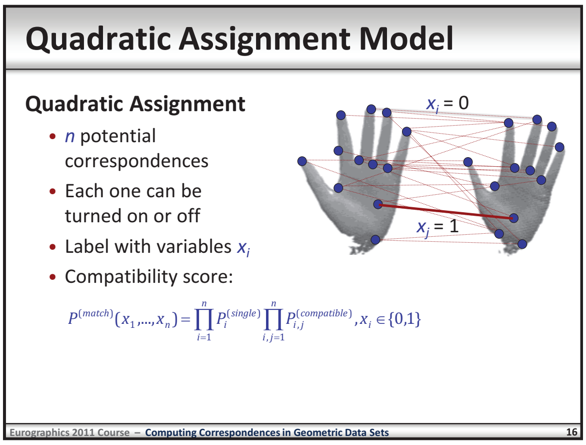
- Detect feature points
  - Locally unique points
  - Such as: maxima of Gaussian curvature
  - E.g.: Geometric MLS-SIFT Features
- Local matching: potential correspondences
  - Descriptors
  - E.g. curvature histograms
- Global filtering: correct subset
  - Quadratic assignment
  - Spectral relaxation [Leordeanu et al. 05]
  - RANSAC



Quadratic Assignment

Most difficult part: Global filtering

- Find a consistent subset
- Pairwise consistency:
  - Correspondence pair must preserve intrinsic distance
- Maximize number of pairwise consistent pairs
  - Quadratic assignment (in general: NP-hard)

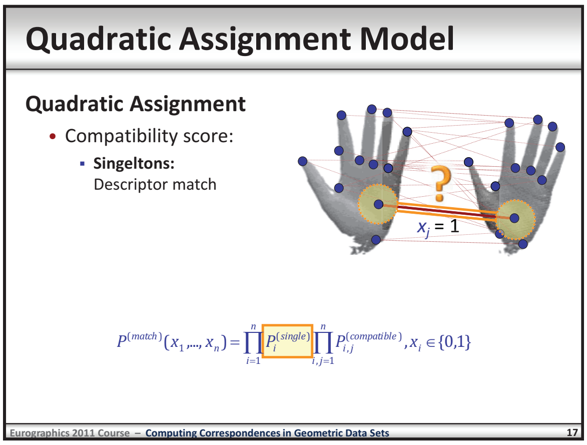


Quadratic Assignment Model

Quadratic Assignment

- *n* potential correspondences
- Each one can be turned on or off
- Label with variables x_i
- Compatibility score:

$$P^{(match)}(x_1,\dots,x_n) = \prod_{i=1}^{n} P_i^{(single)} \prod_{i,j=1}^{n} P_{i,j}^{(compatible)}, x_i \in \{0,1\}$$

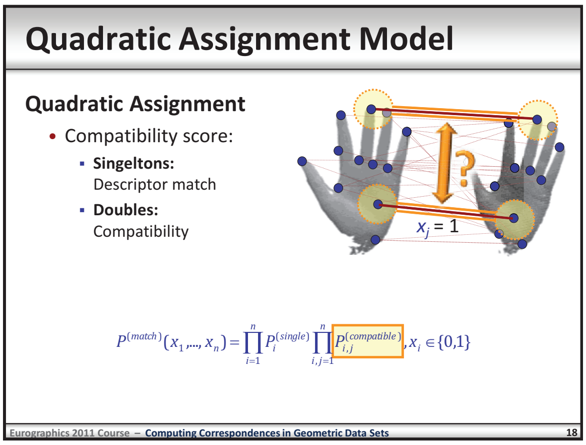


Quadratic Assignment Model

Quadratic Assignment

- Compatibility score:
  - **Singeltons:** Descriptor match

$$P^{(match)}(x_1,\dots,x_n) = \prod_{i=1}^{n} \boxed{P_i^{(single)}} \prod_{i,j=1}^{n} P_{i,j}^{(compatible)}, x_i \in \{0,1\}$$

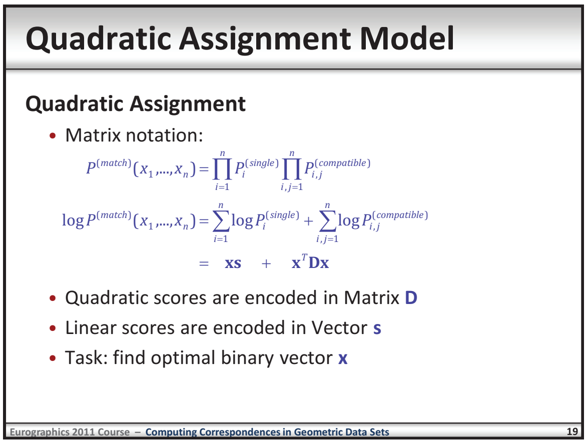


Quadratic Assignment Model

Quadratic Assignment

- Compatibility score:
  - **Singeltons:** Descriptor match
  - **Doubles:** Compatibility

$$P^{(match)}(x_1,\dots,x_n) = \prod_{i=1}^{n} P_i^{(single)} \prod_{i,j=1}^{n} \boxed{P_{i,j}^{(compatible)}}, x_i \in \{0,1\}$$



Quadratic Assignment Model

Quadratic Assignment

- Matrix notation:

$$P^{(match)}(x_1,\ldots,x_n) = \prod_{i=1}^{n} P_i^{(single)} \prod_{i,j=1}^{n} P_{i,j}^{(compatible)}$$

$$\log P^{(match)}(x_1,\ldots,x_n) = \sum_{i=1}^{n} \log P_i^{(single)} + \sum_{i,j=1}^{n} \log P_{i,j}^{(compatible)}$$

$$= \quad \mathbf{xs} \quad + \quad \mathbf{x}^T\mathbf{Dx}$$

- Quadratic scores are encoded in Matrix **D**
- Linear scores are encoded in Vector **s**
- Task: find optimal binary vector **x**

Computing Correspondences in Geometric Datasets

Global Shape Matching

Section 3.4b: Global, Isometric, Pairwise

Spectral Matching and Applications

Quadratic Assignment Model

Quadratic Assignment

- Matrix notation:

$$P^{(match)}(x_1,\ldots,x_n) = \prod_{i=1}^{n} P_i^{(single)} \prod_{i,j=1}^{n} P_{i,j}^{(compatible)}$$

$$\log P^{(match)}(x_1,\ldots,x_n) = \sum_{i=1}^{n} \log P_i^{(single)} + \sum_{i,j=1}^{n} \log P_{i,j}^{(compatible)}$$

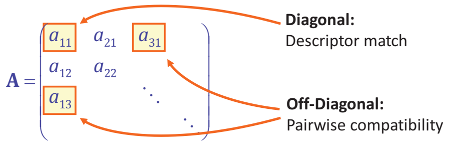
$$= \quad \mathbf{xs} \quad + \quad \mathbf{x}^T\mathbf{Dx}$$

- Quadratic scores are encoded in Matrix **D**
- Linear scores are encoded in Vector **s**
- Task: find optimal binary vector **x**

Spectral Matching

Simple & Effective Approximation:

- Spectral matching [Leordeanu & Hebert 05]
- Form compatibility matrix:



Diagonal:
Descriptor match

Off-Diagonal:
Pairwise compatibility

All entries within [0..1]
= [no match...perfect match]

Spectral Matching

Approximate largest clique:

- Compute eigenvector with largest eigenvalue
- Maximizes Rayleigh quotient:

$$\arg\max \frac{\mathbf{x}^T\mathbf{Ax}}{\|\mathbf{x}\|^2}$$

- "Best yield" for bounded norm
  - The more consistent pairs (rows of 1s), the better
  - Approximates largest clique
- Implementation
  - For example: power iteration

Spectral Matching

Post-processing

- Greedy quantization
  - Select largest remaining entry, set it to 1
  - Set all entries to 0 that are not pairwise consistent with current set
  - Iterate until all entries are quantized

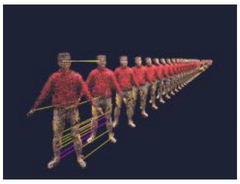
In practice...

- This algorithm turns out to work quite well.
- Very easy to implement
- Limited to (approx.) quadratic assignment model

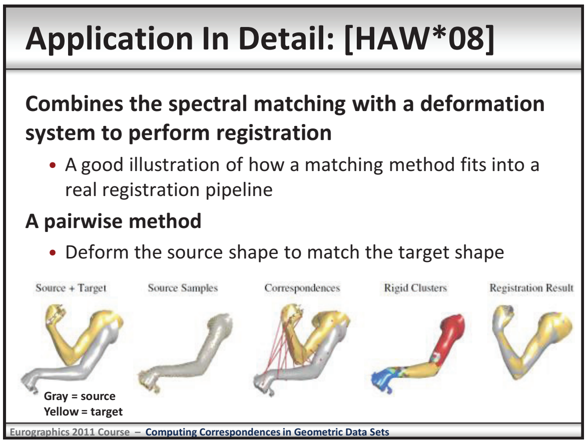
Spectral Matching Example

Application to Animations

- **Feature points:**
  Geometric MLS-SIFT features [Li et al. 2005]
- **Descriptors:**
  Curvature & color ring histograms
- **Global Filtering:**
  Spectral matching
- **Pairwise animation matching:**
  Low precision passive stereo data



Data courtesy of C. Theobald, MPI Informatik



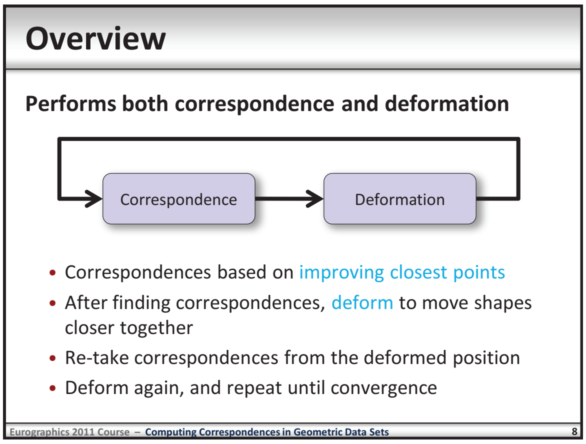
Application In Detail: [HAW*08]

Combines the spectral matching with a deformation system to perform registration

- A good illustration of how a matching method fits into a real registration pipeline

A pairwise method

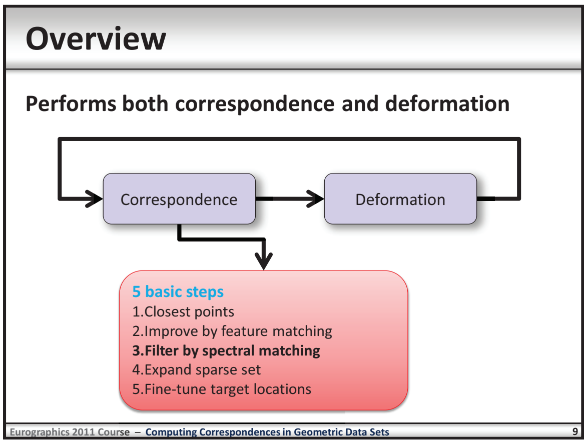
- Deform the source shape to match the target shape



Overview

Performs both correspondence and deformation

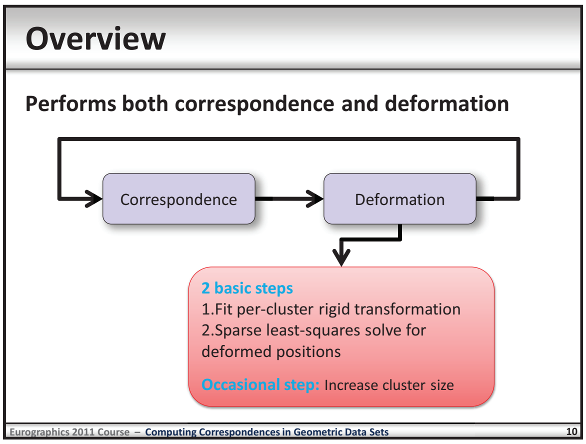
- Correspondences based on improving closest points
- After finding correspondences, deform to move shapes closer together
- Re-take correspondences from the deformed position
- Deform again, and repeat until convergence



Overview

Performs both correspondence and deformation

5 basic steps
1. Closest points
2. Improve by feature matching
3. **Filter by spectral matching**
4. Expand sparse set
5. Fine-tune target locations

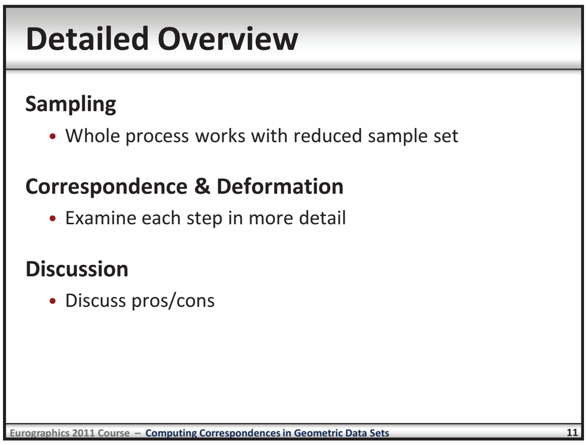


Overview

Performs both correspondence and deformation

2 basic steps
1. Fit per-cluster rigid transformation
2. Sparse least-squares solve for deformed positions

Occasional step: Increase cluster size



Detailed Overview

Sampling

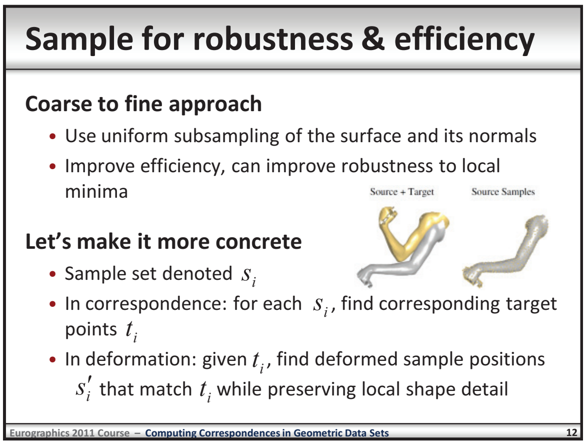
- Whole process works with reduced sample set

Correspondence & Deformation

- Examine each step in more detail

Discussion

- Discuss pros/cons



Sample for robustness & efficiency

Coarse to fine approach

- Use uniform subsampling of the surface and its normals
- Improve efficiency, can improve robustness to local minima

Let's make it more concrete

- Sample set denoted s_i
- In correspondence: for each s_i, find corresponding target points t_i
- In deformation: given t_i, find deformed sample positions s_i' that match t_i while preserving local shape detail

Correspondence Step #1

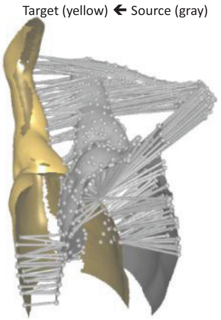
Find closest points

- For each source sample, find the closest target sample
  - s = sample point on source
  - t = sample point on target

$$\arg\min_{t \in \hat{T}} \|s - t\|^2$$

- Usually pretty bad

Target (yellow) 🡐 Source (gray)



Closest point correspondences

Correspondence Step #2

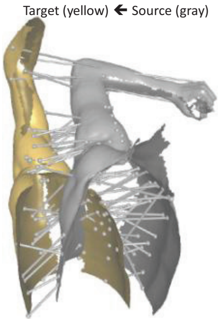
Improve by feature matching

- Search target's neighbors to see if there's better feature match, replace target
  - Let f(s) be feature value of s

$$t \leftarrow \arg\min_{t' \in N(t)} \|f(s) - f(t')\|^2$$

- Iterate until we stop moving
- If we move too much, discard correspondence
- Much better, but still outliers

Target (yellow) 🡐 Source (gray)



Feature-matched correspondences

Correspondence Step #3

Filter by spectral matching

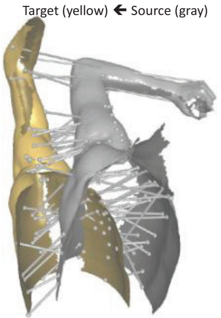
- ***(First some preprocessing)***
- Construct *k*-nn graph on both src & tgt sample set (*k* = 15)
- Length of shortest path on graph gives approx. geodesic distances on src & tgt

$$d_g(s_i, s_j) \quad d_g(t_i, t_j)$$

- Goal is to filter these ----------🡒

and keep a subset which is geodesically consistent

Target (yellow) 🡐 Source (gray)



Feature-matched correspondences

Correspondence Step #3

Filter by spectral matching

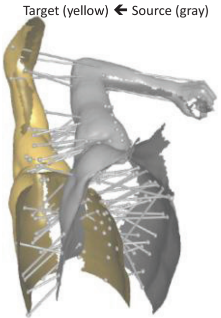
- Construct affinity matrix M using these shortest path distances
- Consistency term & matrix

$$c_{ij} = \min\{\frac{d_g(s_i, s_j)}{d_g(t_i, t_j)}, \frac{d_g(t_i, t_j)}{d_g(s_i, s_j)}\}, \ c_{ii} = 1$$

$$\mathbf{M}_{ij} = \begin{cases} (\frac{c_{ij} - c_0}{1 - c_0})^2 & c_{ij} > c_0, \\ 0 & \text{otherwise,} \end{cases}$$

  - Threshold c_0 = 0.7 gives how much error in consistency we are willing to accept

Target (yellow) 🡐 Source (gray)



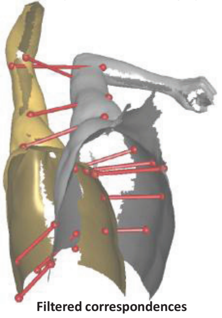
Feature-matched correspondences

Correspondence Step #3

Filter by spectral matching

- Apply spectral matching: find eigenvector with largest eigenvalue 🡒 score for each correspondence
- Iteratively add corresp. with largest score while consistency with the rest is above c_0
- Gives kernel correspondences
- Filtered matches usually sparse

Target (yellow) 🡐 Source (gray)



Filtered correspondences

Correspondence Step #4

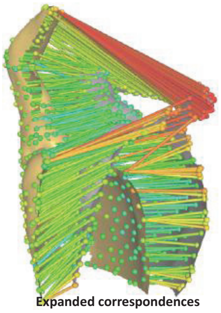
Expand sparse set

- Lots of samples have no target position
- For these, find best target position that respects geodesic distances to kernel set

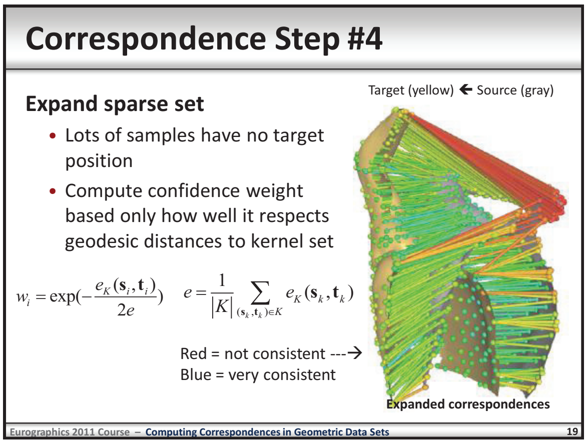
$$\mathbf{t}_i = \arg\min_{\mathbf{t} \in N_g(\mathbf{t}_j, \bar{T})} e_K(\mathbf{s}_i, \mathbf{t})$$

$$e_K(\mathbf{s}, \mathbf{t}) = \sum_{(\mathbf{s}_k, \mathbf{t}_k) \in K} \left[d_g(\mathbf{s}, \mathbf{s}_k) - d_g(\mathbf{t}, \mathbf{t}_k) \right]^2$$

Target (yellow) 🡐 Source (gray)



Expanded correspondences



Correspondence Step #4

Expand sparse set

- Lots of samples have no target position
- Compute confidence weight based only how well it respects geodesic distances to kernel set

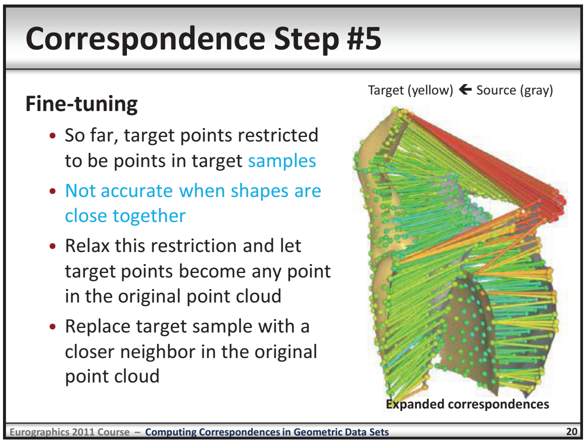
$$w_i = \exp(-\frac{e_K(\mathbf{s}_i, \mathbf{t}_i)}{2e}) \quad e = \frac{1}{|K|} \sum_{(\mathbf{s}_k, \mathbf{t}_k) \in K} e_K(\mathbf{s}_k, \mathbf{t}_k)$$

Red = not consistent ---→
Blue = very consistent

Target (yellow) ← Source (gray)

Expanded correspondences

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 19



Correspondence Step #5

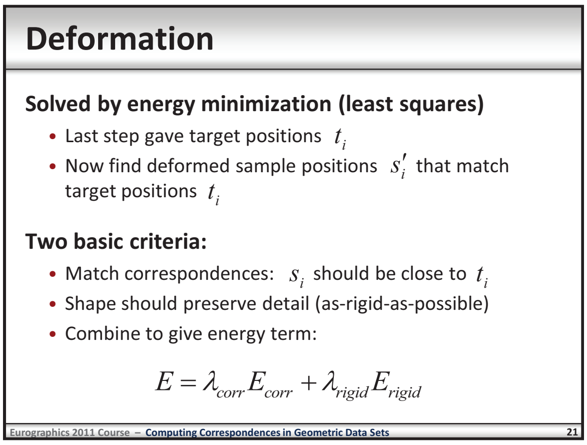
Fine-tuning

- So far, target points restricted to be points in target samples
- Not accurate when shapes are close together
- Relax this restriction and let target points become any point in the original point cloud
- Replace target sample with a closer neighbor in the original point cloud

Target (yellow) ← Source (gray)

Expanded correspondences

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 20



Deformation

Solved by energy minimization (least squares)

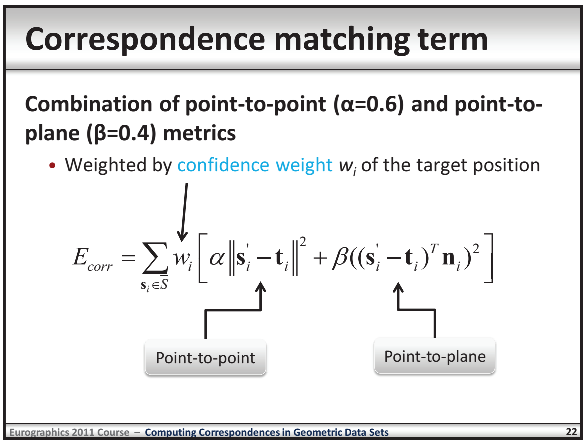
- Last step gave target positions t_i
- Now find deformed sample positions s_i' that match target positions t_i

Two basic criteria:

- Match correspondences: s_i should be close to t_i
- Shape should preserve detail (as-rigid-as-possible)
- Combine to give energy term:

$$E = \lambda_{corr} E_{corr} + \lambda_{rigid} E_{rigid}$$

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 21



Correspondence matching term

Combination of point-to-point (α=0.6) and point-to-plane (β=0.4) metrics

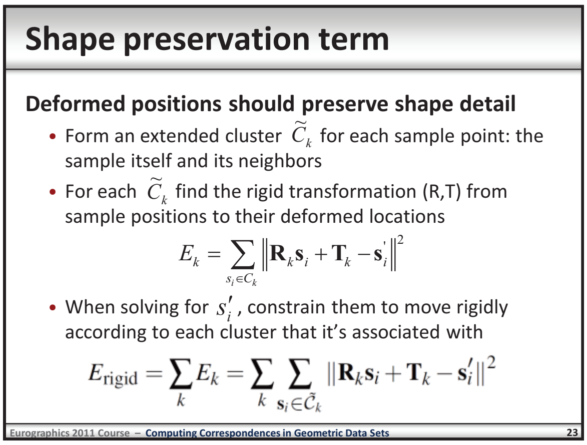
- Weighted by confidence weight w_i of the target position

$$E_{corr} = \sum_{\mathbf{s}_i \in \bar{S}} w_i \left[\alpha \left\| \mathbf{s}_i' - \mathbf{t}_i \right\|^2 + \beta((\mathbf{s}_i' - \mathbf{t}_i)^T \mathbf{n}_i)^2 \right]$$

Point-to-point

Point-to-plane

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 22



Shape preservation term

Deformed positions should preserve shape detail

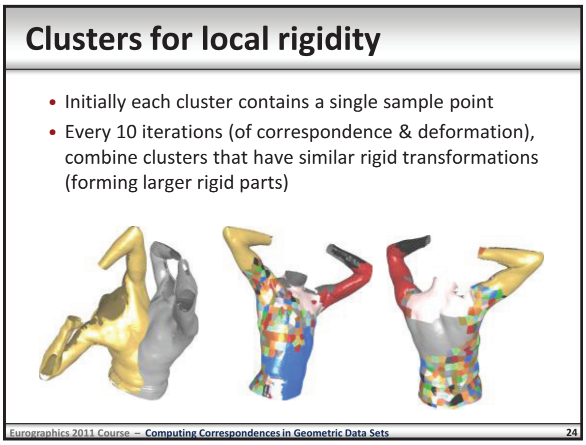
- Form an extended cluster $\widetilde{C}_k$ for each sample point: the sample itself and its neighbors
- For each $\widetilde{C}_k$ find the rigid transformation (R,T) from sample positions to their deformed locations

$$E_k = \sum_{s_i \in C_k} \left\| \mathbf{R}_k \mathbf{s}_i + \mathbf{T}_k - \mathbf{s}_i' \right\|^2$$

- When solving for s_i', constrain them to move rigidly according to each cluster that it’s associated with

$$E_{\text{rigid}} = \sum_k E_k = \sum_k \sum_{\mathbf{s}_i \in \tilde{C}_k} \|\mathbf{R}_k \mathbf{s}_i + \mathbf{T}_k - \mathbf{s}_i'\|^2$$

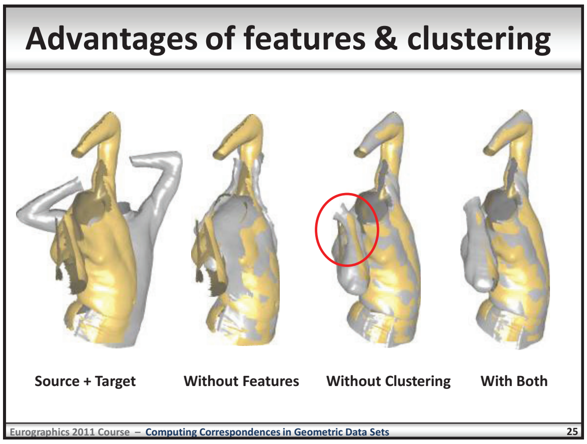
Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 23



Clusters for local rigidity

- Initially each cluster contains a single sample point
- Every 10 iterations (of correspondence & deformation), combine clusters that have similar rigid transformations (forming larger rigid parts)

Eurographics 2011 Course – Computing Correspondences in Geometric Data Sets 24



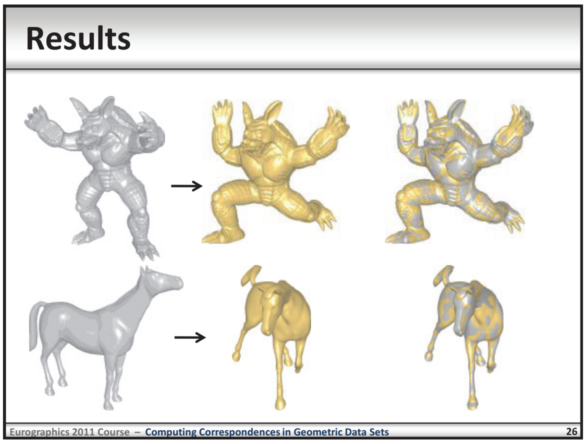
Advantages of features & clustering

Source + Target

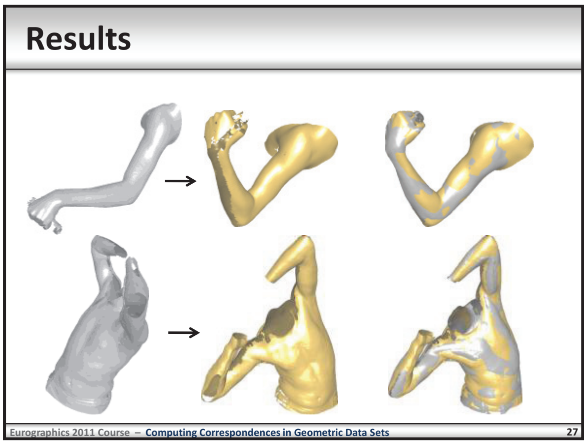
Without Features

Without Clustering

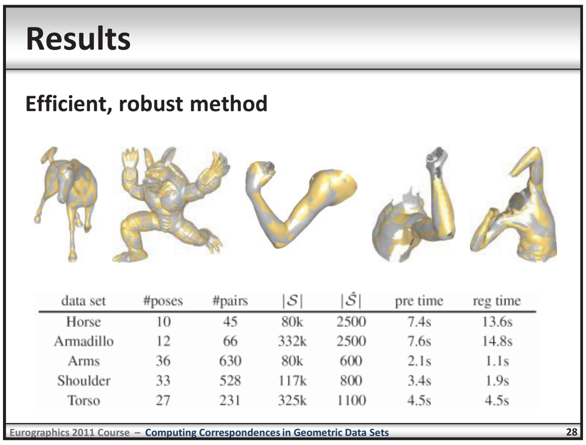
With Both



Results



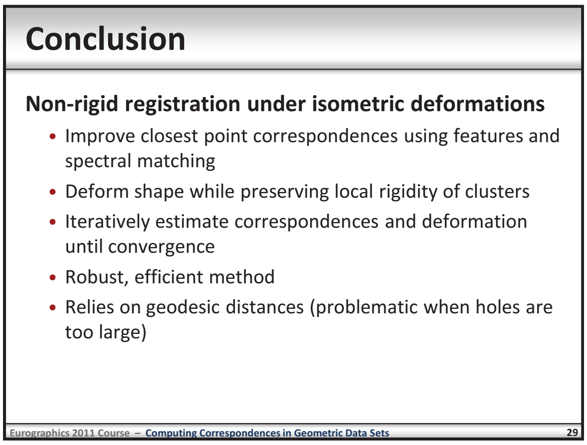
Results



Results

Efficient, robust method

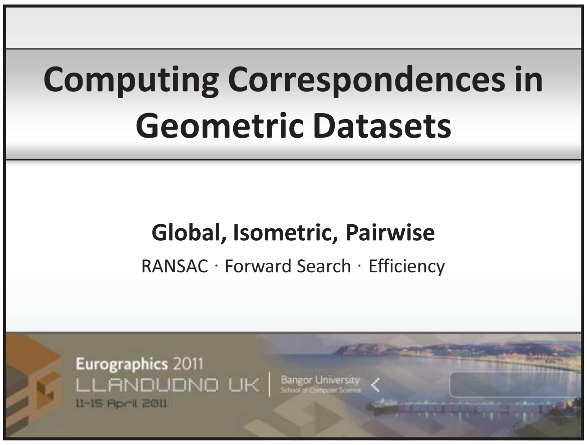
data set	#poses	#pairs	$\lvert\mathcal{S}\rvert$	$\lvert\hat{\mathcal{S}}\rvert$	pre time	reg time
Horse	10	45	80k	2500	7.4s	13.6s
Armadillo	12	66	332k	2500	7.6s	14.8s
Arms	36	630	80k	600	2.1s	1.1s
Shoulder	33	528	117k	800	3.4s	1.9s
Torso	27	231	325k	1100	4.5s	4.5s



Conclusion

Non-rigid registration under isometric deformations

- Improve closest point correspondences using features and spectral matching
- Deform shape while preserving local rigidity of clusters
- Iteratively estimate correspondences and deformation until convergence
- Robust, efficient method
- Relies on geodesic distances (problematic when holes are too large)

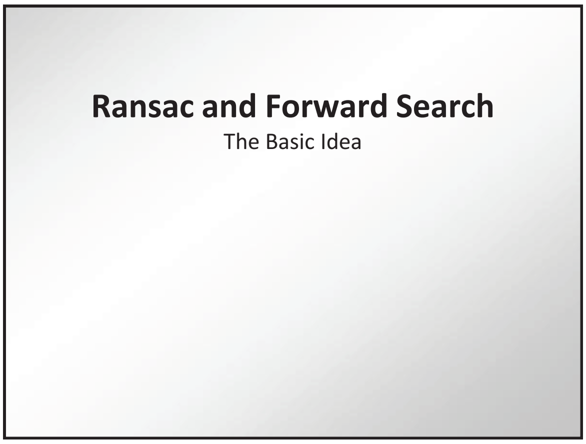


Computing Correspondences in Geometric Datasets

Global, Isometric, Pairwise

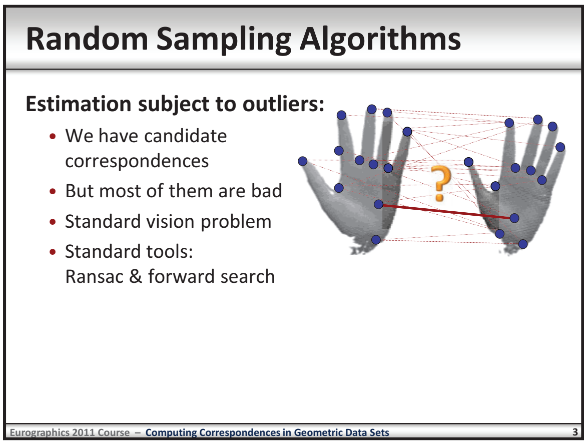
RANSAC · Forward Search · Efficiency

Eurographics 2011
LLANDUDNO UK
11-15 April 2011
Bangor University
School of Computer Science



Ransac and Forward Search

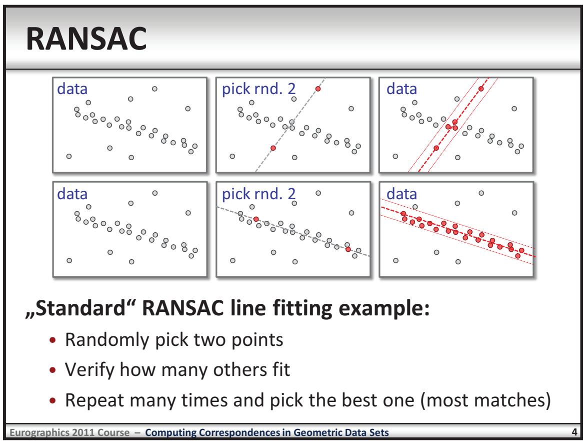
The Basic Idea



Random Sampling Algorithms

Estimation subject to outliers:

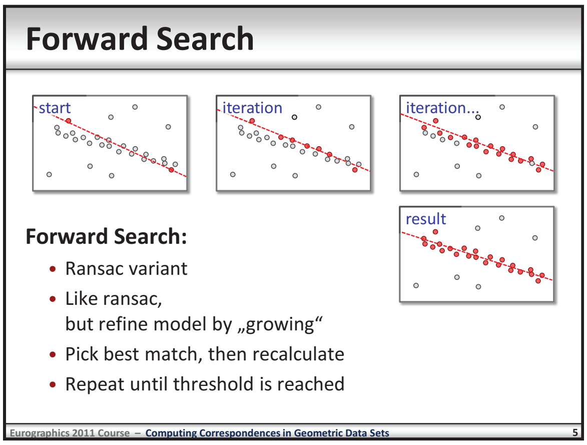
- We have candidate correspondences
- But most of them are bad
- Standard vision problem
- Standard tools: Ransac & forward search



RANSAC

„Standard“ RANSAC line fitting example:

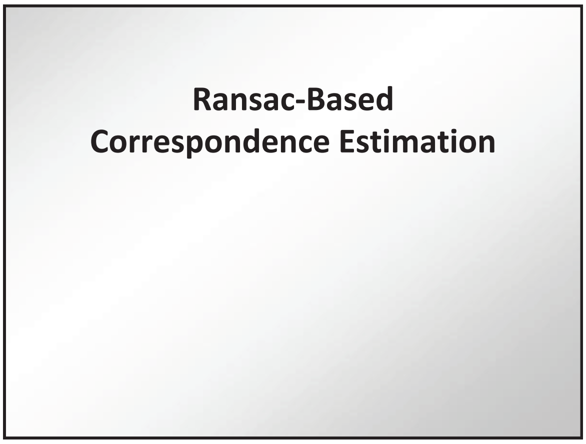
- Randomly pick two points
- Verify how many others fit
- Repeat many times and pick the best one (most matches)



Forward Search

Forward Search:

- Ransac variant
- Like ransac, but refine model by „growing“
- Pick best match, then recalculate
- Repeat until threshold is reached



Ransac-Based Correspondence Estimation

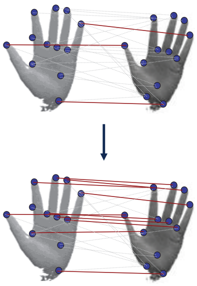
RANSAC/FWS Algorithm

Idea

- Starting correspondence
- Add more that are consistent
  - Preserve intrinsic distances
- Importance sampling algorithm

Advantages

- Efficient (small initial set)
- General (arbitrary criteria)



Ransac/FWS Details

Algorithm: Simple Idea

- Select correspondences with probability proportional to their plausibility
- First correspondence: Descriptors
- Second: Preserve distance (distribution peaks)
- Third: Preserve distance (even fewer choices)

  ...
- Rapidly becomes deterministic
- Repeat multiple times (typ.: 100x)
  - Choose the largest solution (larges #correspondences)

Ransac/FWS Details

Provably Efficient:

- Theoretically efficient (details later)
- Faster in practice (using descriptors)

Flexible:

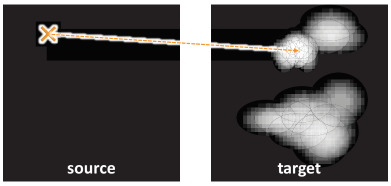
- In later iterations (> 3 correspondences), allow for outlier geodesics
- Can handle topological noise

Foreward Search Algorithm

Forward Search

- Add correspondences incrementally
- Compute match probabilities given the information already decided on
- Iterate until no more matches can found that meet a certain error threshold
- Outer Loop:
  - Iterate the algorithm with random choices
  - Pick the best (i.e., largest) solution

Foreward Search Algorithm

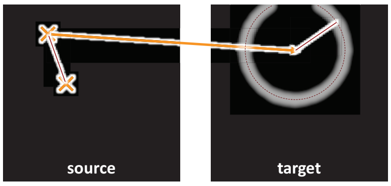


Descriptor matching scores

Step 1:

- Start with one correspondence
  - Target side importance sampling:
    prefer good descriptor matches
  - Optional source side imp. sampl: prefer unique descriptors

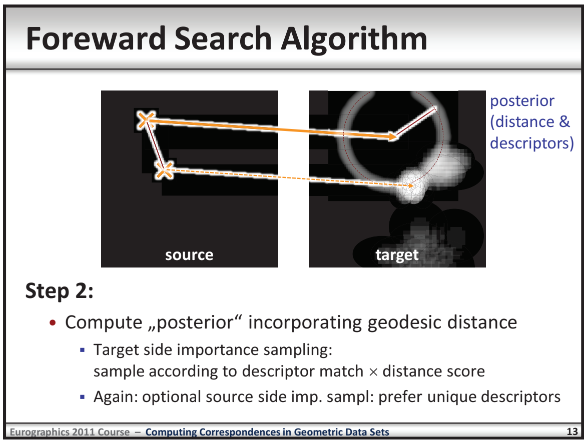
Foreward Search Algorithm



posterior (distance)

Step 2:

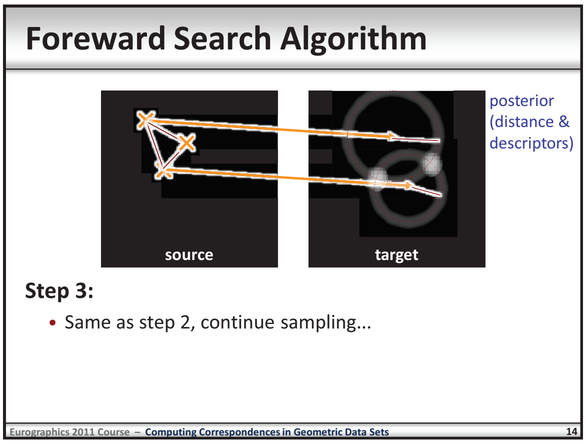
- Compute „posterior“ incorporating geodesic distance
  - Target side importance sampling:
    sample according to descriptor match × distance score
  - Again: optional source side imp. sampl: prefer unique descriptors



Foreward Search Algorithm

Step 2:

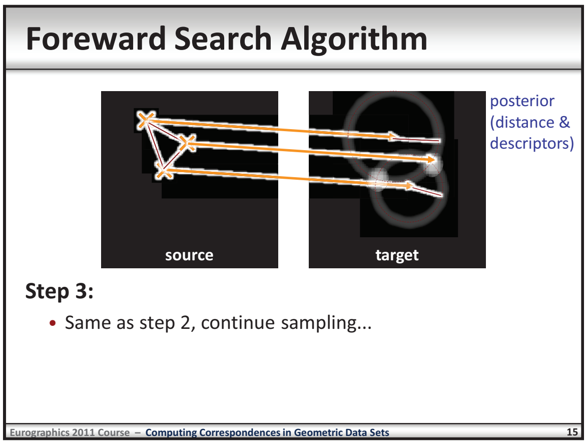
- Compute „posterior“ incorporating geodesic distance
  - Target side importance sampling:
    sample according to descriptor match × distance score
  - Again: optional source side imp. sampl: prefer unique descriptors



Foreward Search Algorithm

Step 3:

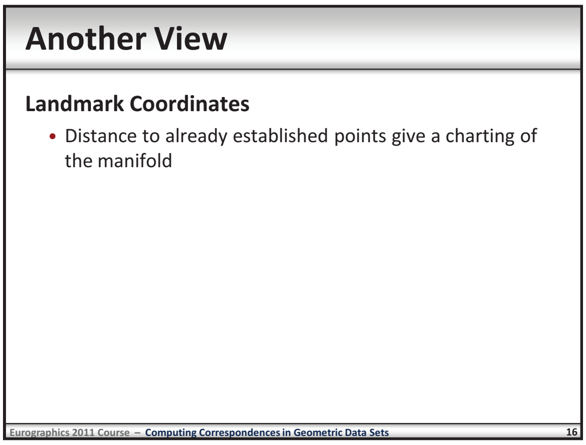
- Same as step 2, continue sampling...



Foreward Search Algorithm

Step 3:

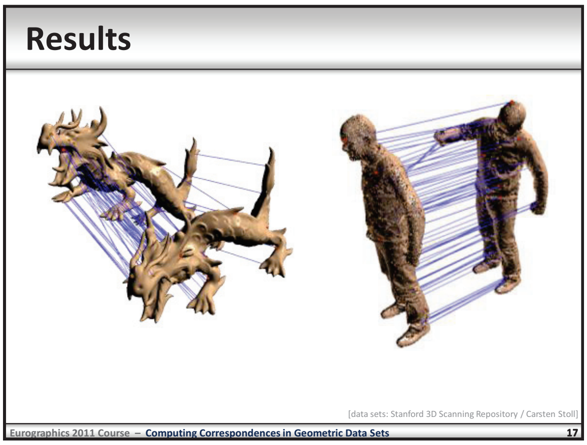
- Same as step 2, continue sampling...



Another View

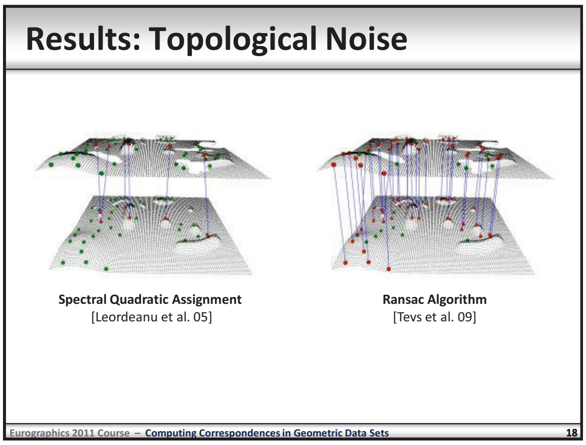
Landmark Coordinates

- Distance to already established points give a charting of the manifold



Results

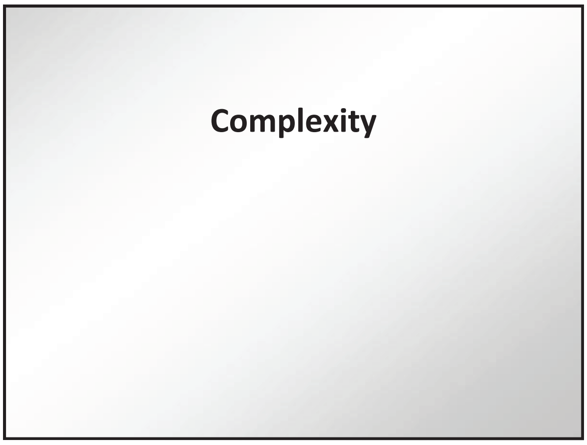
[data sets: Stanford 3D Scanning Repository / Carsten Stoll]



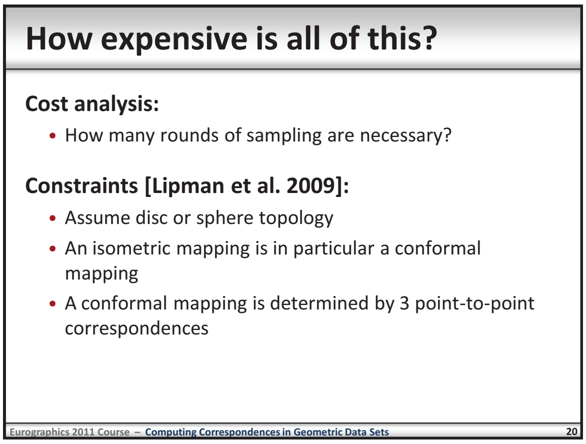
Results: Topological Noise

Spectral Quadratic Assignment
[Leordeanu et al. 05]

Ransac Algorithm
[Tevs et al. 09]



Complexity



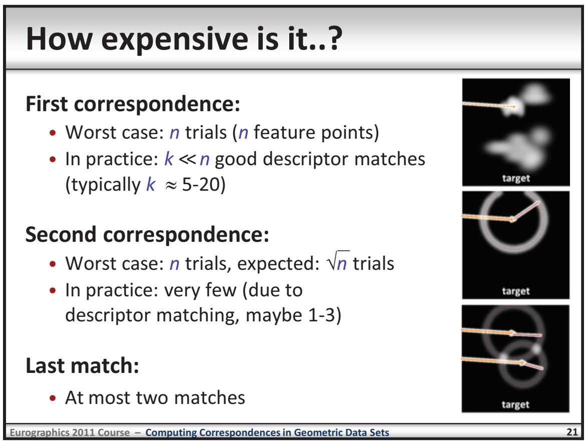
How expensive is all of this?

Cost analysis:

- How many rounds of sampling are necessary?

Constraints [Lipman et al. 2009]:

- Assume disc or sphere topology
- An isometric mapping is in particular a conformal mapping
- A conformal mapping is determined by 3 point-to-point correspondences



How expensive is it..?

target
target
target

First correspondence:

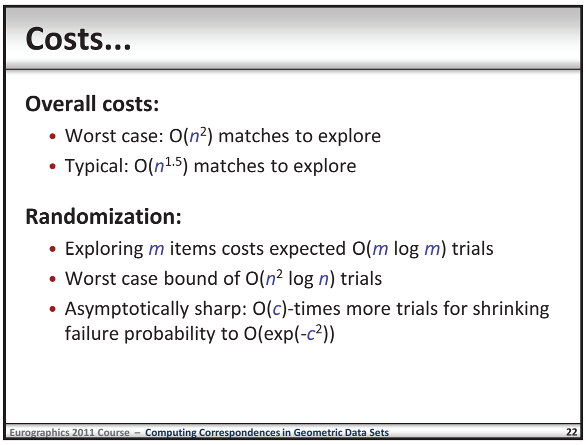
- Worst case: n trials (n feature points)
- In practice: $k \ll n$ good descriptor matches (typically $k \approx$ 5-20)

Second correspondence:

- Worst case: n trials, expected: $\sqrt{n}$ trials
- In practice: very few (due to descriptor matching, maybe 1-3)

Last match:

- At most two matches



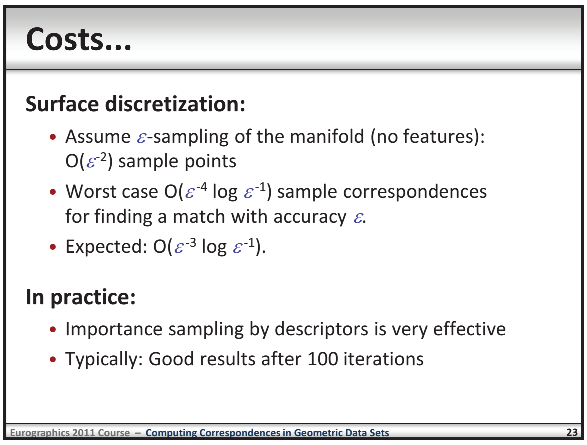
Costs...

Overall costs:

- Worst case: O(n^2) matches to explore
- Typical: O($n^{1.5}$) matches to explore

Randomization:

- Exploring m items costs expected O(m log m) trials
- Worst case bound of O(n^2 log n) trials
- Asymptotically sharp: O(c)-times more trials for shrinking failure probability to O(exp(-c^2))



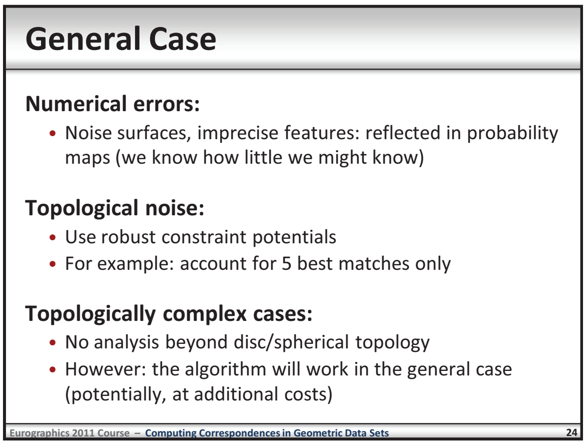
Costs...

Surface discretization:

- Assume ε-sampling of the manifold (no features): O(ε^{-2}) sample points
- Worst case O(ε^{-4} log ε^{-1}) sample correspondences for finding a match with accuracy ε.
- Expected: O(ε^{-3} log ε^{-1}).

In practice:

- Importance sampling by descriptors is very effective
- Typically: Good results after 100 iterations



General Case

Numerical errors:

- Noise surfaces, imprecise features: reflected in probability maps (we know how little we might know)

Topological noise:

- Use robust constraint potentials
- For example: account for 5 best matches only

Topologically complex cases:

- No analysis beyond disc/spherical topology
- However: the algorithm will work in the general case (potentially, at additional costs)

Partial and Approximate Symmetry Detection for 3D Geometry

Partial Symmetry Detection

Given

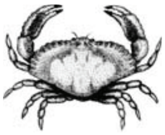
Shape model (represented as point cloud, mesh, ...)

Goal

Identify and extract ***similar*** (symmetric) patches of different ***size*** across different ***resolutions***

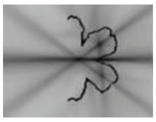
Symmetry in Nature

"Symmetry is a complexity-reducing concept [...]; seek it everywhere."
- Alan J. Perlis



"Females of several species, including [...] humans, prefer symmetrical males."
- Chris Evan

Related Work



[Podolak et al. `06]

[Loy and Eklundh `06]

Hough transform on feature points

[Gal and Cohen-Or `05]

tradeoff memory for speed

Symmetry for Geometry Processing

[Katz and Tal `04]

[Funkhouser et al. `05]

[Khazdan et al. `04]

[Sharf et al. `04]

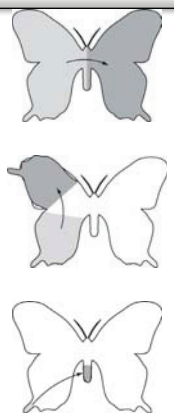
Types of Symmetry

Transform Types:

Reflection

Rotation +

Uniform Scaling



Contributions

Automatic detection of discrete symmetries !
reflection, rigid transform, uniform scaling

Symmetry graphs !
high level structural information about object

Output sensitive algorithms !
low memory requirements

Problem Characteristics

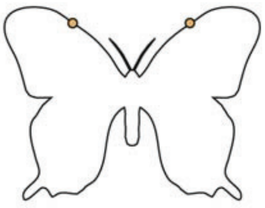
Difficulties

- Which parts are symmetric !
  objects not pre-segmented
- Space of transforms: rotation + translation
- Brute force search is not feasible

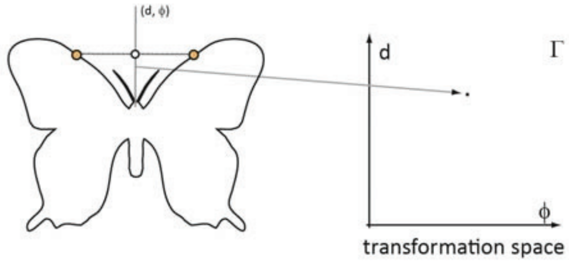
Easy

- Proposed symmetries ! easy to validate

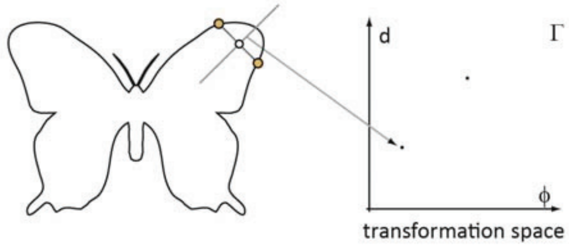
Reflective Symmetry



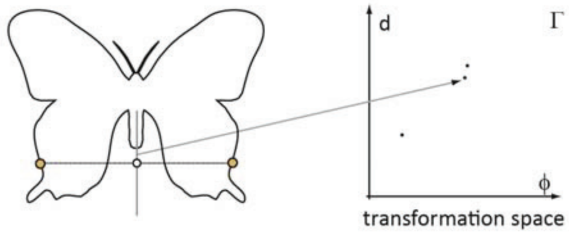
Reflective Symmetry: A Pair Votes



Reflective Symmetry: Voting Continues



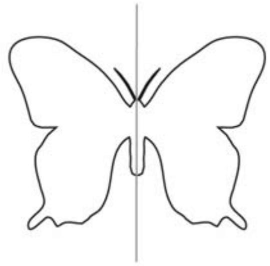
Reflective Symmetry: Voting Continues

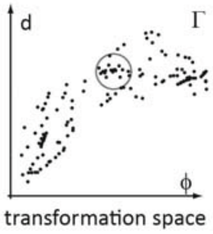


Reflective Symmetry: Largest Cluster

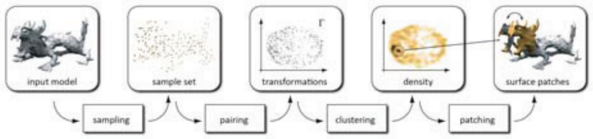
He

Sp

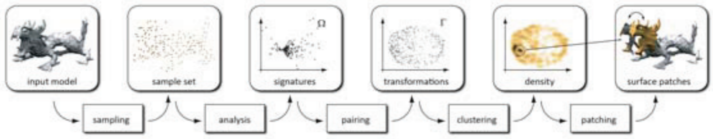




Pipeline



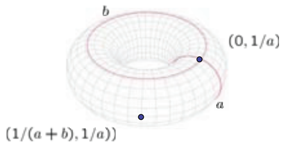
Pipeline



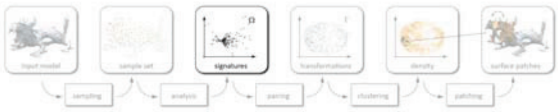
Pruning: Local Signatures

Local signature ! invariant under transforms

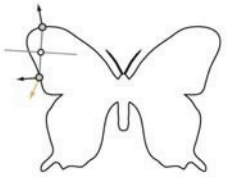
Signatures disagree ! points don't correspond

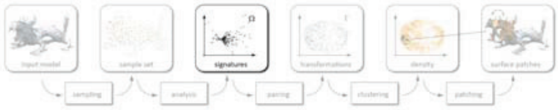


Use (κ_1, κ_2) for curvature based pruning

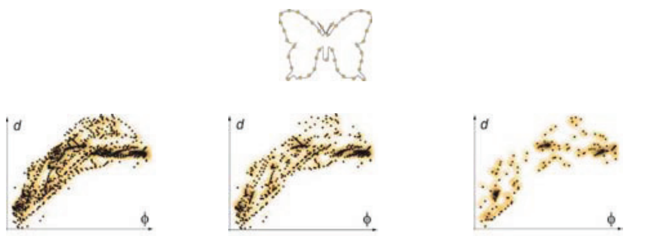


Reflection: Normal-based Pruning





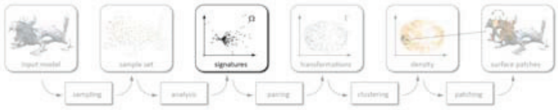
Point Pair Pruning



all pairs

curvature based

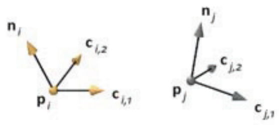
curvature + normal based



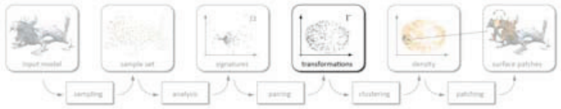
Transformations

Reflection ! point-pairs

Rigid transform ! more information



Robust estimation of
principal curvature frames
[Cohen-Steiner et al. `03]

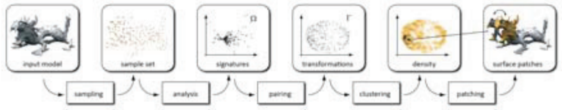


Random Sampling

Height of clusters related to symmetric region size

Random samples !
larger regions likely to be detected earlier

Output sensitive

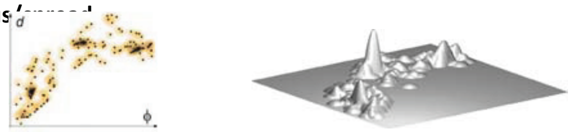


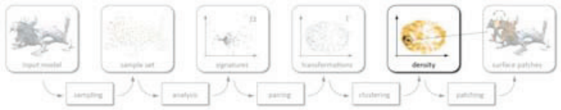
Mean-Shift Clustering

Kernel:

Radially symmetric

Radius/spread





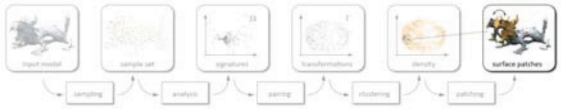
Model Reduction: Chambord

Verification

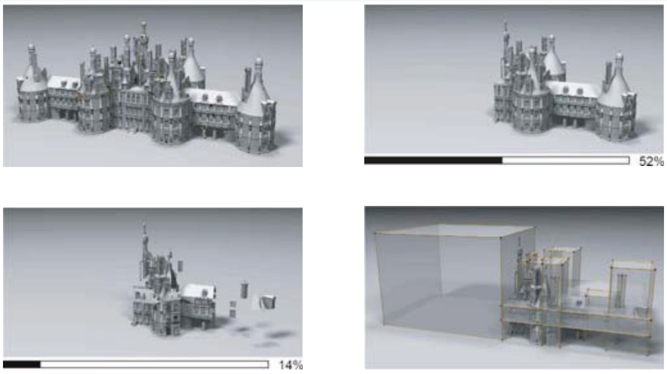
Clustering gives a good guess

Verify ! build symmetric patches

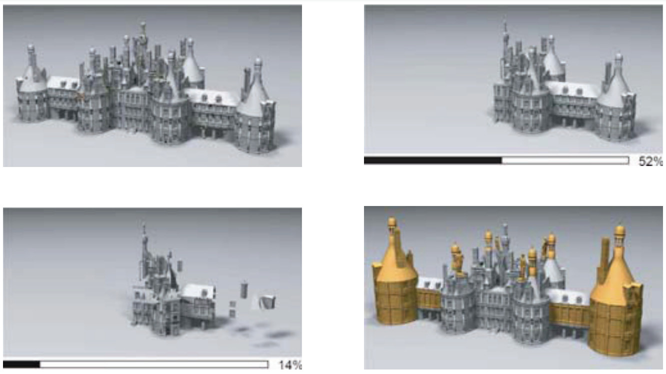
Locally refine solution using ICP algorithm
[Besl and McKay `92]



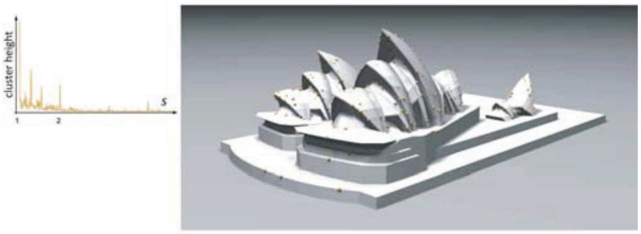
Model Reduction: Chambord



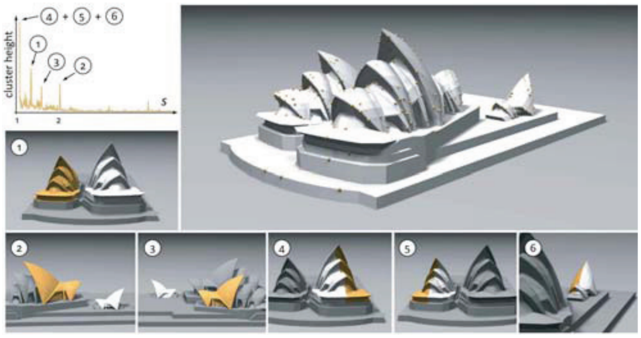
Model Reduction: Chambord



Sydney Opera House



Sydney Opera House



Approximate Symmetry: Dragon

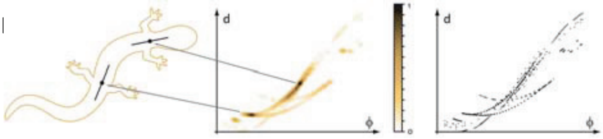
detected symmetries

correction field

UNITS: fraction of bounding box diagonal

Limitations

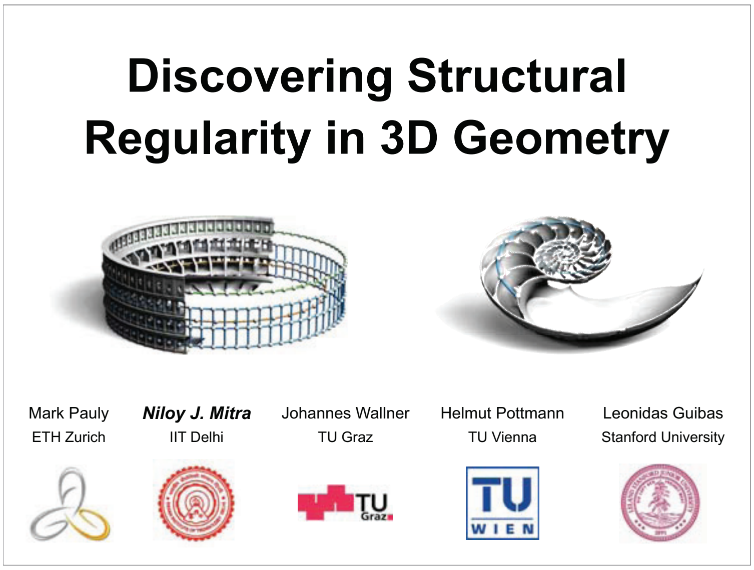
Cannot differentiate between small sized symmetries and com



[Castro et al. `06]

Articulated Motion: Horses

‘symmetry’ detection between two objects ! registration



Discovering Structural Regularity in 3D Geometry

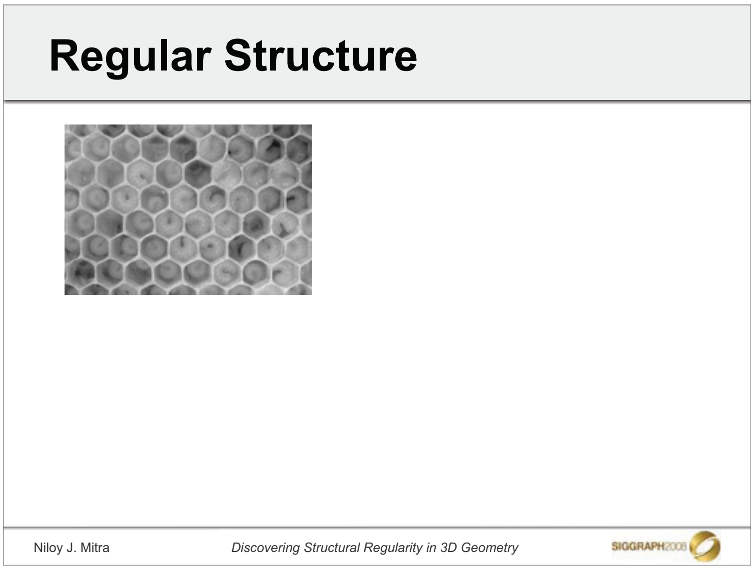
Mark Pauly
ETH Zurich

Niloy J. Mitra
IIT Delhi

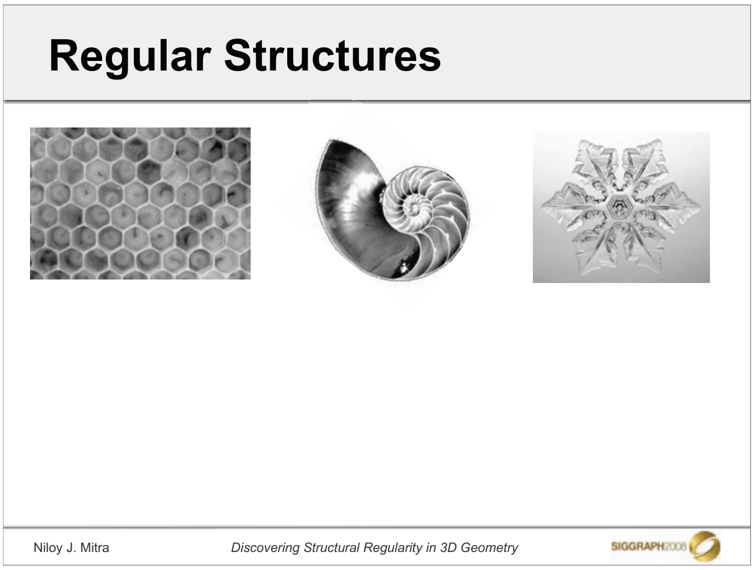
Johannes Wallner
TU Graz

Helmut Pottmann
TU Vienna

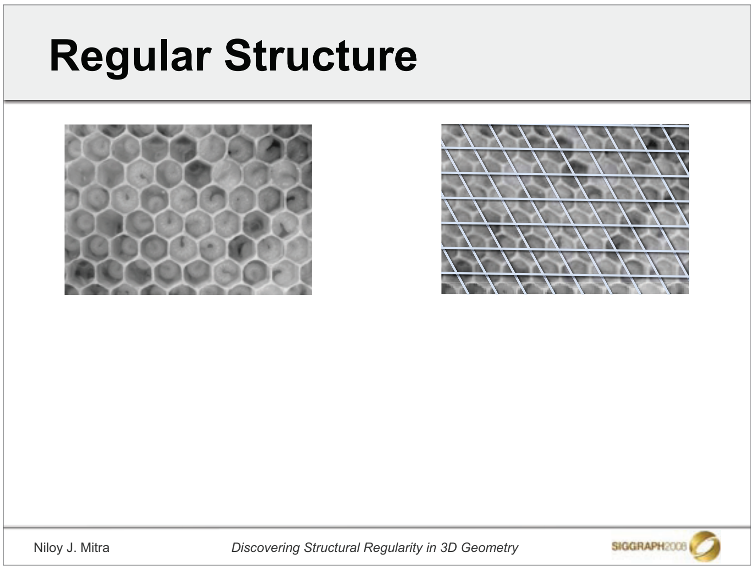
Leonidas Guibas
Stanford University



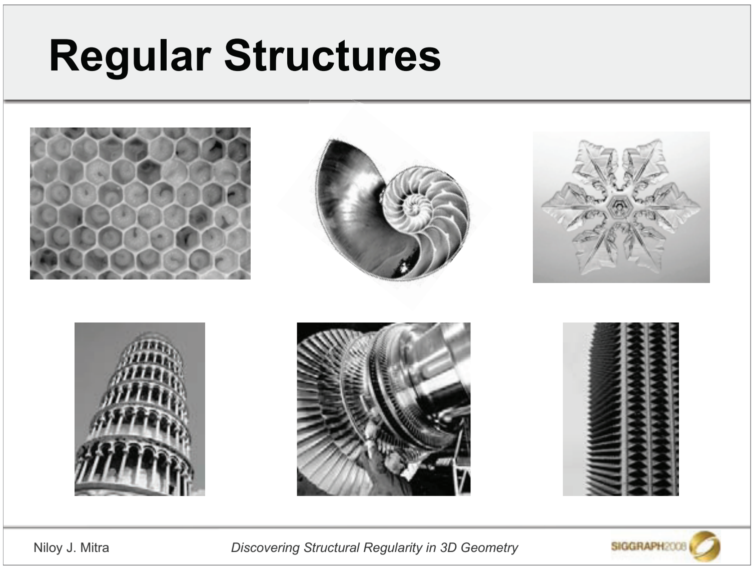
Regular Structure



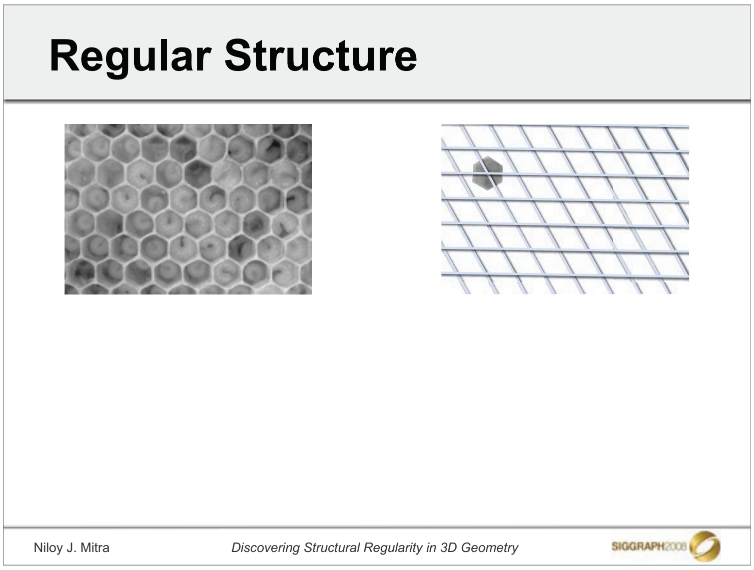
Regular Structures



Regular Structure

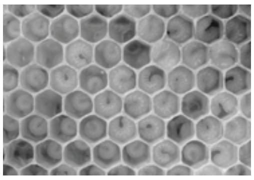


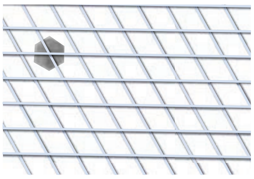
Regular Structures

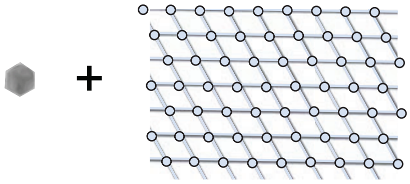


Regular Structure

Regular Structure







Motivation

- Regularity → form, semantics

Motivation

Motivation

- Regularity → form, semantics

- Scan cleaning, completion

Motivation

- Regularity → form, semantics

Motivation

- Regularity → form, semantics

- Scan cleaning, completion
- Compression

Motivation

- Regularity ⟶ form, semantics

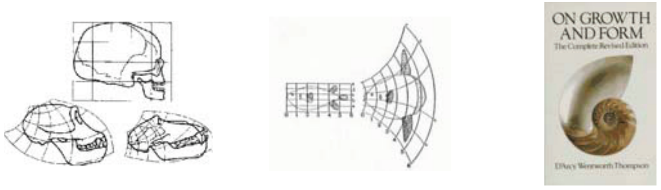
- Scan cleaning, completion
- Compression
- Geometric edits, synthesis

Motivation

- Regularity ⟶ form, semantics

- Scan cleaning, completion
- Compression
- Geometric edits, synthesis
- Growth laws or design principles

Inspiration

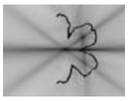


On Growth and Form
[Thompson 1917]

Related Work

Related Work

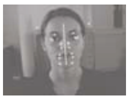
[Podolak et al. `06]

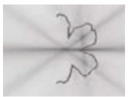

[Loy, Eklundh `06]

[Mitra et al. `06]

[Martinet et al. `07]

Related Work


[Podolak et al. `06]


[Loy, Eklundh `06]

[Mitra et al. `06]

[Martinet et al. `07]

[Funkhouser et al. `05]

[Thrun, Wegbreit `05]

[Shimari et al. `06]

[Liu et al. `08]

Related Work

[Podolak et al. `06]

[Loy, Eklundh `06]

[Mitra et al. `06]

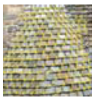
[Martinet et al. `07]

[Funkhouser et al. `05]

[Thrun, Wegbreit `05]

[Shimari et al. `06]

[Liu et al. `08]

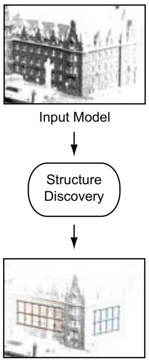

[Hays et al. `06]

[Mueller et al. `07]

[Baudes et al. `08]

[Wang et al. `08]

Structure Discovery



Related Work

[Podolak et al. `06]

[Loy, Eklundh `06]

[Mitra et al. `06]

[Martinet et al. `07]

[Funkhouser et al. `05]

[Thrun, Wegbreit `05]

[Shimari et al. `06]

[Liu et al. `08]

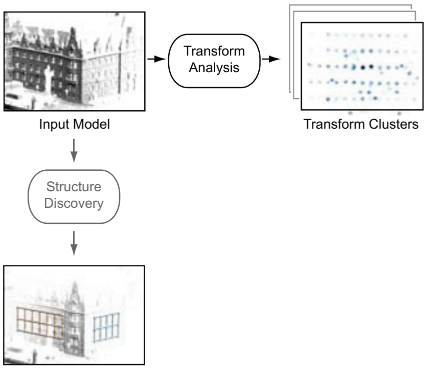
[Hays et al. `06]

[Mueller et al. `07]

[Baudes et al. `08]

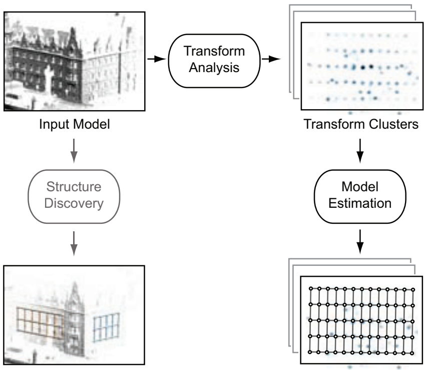
[Wang et al. `08]

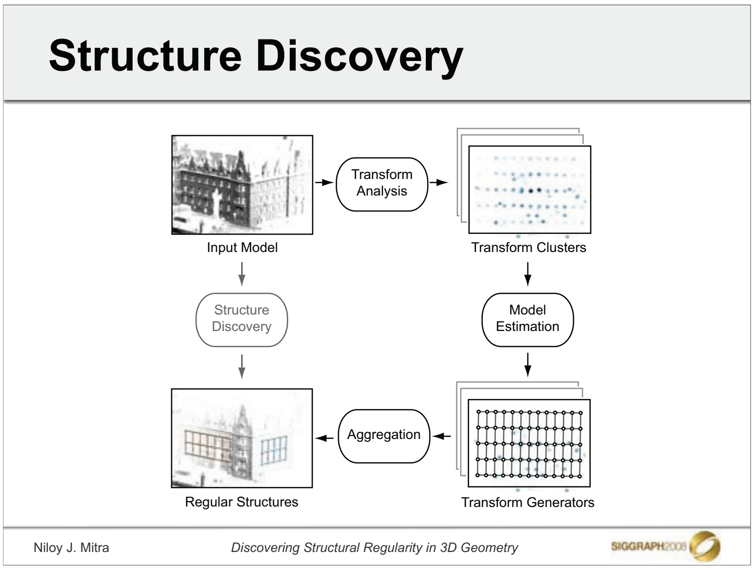
Structure Discovery



Structure Discovery

Structure Discovery





Structure Discovery

Input Model

Transform Analysis

Transform Clusters

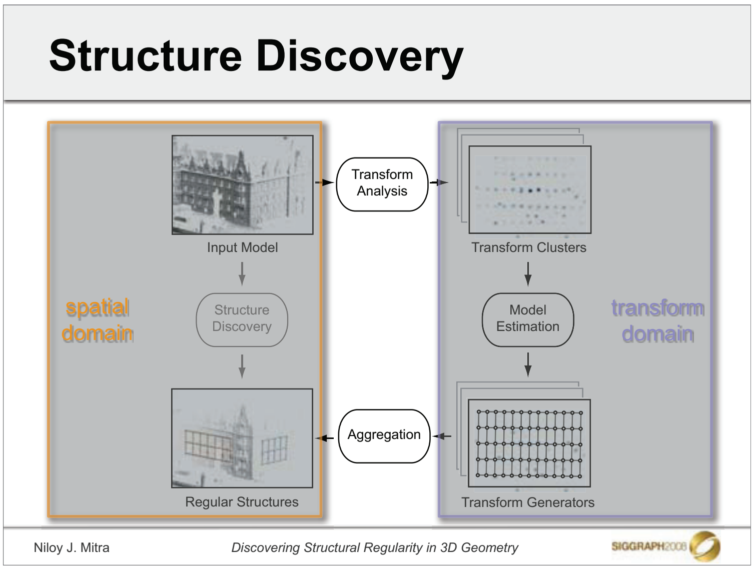
Structure Discovery

Model Estimation

Aggregation

Regular Structures

Transform Generators



Structure Discovery

spatial domain

Input Model

Transform Analysis

Transform Clusters

transform domain

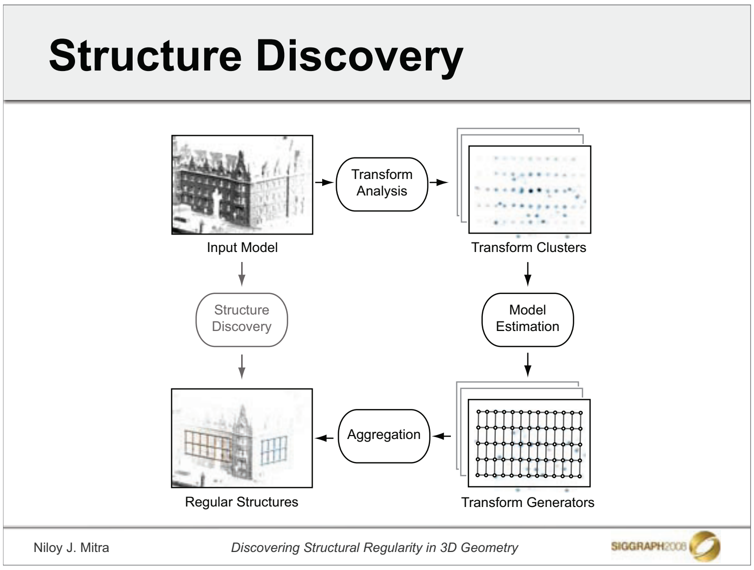
Structure Discovery

Model Estimation

Aggregation

Regular Structures

Transform Generators



Structure Discovery

Input Model

Transform Analysis

Transform Clusters

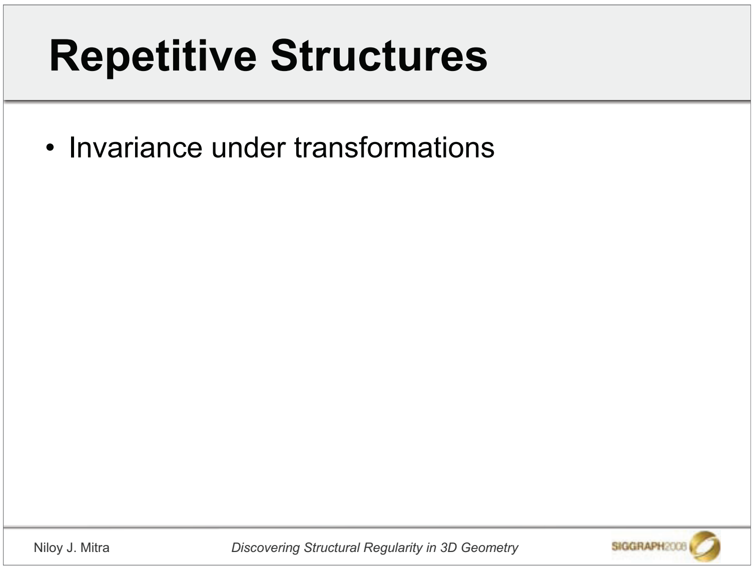
Structure Discovery

Model Estimation

Aggregation

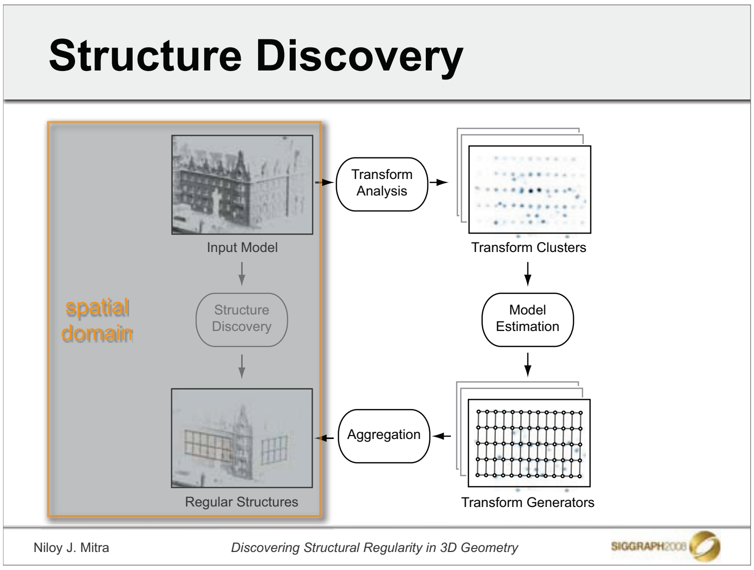
Regular Structures

Transform Generators



Repetitive Structures

- Invariance under transformations



Structure Discovery

spatial domain

Input Model

Transform Analysis

Transform Clusters

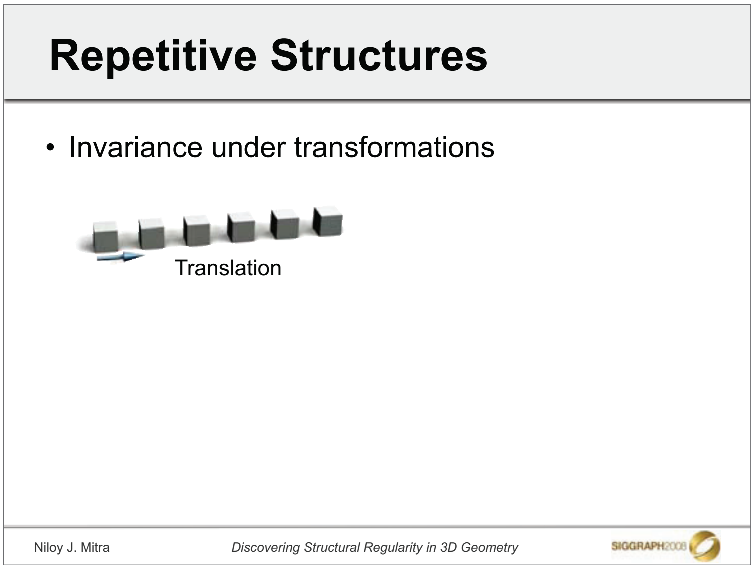
Structure Discovery

Model Estimation

Aggregation

Regular Structures

Transform Generators



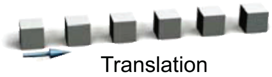
Repetitive Structures

- Invariance under transformations

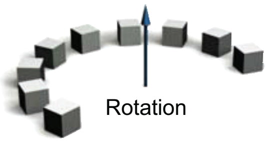
Translation

Repetitive Structures

- Invariance under transformations



Translation



Rotation

Repetitive Structures

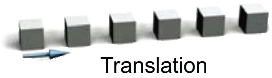
- Invariance under transformations

Translation + Rotation

Rotation + Scaling

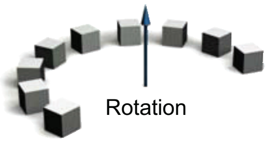
Repetitive Structures

- Invariance under transformations



Translation

Scaling



Rotation

Repetitive Structures

- Invariance under transformations

Translation + Rotation

Rotation + Scaling

1-parameter patterns

Repetitive Structures

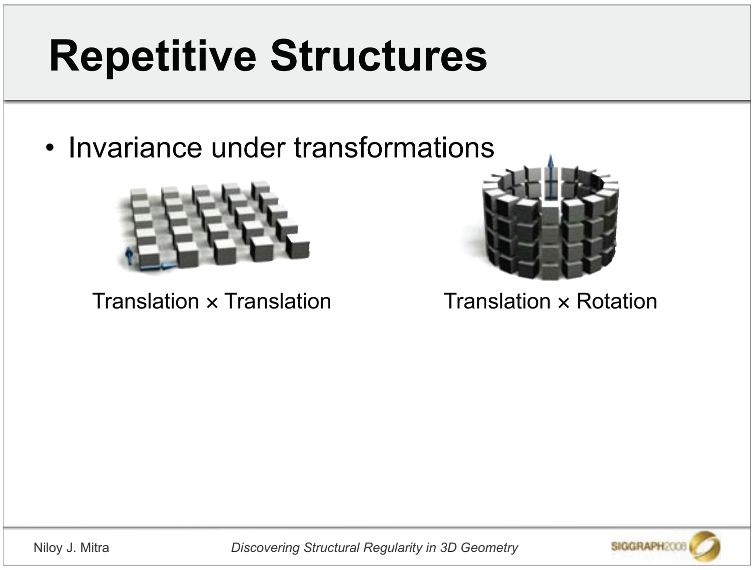
- Invariance under transformations

Translation + Rotation

Repetitive Structures

- Invariance under transformations

Translation × Translation

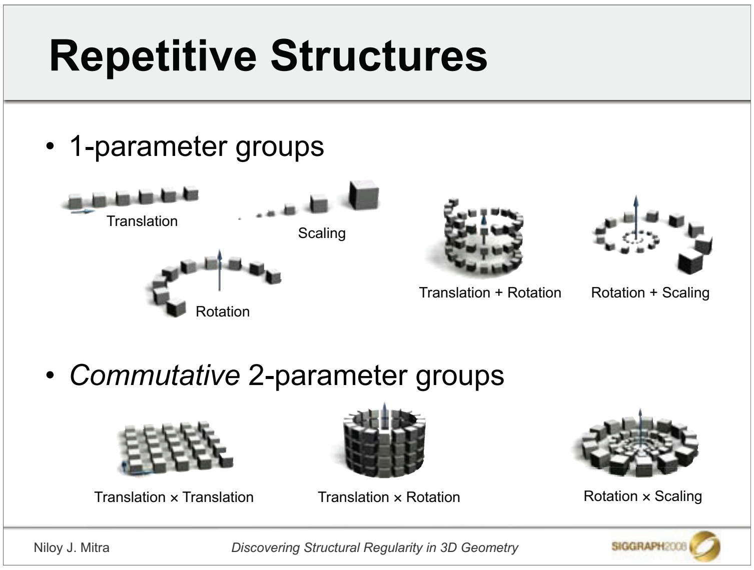


Repetitive Structures

- Invariance under transformations

Translation × Translation

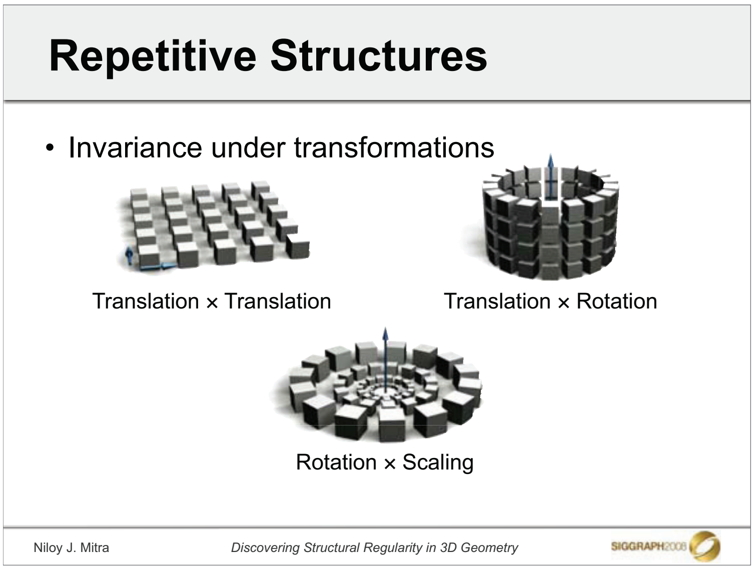
Translation × Rotation



Repetitive Structures

- 1-parameter groups

- *Commutative* 2-parameter groups



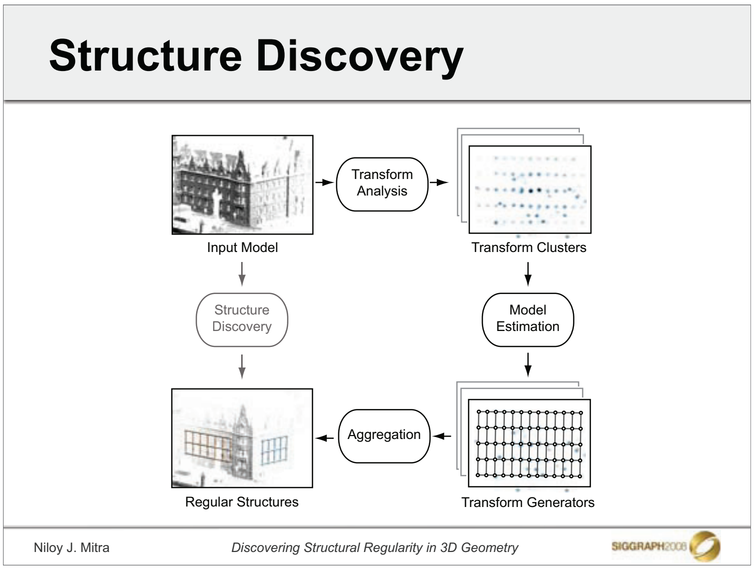
Repetitive Structures

- Invariance under transformations

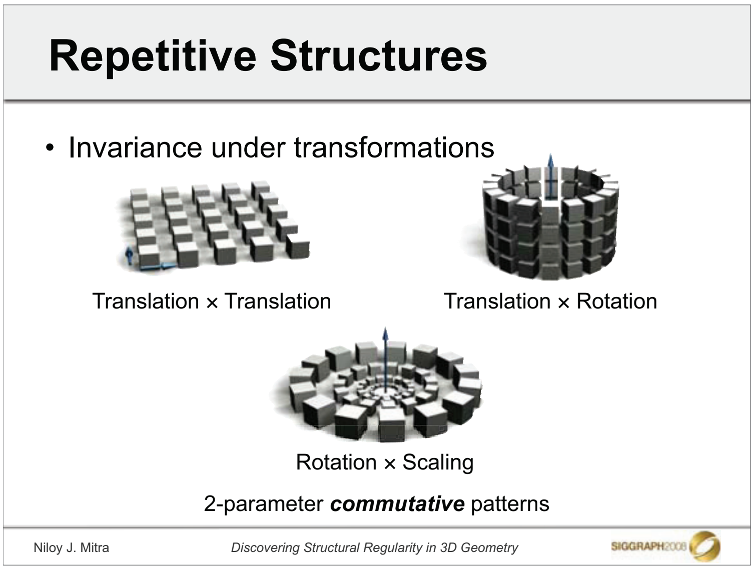
Translation × Translation

Translation × Rotation

Rotation × Scaling



Structure Discovery



Repetitive Structures

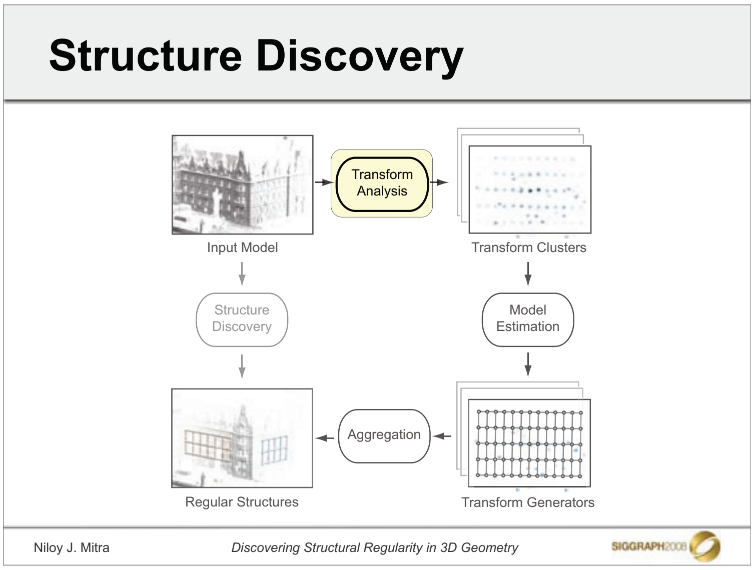
- Invariance under transformations

Translation × Translation

Translation × Rotation

Rotation × Scaling

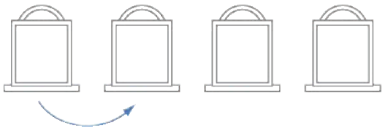
2-parameter ***commutative*** patterns



Structure Discovery

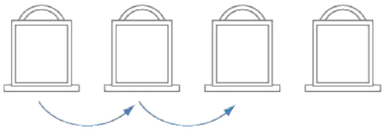
Transformations

Transformations



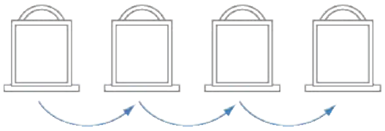
Transformations [Mitra et al. `06]

Transformations

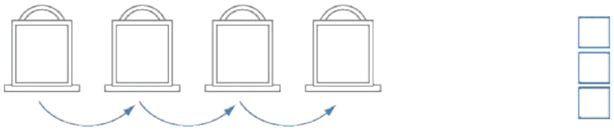


Transformations

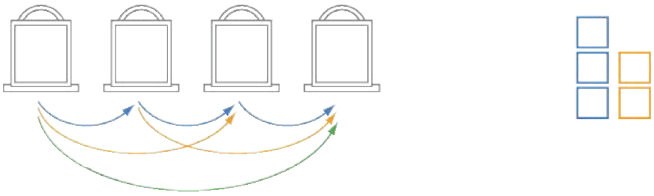
Transformations



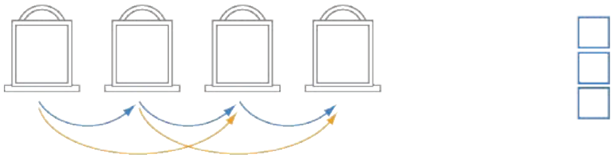
Transformations



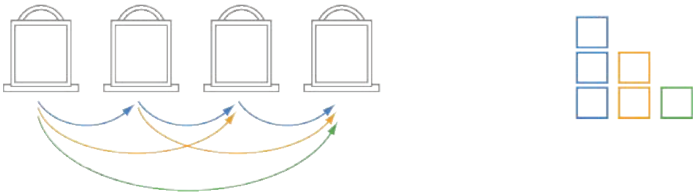
Transformations



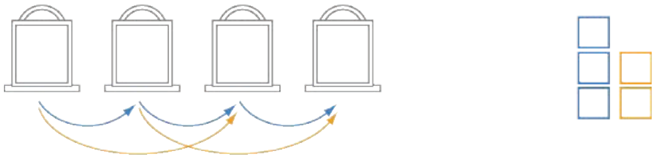
Transformations



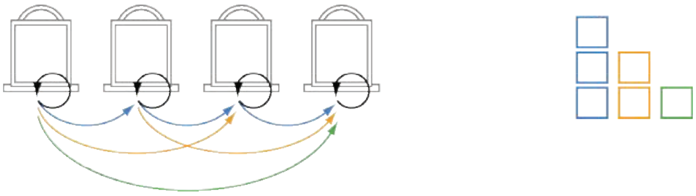
Transformations

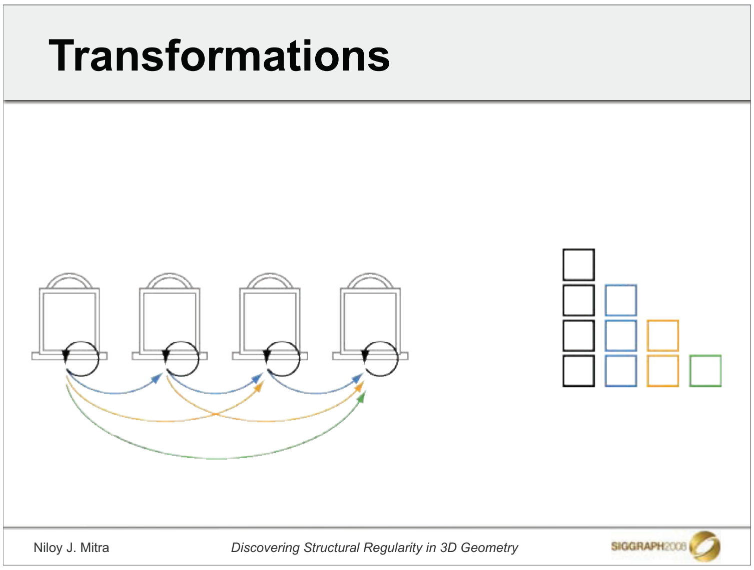


Transformations

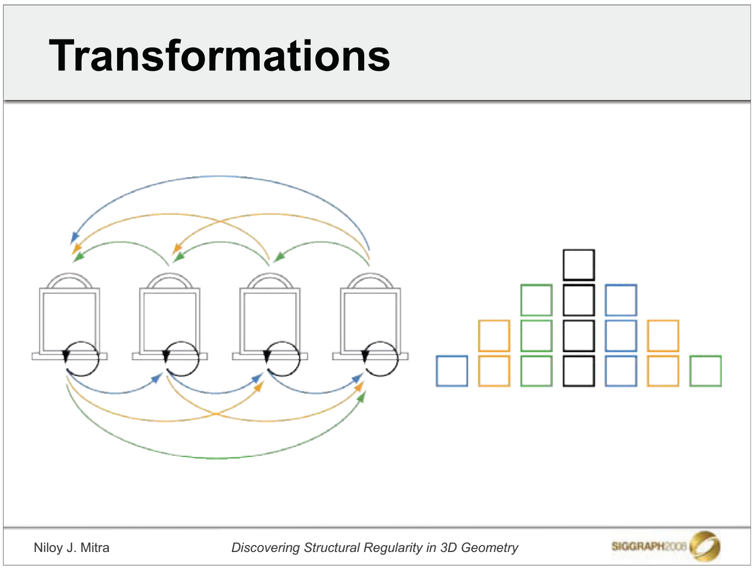


Transformations

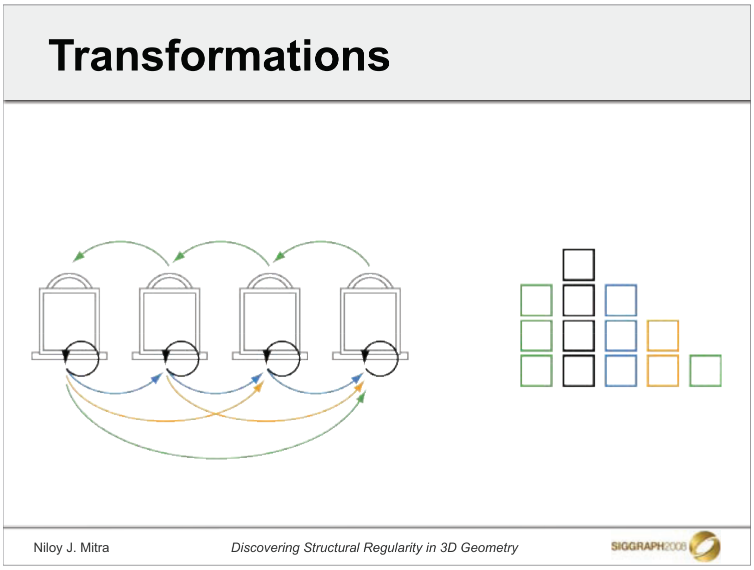




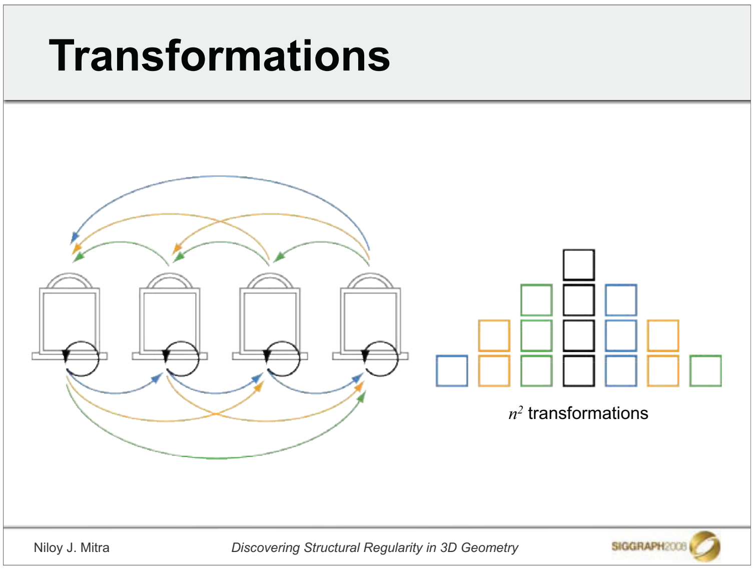
Transformations



Transformations

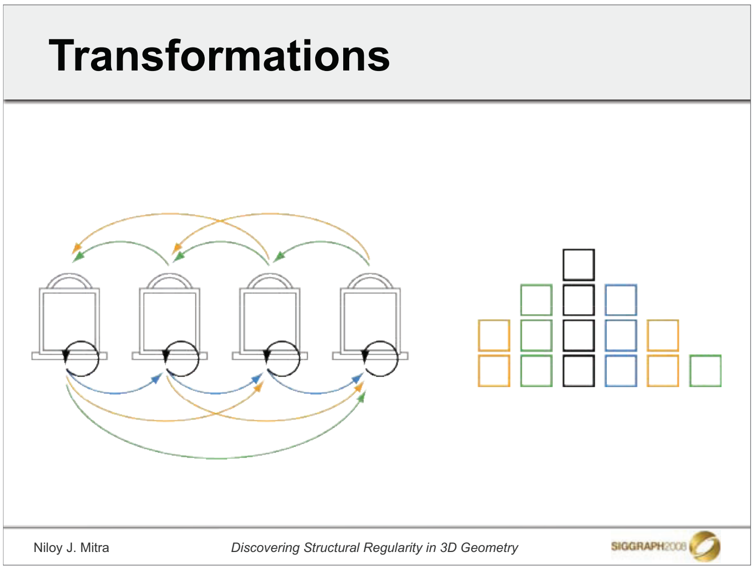


Transformations

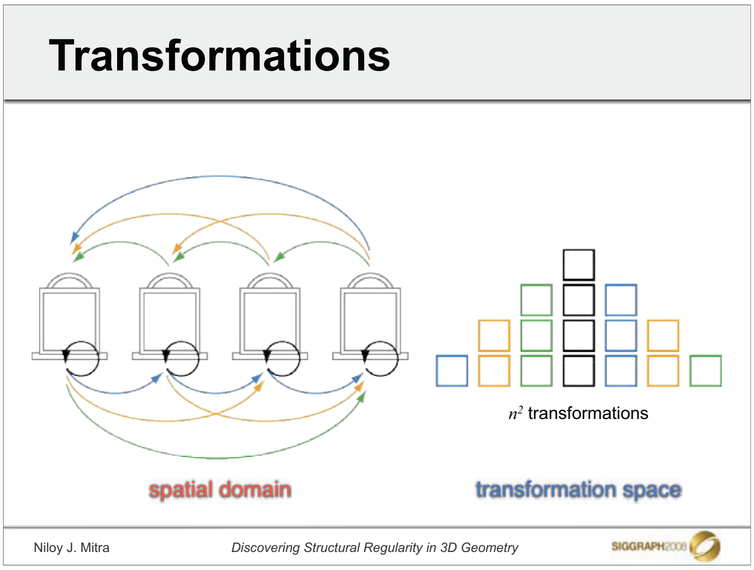


Transformations

n^2 transformations



Transformations



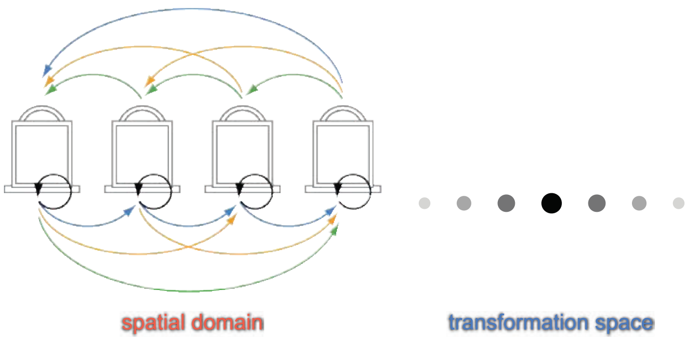
Transformations

n^2 transformations

spatial domain

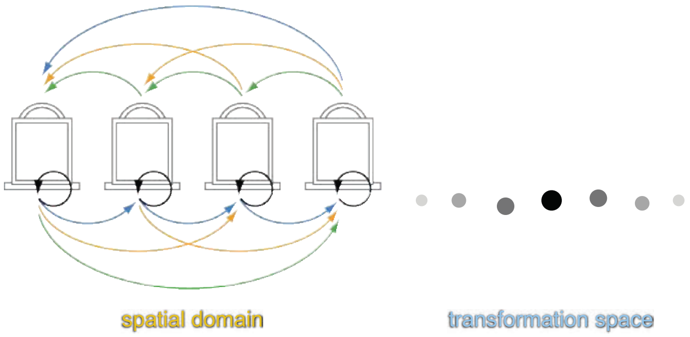
transformation space

Transformations

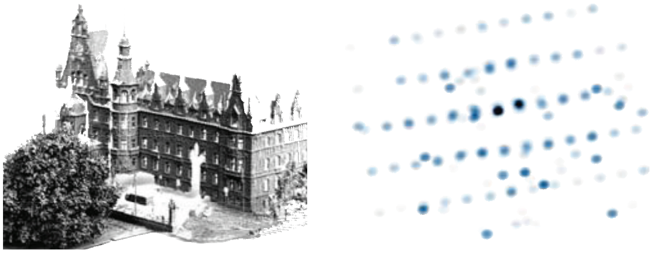


Model Estimation

Transformations

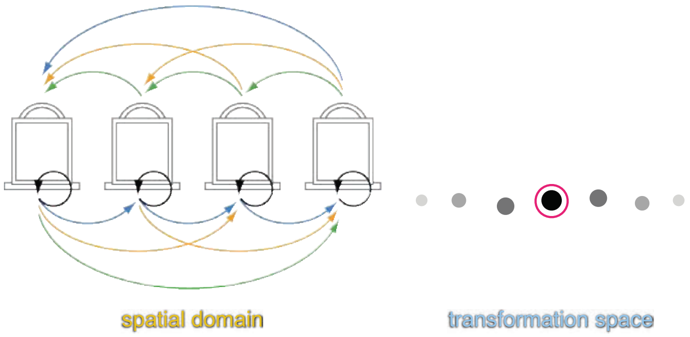


Model Estimation

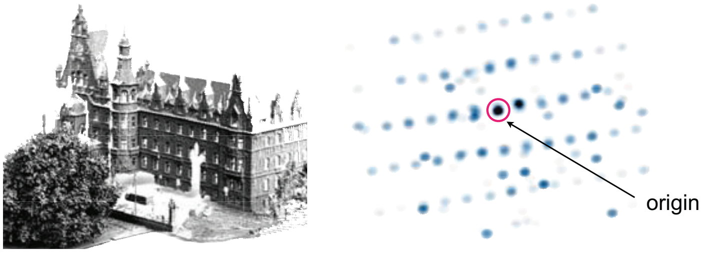


density plot of
pair-wise transformations

Transformations

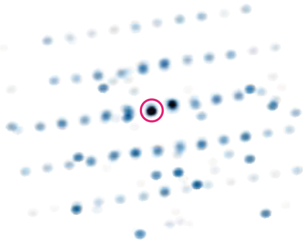


Model Estimation



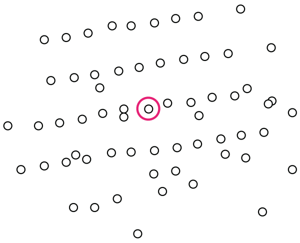
density plot of
pair-wise transformations

Model Estimation



density plot of
pair-wise transformations

Model Estimation



cluster centers

Transform Mapping

Transform Mapping

$$\mathbf{G}_1^i.\mathbf{G}_1^j \rightarrow \{i\mathbf{g}_1 + j\mathbf{g}_2\}$$

Transform Mapping

$$\mathbf{G}_1^i.\mathbf{G}_1^j \rightarrow \{i\mathbf{g}_1 + j\mathbf{g}_2\}$$

$$\mathrm{I} \rightarrow \{\mathbf{0}\}$$

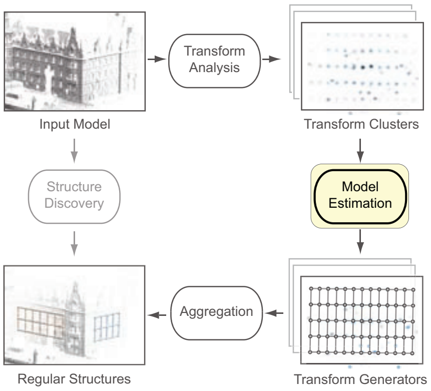
Transform Mapping

$$\mathbf{G}_1^i.\mathbf{G}_1^j \rightarrow \{i\mathbf{g}_1 + j\mathbf{g}_2\}$$

$$\mathrm{I} \rightarrow \{\mathbf{0}\}$$

Translation x Translation	$T \rightarrow (t_1, t_2)$
Rotation x Scaling	$T \rightarrow (\theta, \log s)$
Translation x Rotation	$T \rightarrow (t, \theta)$

Structure Discovery


Input Model

Transform Analysis

Transform Clusters

Structure Discovery

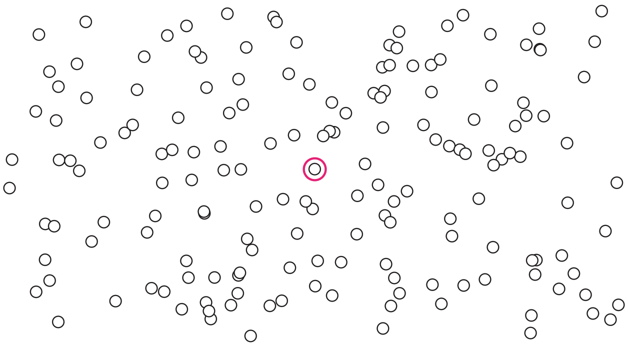
Model Estimation

Regular Structures

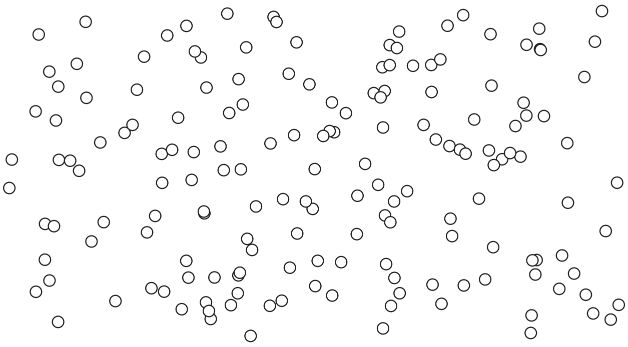
Aggregation

Transform Generators

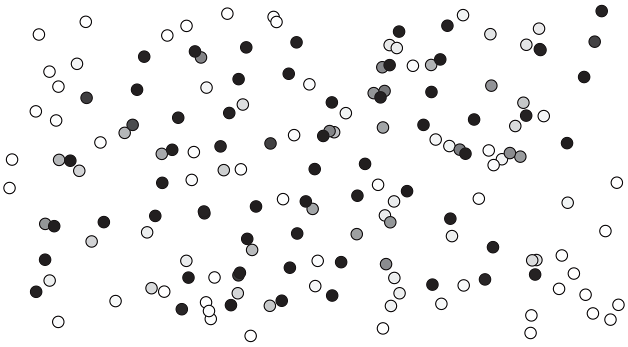
Is there a Pattern?



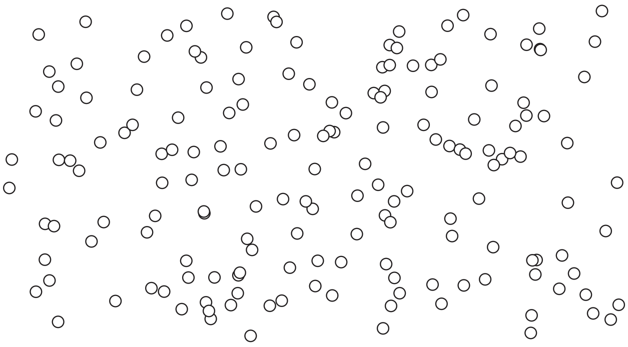
Model Estimation



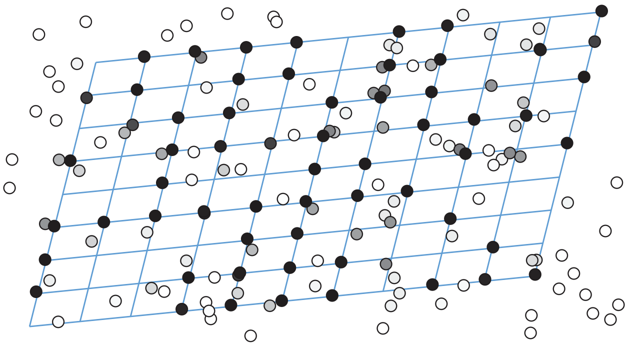
Yes, there is!



Is there a Pattern?



Yes, there is!



Model Estimation

Model Estimation

- Grid fitting
  - ***input:*** cluster centers

$$C = \{\mathbf{c}_1, \ldots, \mathbf{c}_n\}$$

Model Estimation

- Global, non-linear optimization

Model Estimation

- Grid fitting
  - ***input:*** cluster centers

$$C = \{\mathbf{c}_1, \ldots, \mathbf{c}_n\}$$

  - ***unknowns:*** grid generators

$$\mathbf{x}_{ij} = i\mathbf{g}_1 + j\mathbf{g}_2$$

grid location

generating vectors

Model Estimation

- Global, non-linear optimization
  - ***simultaneously*** detects ***outliers*** and ***grid structure***

Model Estimation

- Grid fitting
  - ***input:*** cluster centers

$$C = \{\mathbf{c}_1, \ldots, \mathbf{c}_n\}$$

  - ***unknowns:*** grid generators

$$\mathbf{x}_{ij} = i\mathbf{g}_1 + j\mathbf{g}_2$$

$$i \in [-n, n]$$

$$j \in [-m, m]$$

grid location

generating vectors

Model Estimation

- Fitting terms

$$E_{C\to X} = \sum_{k=1}^{|C|} \|\underset{\text{cluster center}}{\mathbf{c}_k} - \underset{\text{closest grid point}}{\mathbf{x}(k)}\|^2$$

$$E_{X\to C} = \sum_i \sum_j \|\underset{\text{grid point}}{\mathbf{x}_{ij}} - \underset{\text{closest cluster center}}{\mathbf{c}(i,j)}\|^2$$

Model Estimation

- Global, non-linear optimization
  - ***simultaneously*** detects ***outliers*** and ***grid structure***

Model Estimation

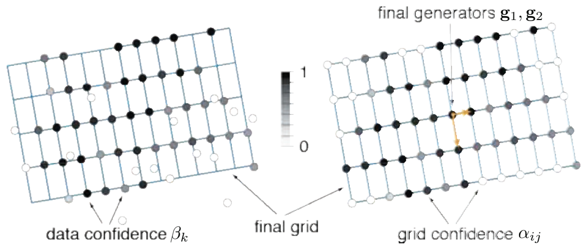
- Fitting terms

$$E_{C\to X} = \sum_{k=1}^{|C|} \overset{\text{data confidence}}{\boxed{\beta_k^2}} \|\underset{\text{cluster center}}{\mathbf{c}_k} - \underset{\text{closest grid point}}{\mathbf{x}(k)}\|^2$$

$$E_{X\to C} = \sum_i \sum_j \overset{\text{grid confidence}}{\boxed{\alpha_{ij}^2}} \|\underset{\text{grid point}}{\mathbf{x}_{ij}} - \underset{\text{closest cluster center}}{\mathbf{c}(i,j)}\|^2$$

Model Estimation

- Global, non-linear optimization
  - ***simultaneously*** detects ***outliers*** and ***grid structure***


final generators $\mathbf{g}_1, \mathbf{g}_2$
1
0
data confidence β_k
final grid
grid confidence α_{ij}

Model Estimation

- Fitting terms

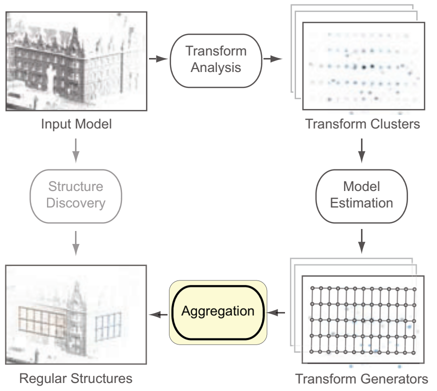
$$E_{C\to X} = \sum_{k=1}^{|C|} \beta_k^2 \|\mathbf{c}_k - \mathbf{x}(k)\|^2$$

$$E_{X\to C} = \sum_i \sum_j \alpha_{ij}^2 \|\mathbf{x}_{ij} - \mathbf{c}(i,j)\|^2$$

- Data and grid confidence terms

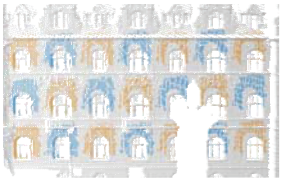
$$E_\alpha = \sum_i \sum_j (1 - \alpha_{ij}^2)^2 \qquad E_\beta = \sum_k (1 - \beta_k^2)^2$$

Structure Discovery



Aggregation

- Region-growing to extract repetitive elements

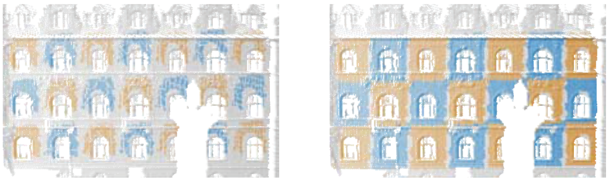


Aggregation

- Region-growing to extract repetitive elements
- Simultaneous registration

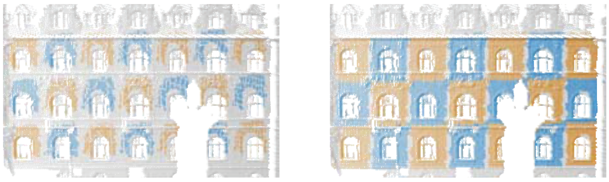
$$\mathbf{H}_+ \approx \mathbf{H} + \epsilon\mathbf{D}\cdot\mathbf{H}$$

$$T_+^k \approx (\mathbf{H} + \epsilon\mathbf{D}\cdot\mathbf{H})^k$$

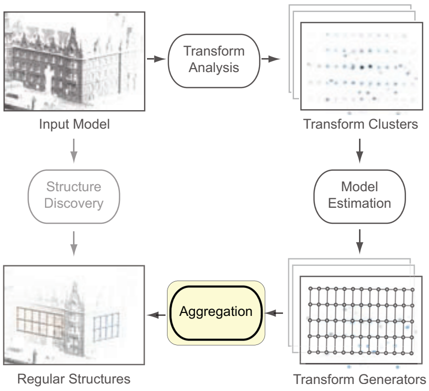


Aggregation

- Region-growing to extract repetitive elements
- Simultaneous registration



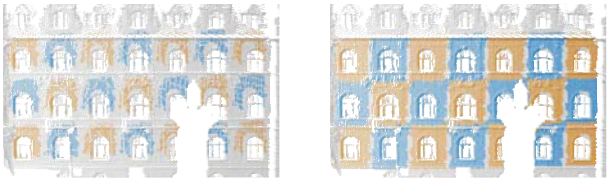
Structure Discovery



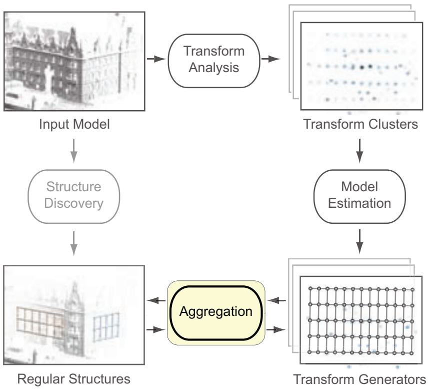
Aggregation

- Region-growing to extract repetitive elements
- Simultaneous registration

$$\mathbf{H}_+ \approx \mathbf{H} + \epsilon\mathbf{D}\cdot\mathbf{H}$$

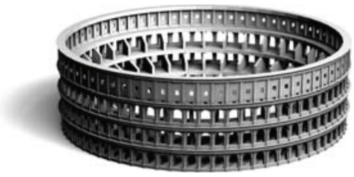


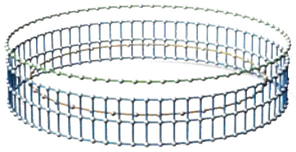
Structure Discovery



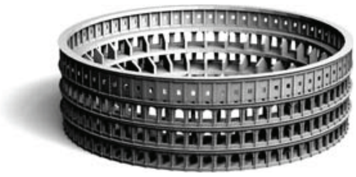
Results and Applications

Amphitheater

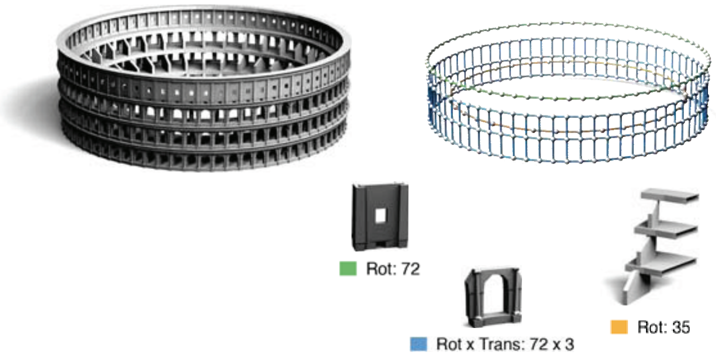




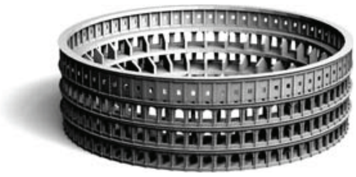
Amphitheater

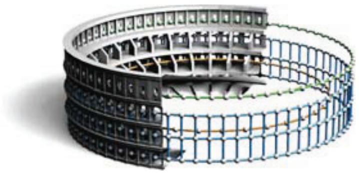


Amphitheater

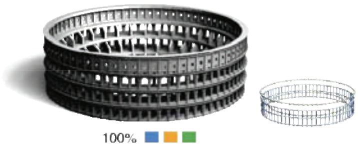


Amphitheater

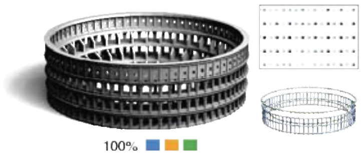




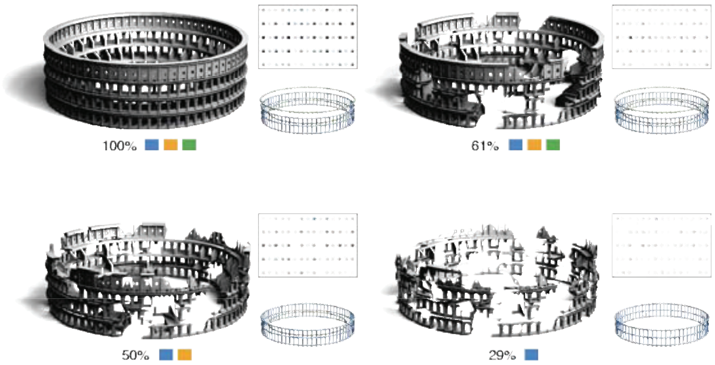
Robustness



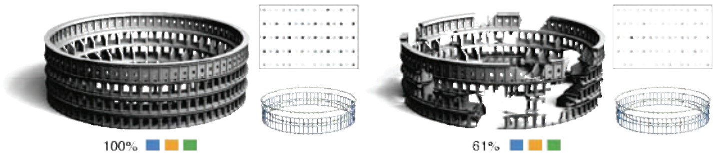
Robustness



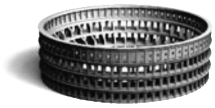
Robustness



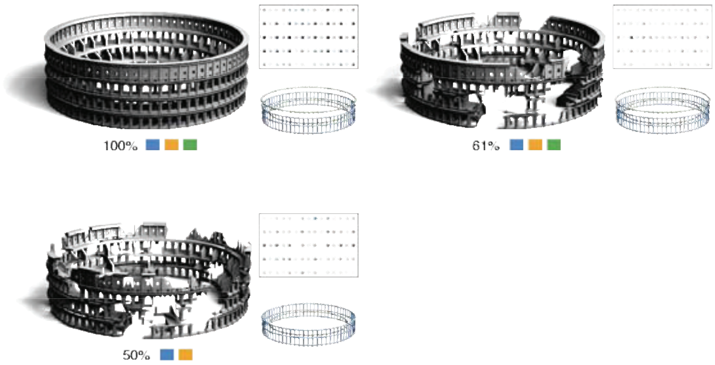
Robustness



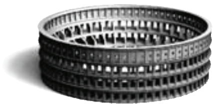
Geometry Synthesis

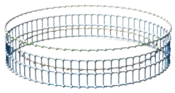


Robustness

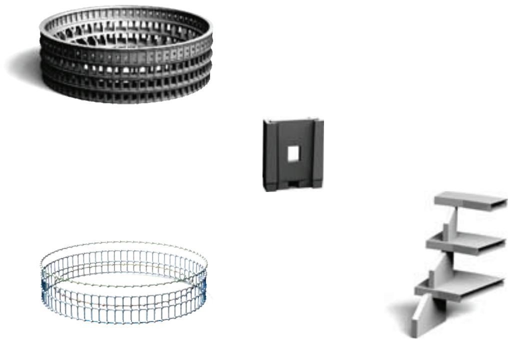


Geometry Synthesis

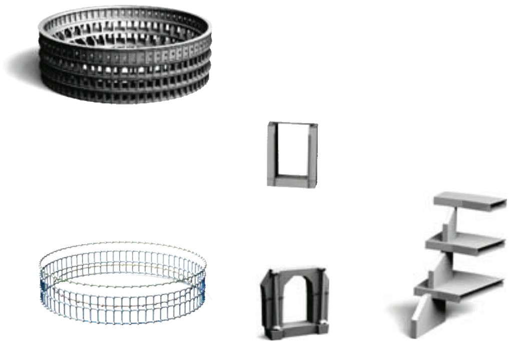




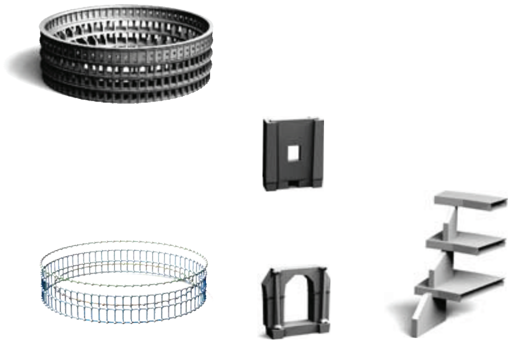
Geometry Synthesis



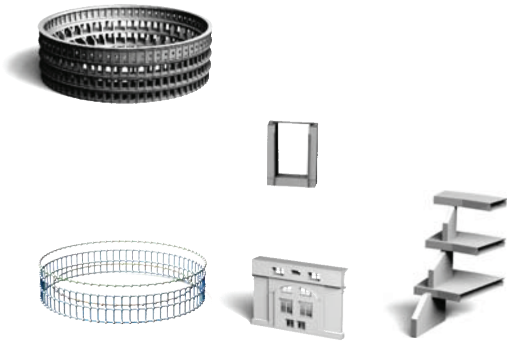
Geometry Synthesis



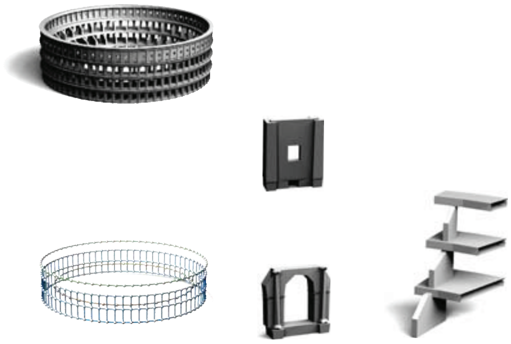
Geometry Synthesis



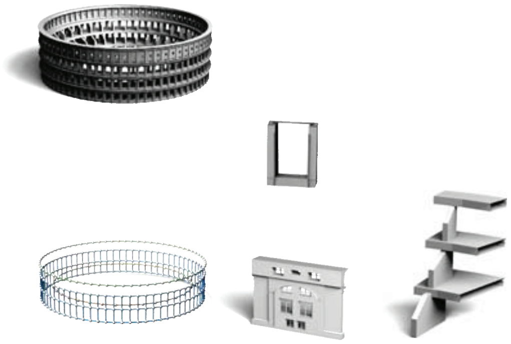
Geometry Synthesis



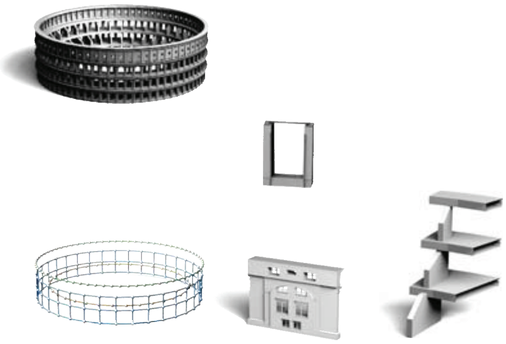
Geometry Synthesis



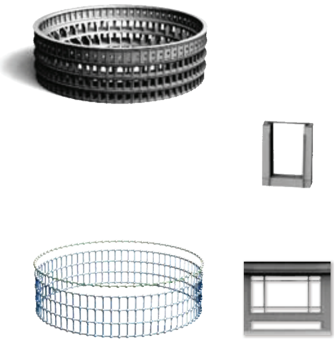
Geometry Synthesis



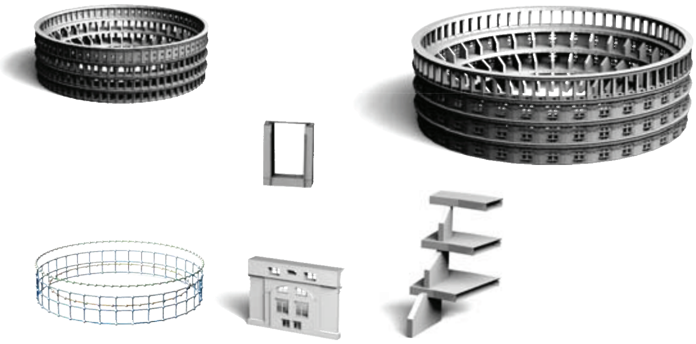
Geometry Synthesis



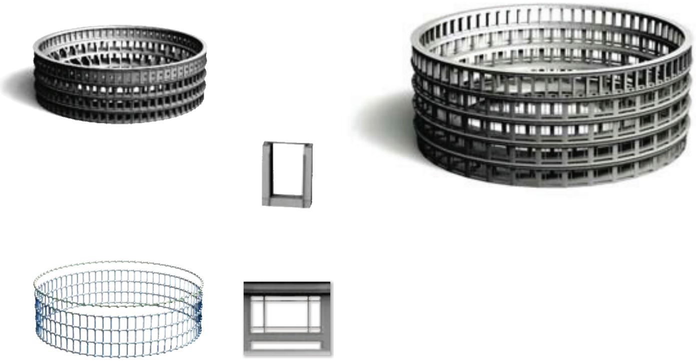
Geometry Synthesis



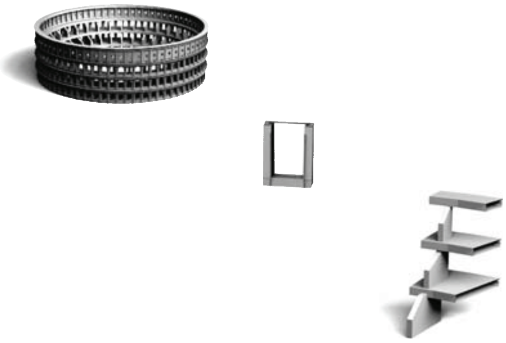
Geometry Synthesis



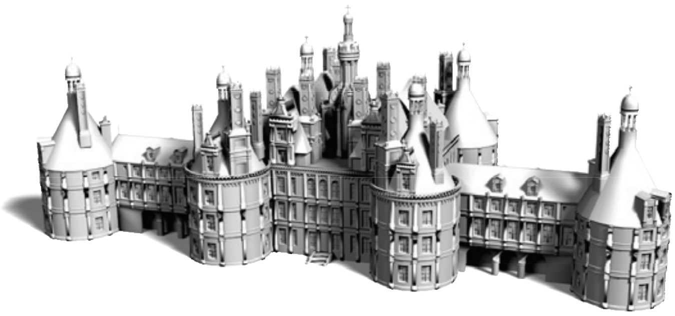
Geometry Synthesis



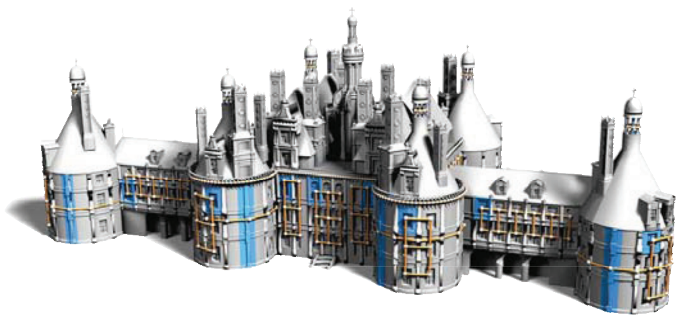
Geometry Synthesis



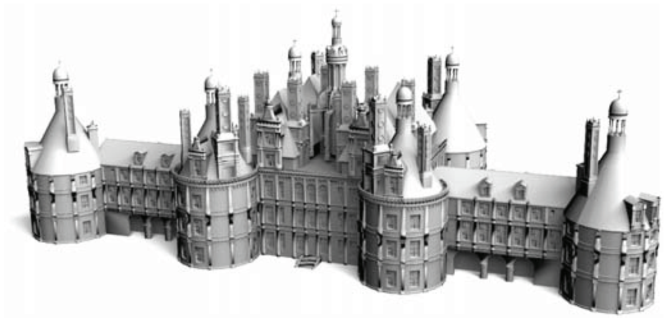
Chambord Castle



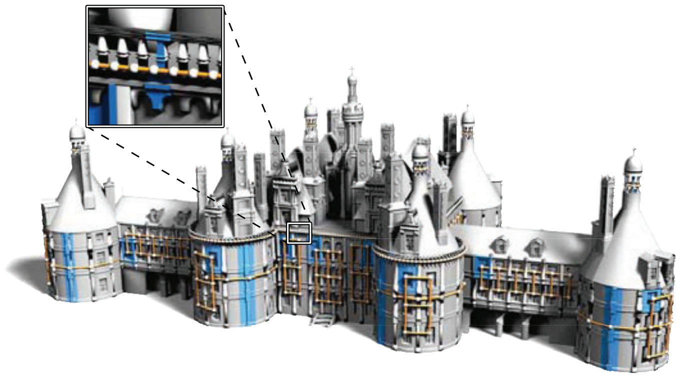
Chambord Castle



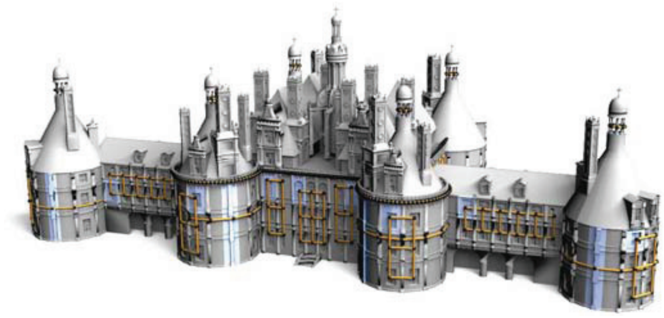
Chambord Castle



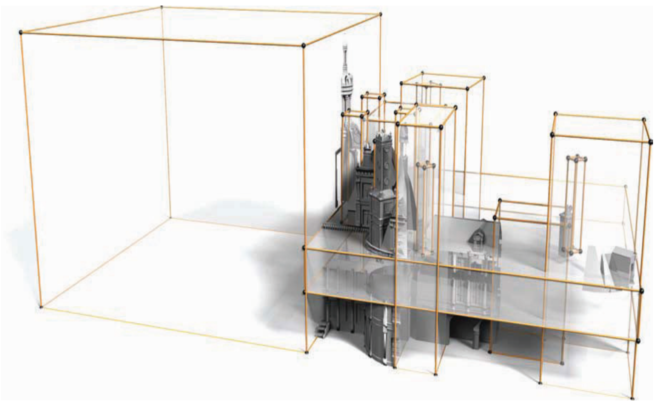
Chambord Castle



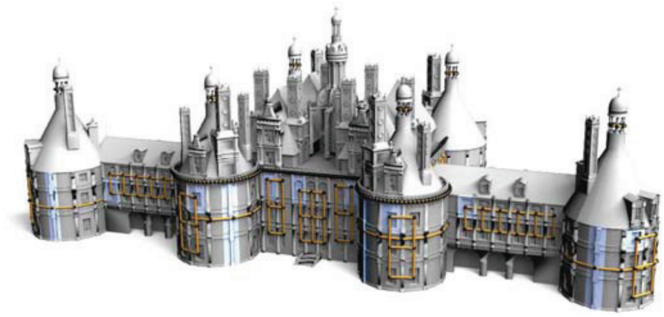
Chambord Castle



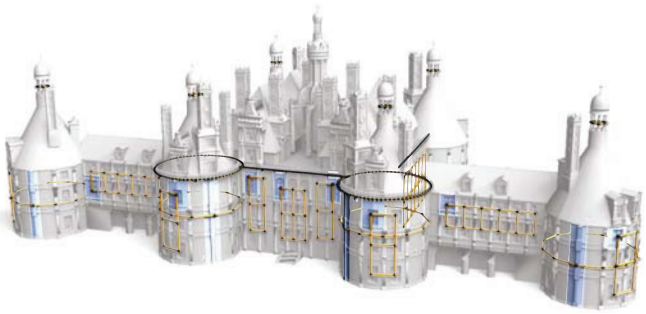
Chambord Castle [Mitra et al. `06]



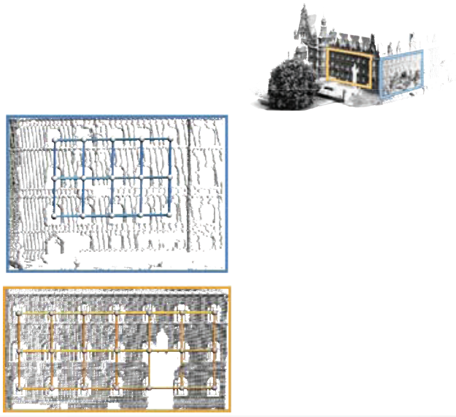
Chambord Castle



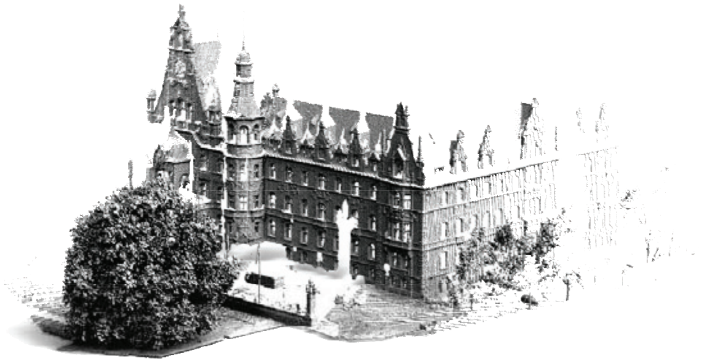
Chambord Castle



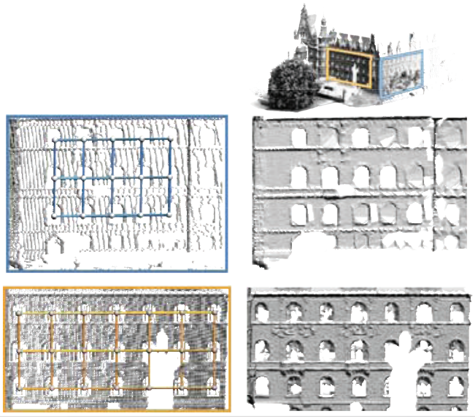
Outdoor Scan



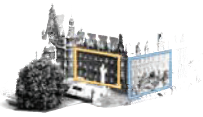
Outdoor Scan



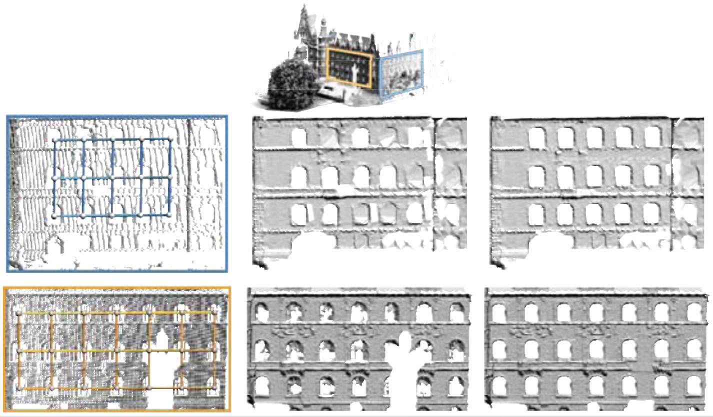
Scan Completion



Outdoor Scan

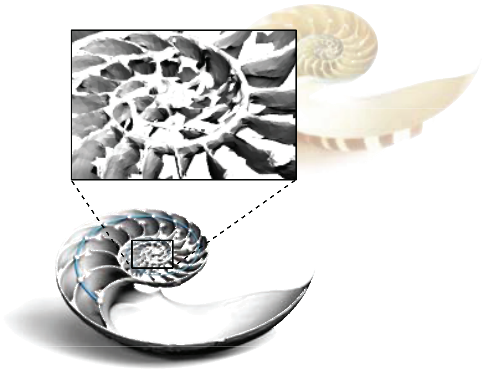


Scan Completion

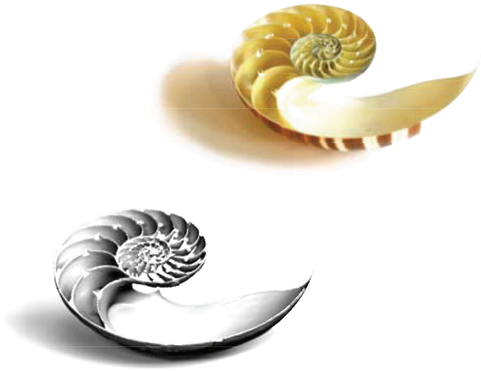


Chambered Nautilus

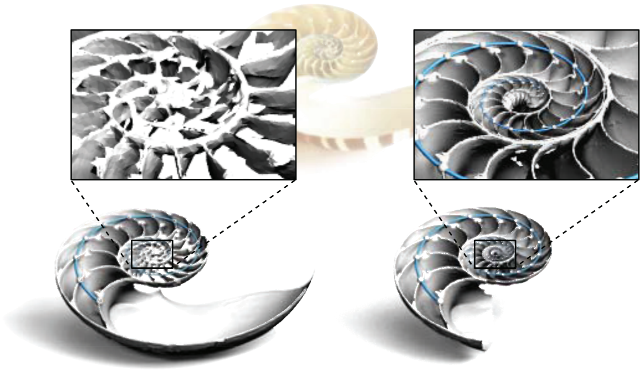
Chambered Nautilus



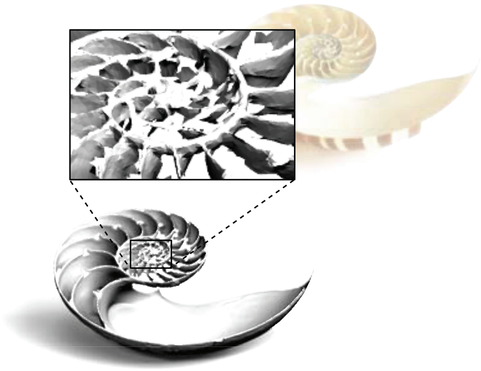
Chambered Nautilus



Scan Completion



Chambered Nautilus



Geometry Synthesis

Geometry Synthesis

Geometry Synthesis

Geometry Synthesis

Geometry Synthesis

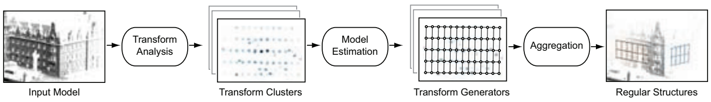
Geometry Synthesis

Observations

Observations

- Warped structures

Structure Discovery



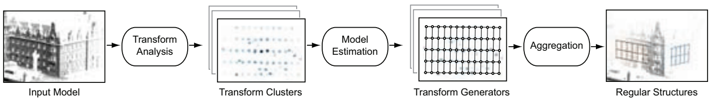
Input Model
Transform Analysis
Transform Clusters
Model Estimation
Transform Generators
Aggregation
Regular Structures

Observations

- Warped structures
- Size of grid *vs* accuracy

Structure Discovery

- Algorithm is fully automatic



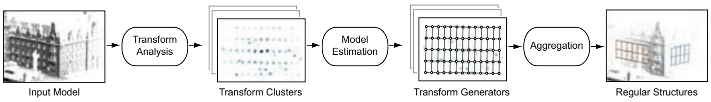
Input Model
Transform Analysis
Transform Clusters
Model Estimation
Transform Generators
Aggregation
Regular Structures

Observations

- Warped structures
- Size of grid *vs* accuracy
- Choice of parameters

Structure Discovery

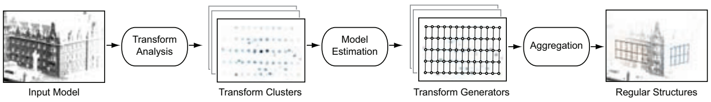
- Algorithm is fully automatic
- Requires no prior information on size, shape, or location of repetitive elements



Input Model
Transform Analysis
Transform Clusters
Model Estimation
Transform Generators
Aggregation
Regular Structures

Structure Discovery

- Algorithm is fully automatic
- Requires no prior information on size, shape, or location of repetitive elements
- Robust, efficient, independent of dimension
  ➙ *general tool for scientific data analysis*



Acknowledgements

- **Funding Agencies:**
  Austrian Science Fund (FWF)
  Darpa HR0011-05-1-0007
  NIH GM-072970
  NSF FRG-0354543
  TCS
- **Data Source:**
  Institute of Cartography and Geoinformatics,
  Leibniz University, Germany

Acknowledgements

Acknowledgements

- **Funding Agencies:**
  Austrian Science Fund (FWF)
  Darpa HR0011-05-1-0007
  NIH GM-072970
  NSF FRG-0354543
  TCS
- **Data Source:**
  Institute of Cartography and Geoinformatics,
  Leibniz University, Germany

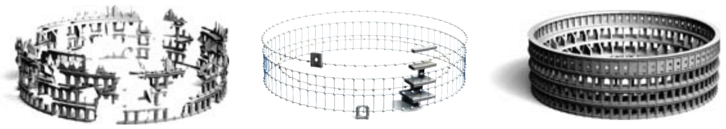
- **Scanning, code snippets:**
  Michael Eigensatz
  Balint Miklos
  Heinz Schmiedhofer

Acknowledgements

- **Funding Agencies:**
  Austrian Science Fund (FWF)
  Darpa HR0011-05-1-0007
  NIH GM-072970
  NSF FRG-0354543
  TCS

Thank You

Thank You



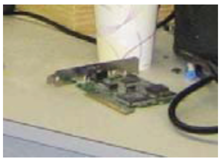
Computing Correspondences in Geometric Datasets

Symmetry

Symmetry Transforms

Motivation

Symmetry is everywhere

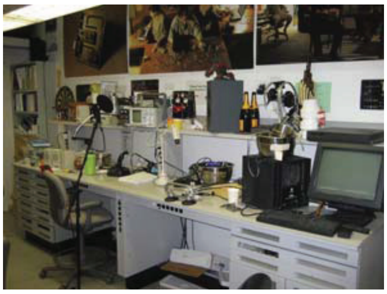


Local Symmetry

[Blum '78]
[Thrun '05]
[Simari '06]

Motivation

Symmetry is everywhere



Motivation

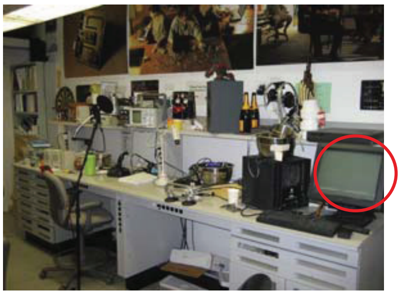
Symmetry is everywhere

Partial Symmetry

[Zabrodsky '95]
[Kazhdan '03]

Motivation

Symmetry is everywhere

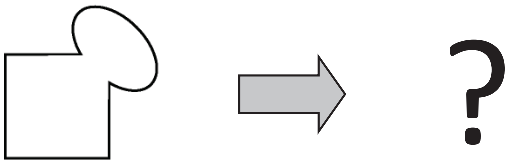


Perfect Symmetry

[Blum '64, '67]
[Wolter '85]
[Minovic '97]
[Martinet '05]

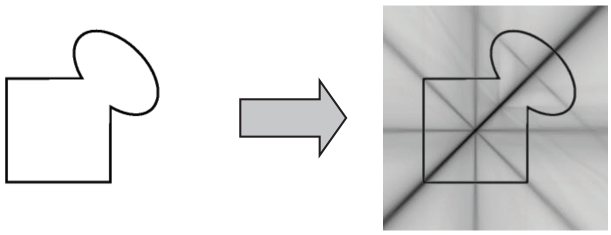
Goal

A computational representation that describes all planar symmetries of a shape



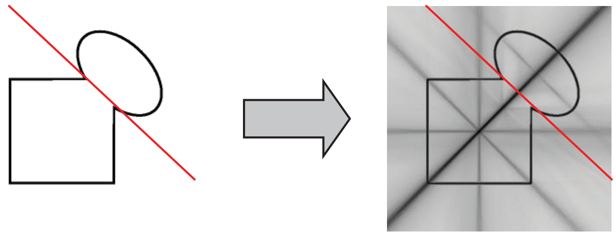
Symmetry Transform

A computational representation that describes all planar symmetries of a shape



Symmetry Transform

A computational representation that describes all planar symmetries of a shape

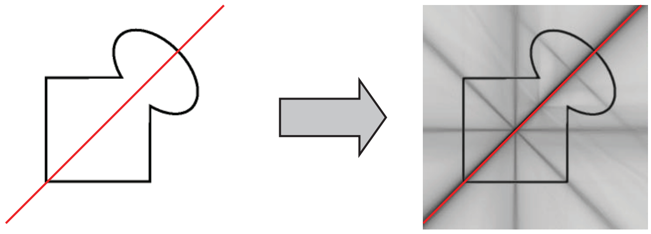


Partial Symmetry

Symmetry = 0.2

Symmetry Transform

A computational representation that describes all planar symmetries of a shape

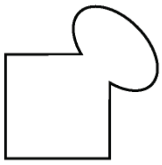


Perfect Symmetry

Symmetry = 1.0

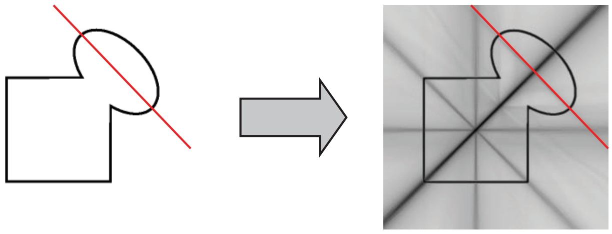
Symmetry Measure

Symmetry of a shape is measured by correlation with its reflection



Symmetry Transform

A computational representation that describes all planar symmetries of a shape

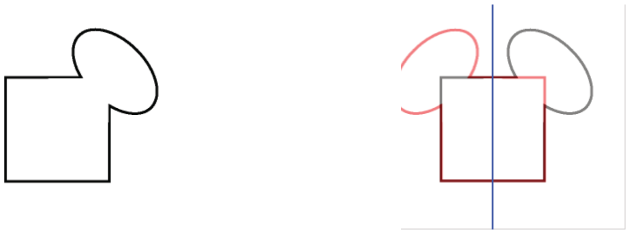


Local Symmetry

Symmetry = 0.3

Symmetry Measure

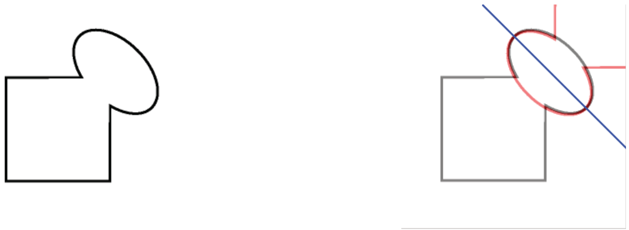
Symmetry of a shape is measured by correlation with its reflection



Symmetry = 0.7

Symmetry Measure

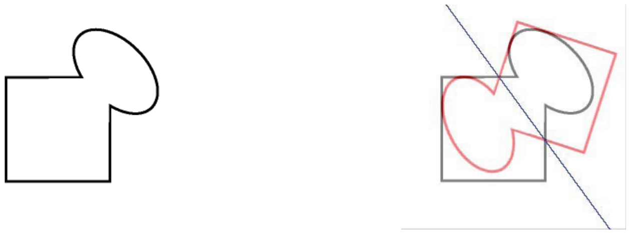
Symmetry of a shape is measured by correlation with its reflection



Symmetry = 0.3

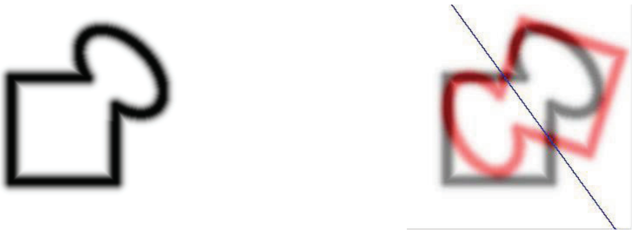
Symmetry Measure

Symmetry of a shape is measured by correlation with its reflection



Symmetry Measure

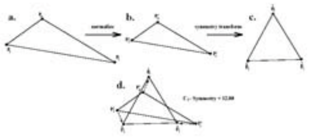
Symmetry of a shape is measured by correlation with its reflection



Symmetry = 0.1

Previous Work

Zabrodsky '95



Kazhdan '03

Thrun '05

Martinet '05

Symmetry Distance

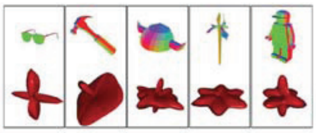
Define the *Symmetry Distance* of a function ***f*** with respect to any transformation γ as the L^2 distance between ***f*** and the nearest function invariant to γ

Can show that Symmetry Measure $D(f,\gamma)=f\cdot\gamma(f)$ is related to symmetry distance by

$$D(f,\gamma)=-2SD^2+\|f\|^2$$

Previous Work

Zabrodsky '95



Kazhdan '03

Thrun '05

Martinet '05

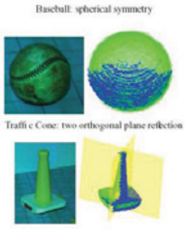
Previous Work

Zabrodsky '95

Kazhdan '03

Thrun '05

Martinet '05


Baseball: spherical symmetry
Traffic Cone: two orthogonal plane reflection

Computing Discrete Transform

Brute Force	$O(n^6)$
Convolution	$O(n^5 \text{Log } n)$
Monte-Carlo	

$O(n^2)$ normal directions

X

$O(n^3 \text{Log } n)$ per direction

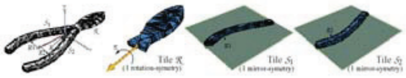
Previous Work

Zabrodsky '95

Kazhdan '03

Thrun '05

Martinet '05



Computing Discrete Transform

Brute Force	$O(n^6)$
Convolution	$O(n^5 \text{Log } n)$
Monte-Carlo	$O(n^4)$ For 3D meshes

- Most of the dot product contains zeros.
- Use Monte-Carlo Importance Sampling.

Computing Discrete Transform

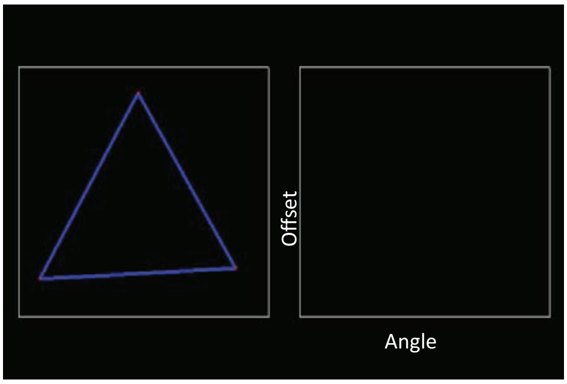
Brute Force	$O(n^6)$
Convolution	
Monte-Carlo	

$O(n^3)$ planes

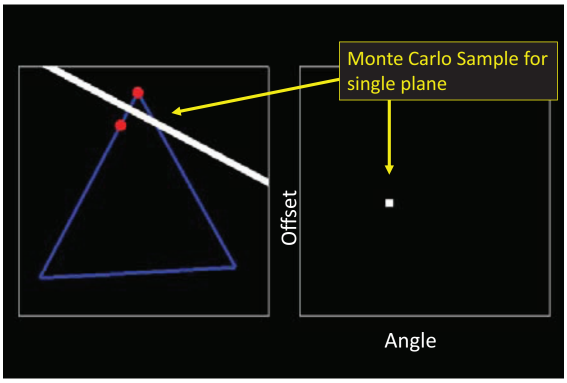
X

$O(n^3)$ dot product

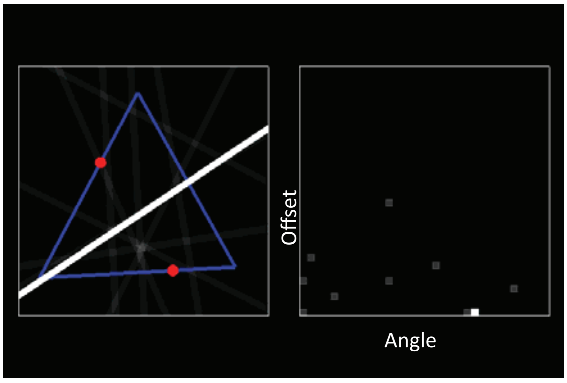
Monte Carlo



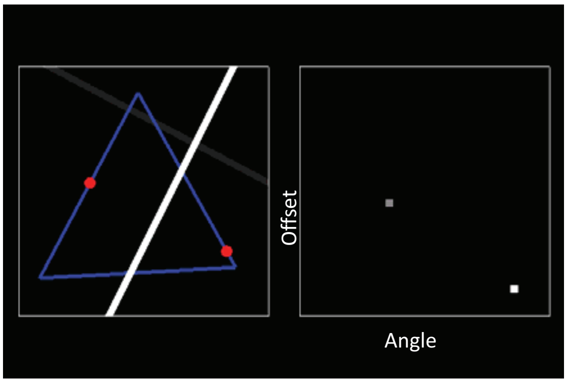
Monte Carlo



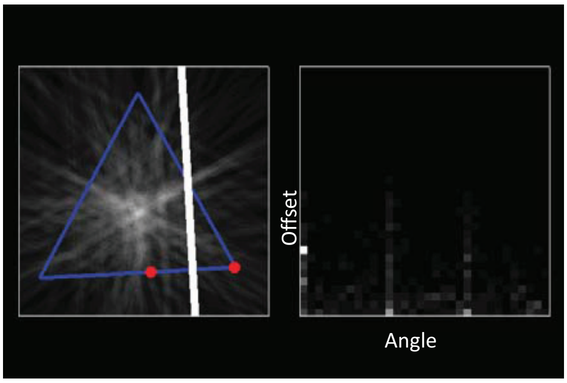
Monte Carlo



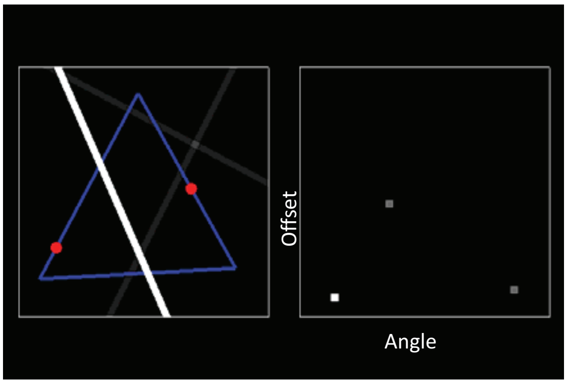
Monte Carlo



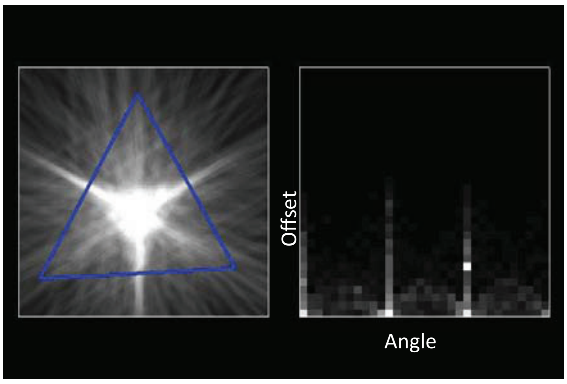
Monte Carlo



Monte Carlo

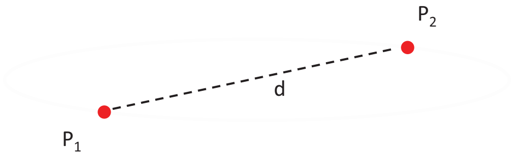


Monte Carlo



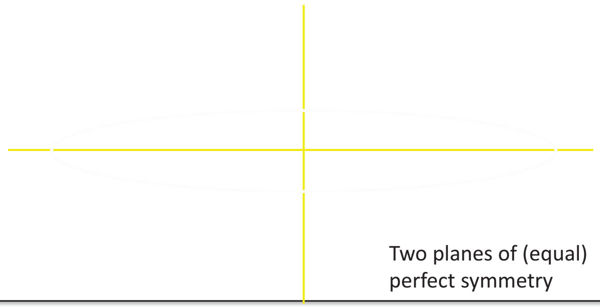
Weighting Samples

Need to weight sample pairs by the inverse of the distance between them



Weighting Samples

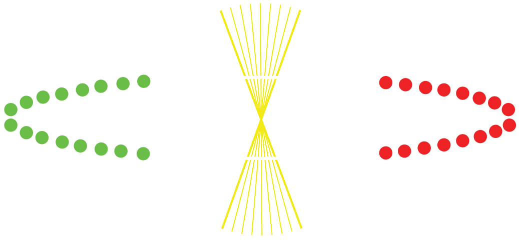
Need to weight sample pairs by the inverse of the distance between them



Two planes of (equal) perfect symmetry

Weighting Samples

Need to weight sample pairs by the inverse of the distance between them



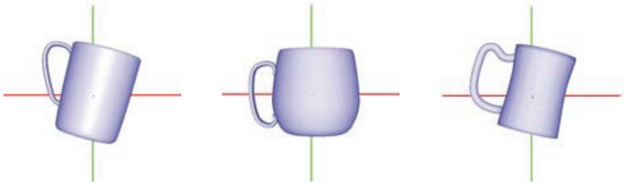
Votes for vertical plane…

Application: Alignment

Motivation:

Composition of range scans

Morphing



PCA Alignment

Application: Alignment

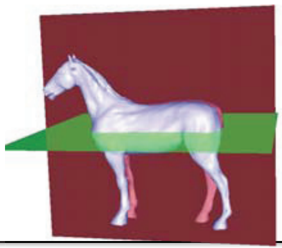
Approach:

Perpendicular planes with the greatest symmetries create a symmetry-based coordinate system.

Application: Alignment

Approach:

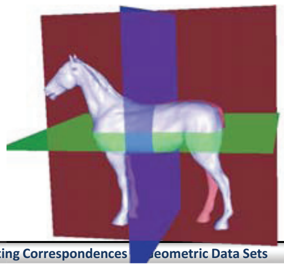
Perpendicular planes with the greatest symmetries create a symmetry-based coordinate system.



Application: Alignment

Approach:

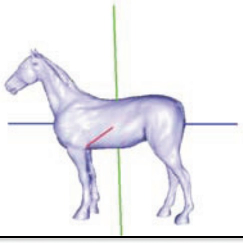
Perpendicular planes with the greatest symmetries create a symmetry-based coordinate system.



Application: Alignment

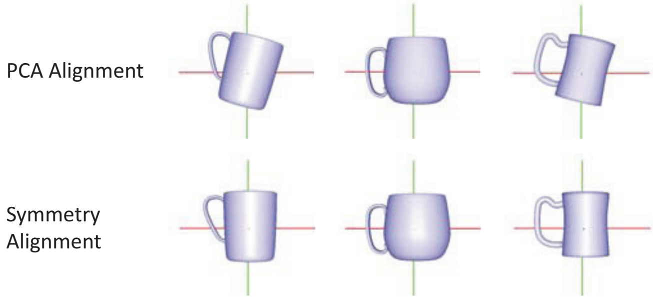
Approach:

Perpendicular planes with the greatest symmetries create a symmetry-based coordinate system.



Application: Alignment

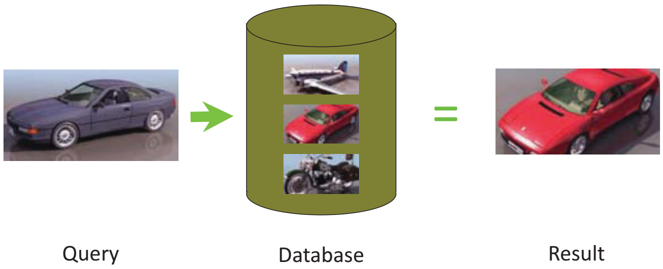
Results:



Application: Matching

Motivation:

Database searching



Application: Matching

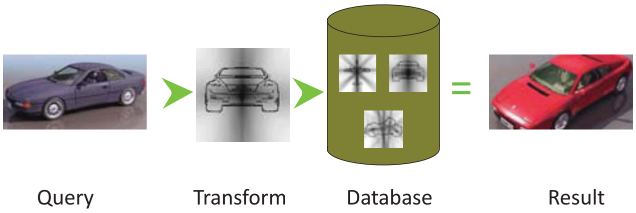
Observation:

All chairs display similar principal symmetries

Application: Matching

Approach:

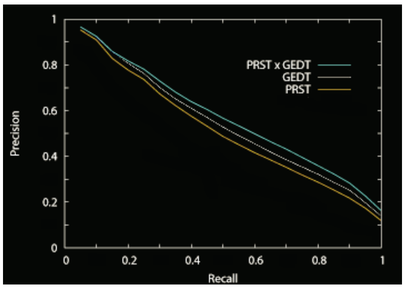
Use Symmetry transform as shape descriptor

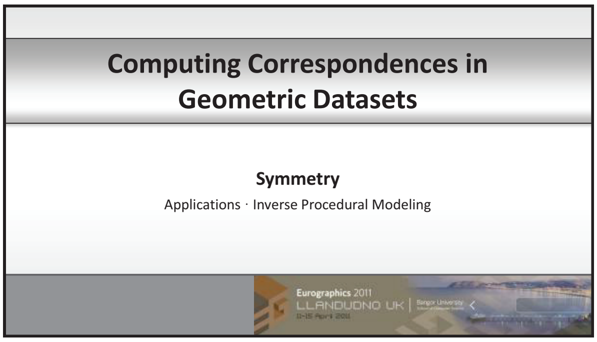


Application: Matching

Results:

Symmetry provides orthogonal information about models and can therefore be combined with other descriptors



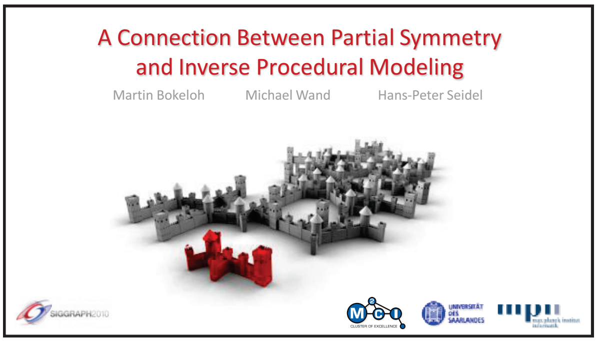


Computing Correspondences in Geometric Datasets

Symmetry

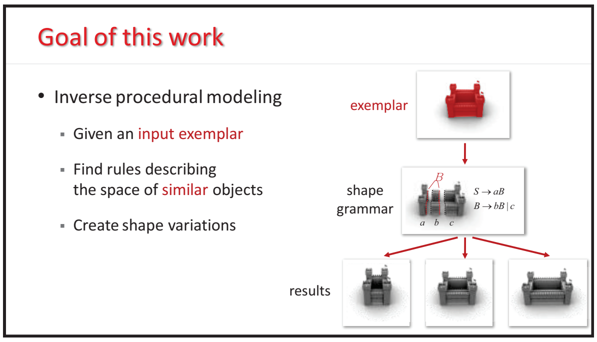
Applications · Inverse Procedural Modeling

Eurographics 2011
LLANDUDNO UK
Bangor University
11-15 April 2011



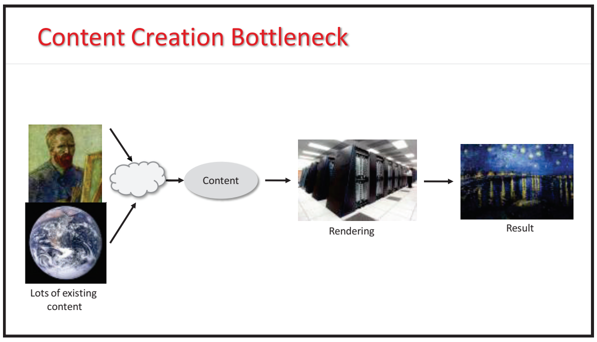
A Connection Between Partial Symmetry and Inverse Procedural Modeling

Martin Bokeloh Michael Wand Hans-Peter Seidel

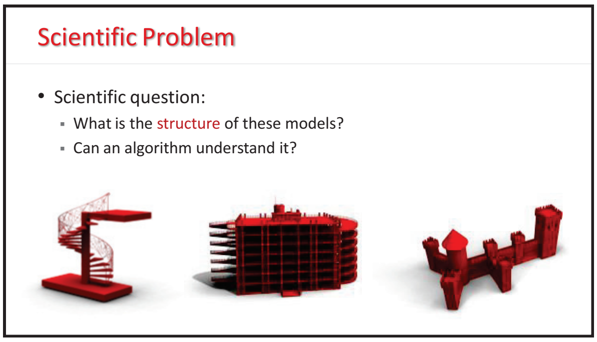


Goal of this work

- Inverse procedural modeling
  - Given an input exemplar
  - Find rules describing the space of similar objects
  - Create shape variations

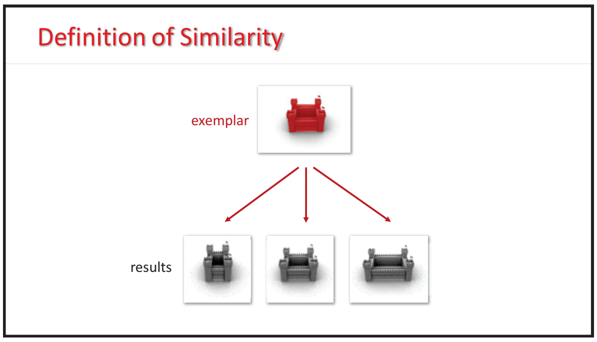


Content Creation Bottleneck



Scientific Problem

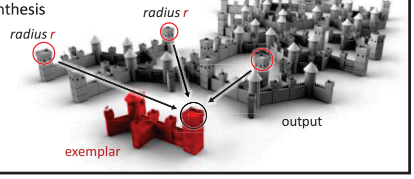
- Scientific question:
  - What is the structure of these models?
  - Can an algorithm understand it?



Definition of Similarity

What are similar models?

- *r*-similarity
  - User parameter *r*
  - Local neighborhoods match exemplar
  - As in texture synthesis



RELATED WORK

Related Work

- Procedural Modeling
  - Plants [Prusinkiewicz and Lindenmayer 1990], [Deussen et al. 1998]
  - Cities & buildings [Parish and Müller 2001], [Wonka et al. 2003], [Müller et al. 2006]

User specifies grammar

Related Work

- Inverse Procedural Modeling
  - Vector graphics [Hart et al. 1997], [Yeh et al. 2009], [Šť'ava et al 2010]

No continuous surfaces

  - From images [Aliaga et al. 2007], [Müller et al. 2007], [Neubert et al. 2007], [Tan et al. 2007], [Xiao et al. 2009]

Predefined class of grammars

Related Work

- Texture Synthesis
  - 2D texture synthesis [Efros and Leung 1999], [Wei and Levoy 2000], [Kwatra et al. 2003], [Kwatra et al. 2005]
  - 3D geometry [Lai et al. 2005], [Nguyen et al. 2005], [Chen and Meng 2009], [Zhou et al. 2006], [Zelinka and Garland 2006], [Bhat et al. 2004], [Sharf et al. 2004], [Lagae et al. 2005]
  - Example-based model synthesis [Merrel et al. 2007], [Merrel and Manocha 2008]

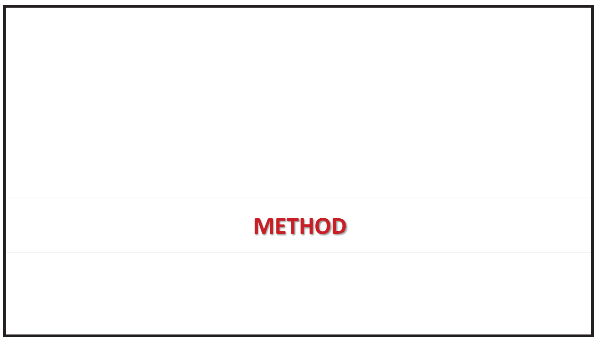
Hard optimization problem,
no procedural description

Related Work

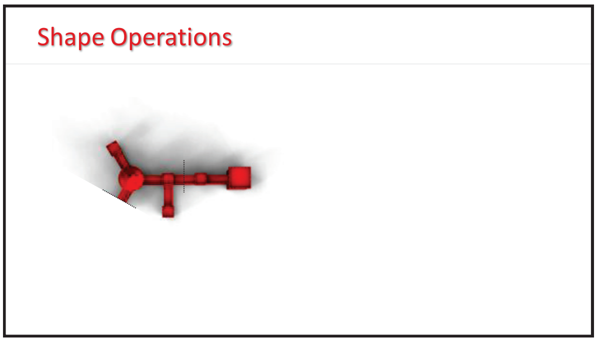
- Symmetry Detection

[Thrun and Wegbreit 2005], [Mitra et al. 2006], [Podolak et al. 2006], [Gal and Cohen-Or 2006], [Mitra et al. 2007], [Pauly et al. 2008], [Bokeloh et al. 2009]

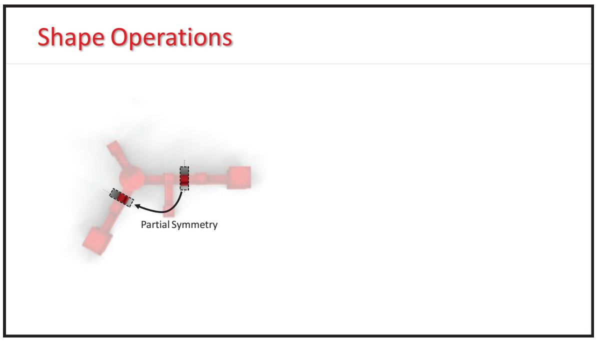
We build upon this work



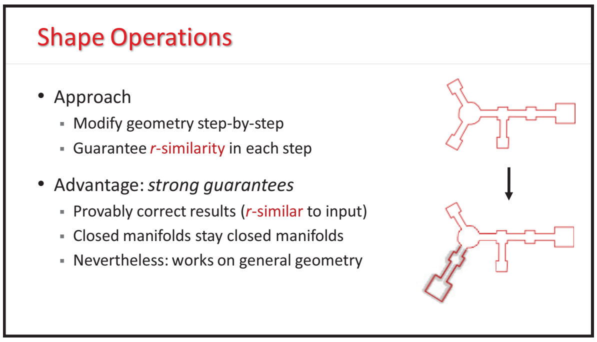
METHOD



Shape Operations



Shape Operations

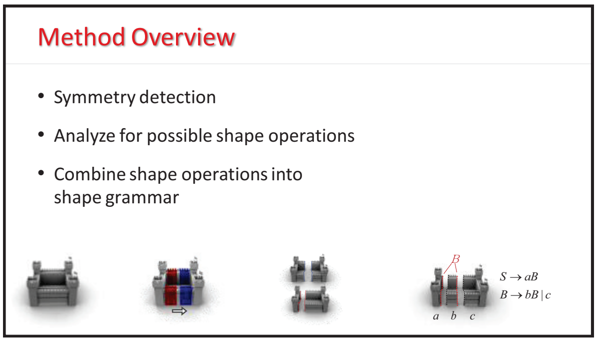


Shape Operations

- Approach
  - Modify geometry step-by-step
  - Guarantee *r*-similarity in each step
- Advantage: *strong guarantees*
  - Provably correct results (*r*-similar to input)
  - Closed manifolds stay closed manifolds
  - Nevertheless: works on general geometry

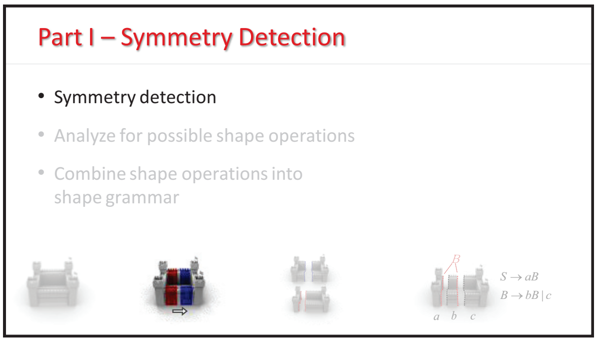
Shape Operations

- Questions
  - How can we find shape operations?
  - What is the structure of their interdependencies?



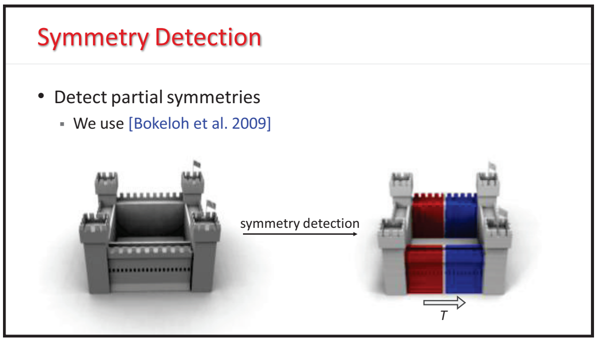
Method Overview

- Symmetry detection
- Analyze for possible shape operations
- Combine shape operations into shape grammar



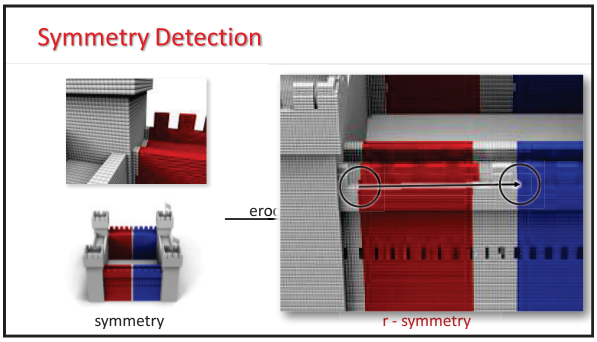
Part I – Symmetry Detection

- Symmetry detection
- Analyze for possible shape operations
- Combine shape operations into shape grammar

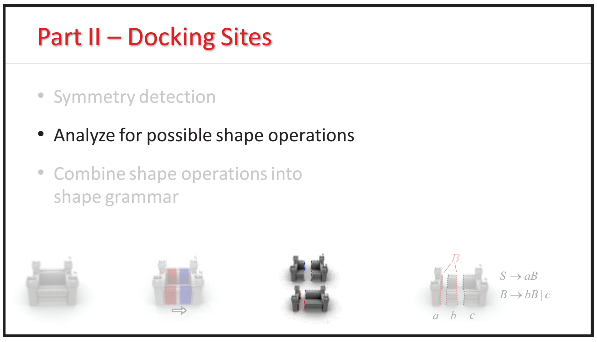


Symmetry Detection

- Detect partial symmetries
  - We use [Bokeloh et al. 2009]

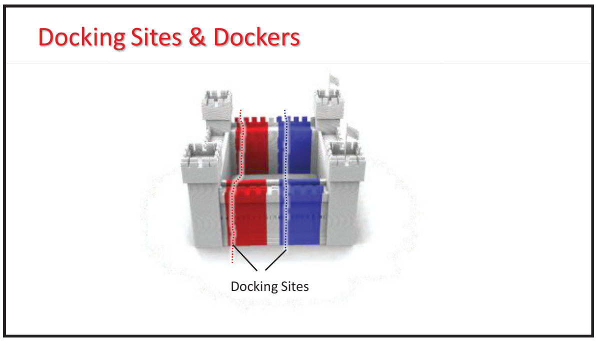


Symmetry Detection

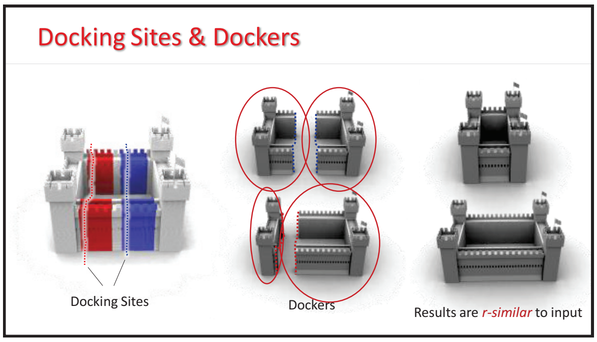


Part II – Docking Sites

- Symmetry detection
- Analyze for possible shape operations
- Combine shape operations into shape grammar

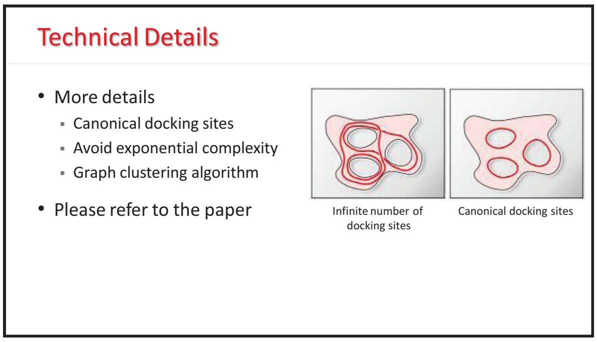


Docking Sites & Dockers



Docking Sites & Dockers

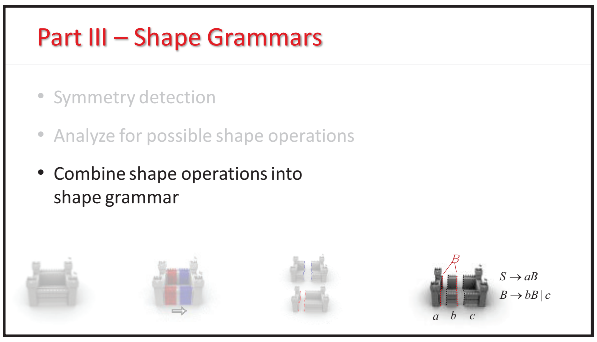
Results are *r-similar* to input



Technical Details

- More details
  - Canonical docking sites
  - Avoid exponential complexity
  - Graph clustering algorithm
- Please refer to the paper

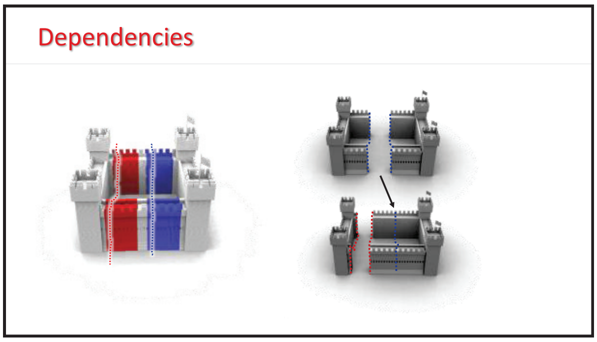
Infinite number of docking sites
Canonical docking sites



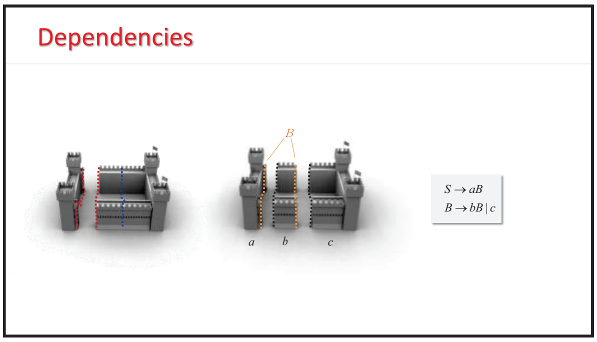
Part III – Shape Grammars

- Symmetry detection
- Analyze for possible shape operations
- Combine shape operations into shape grammar

$S \rightarrow aB$
$B \rightarrow bB \mid c$

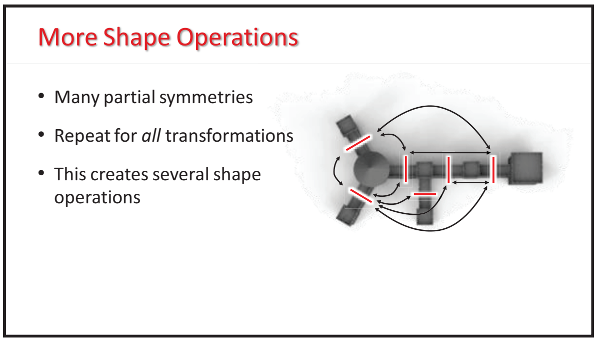


Dependencies



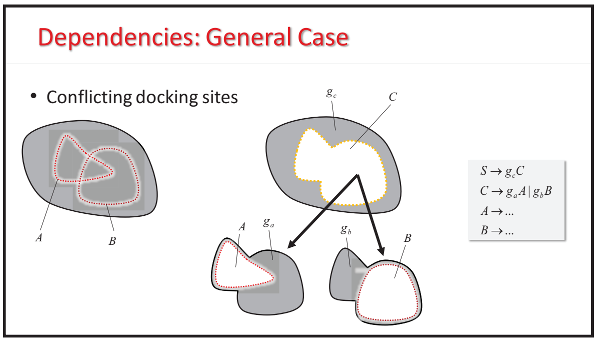
Dependencies

$S \rightarrow aB$
$B \rightarrow bB \mid c$



More Shape Operations

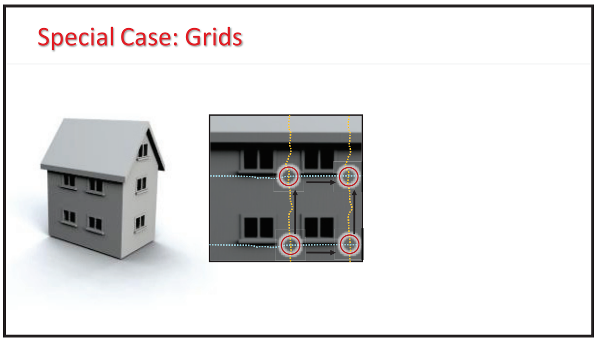
- Many partial symmetries
- Repeat for *all* transformations
- This creates several shape operations



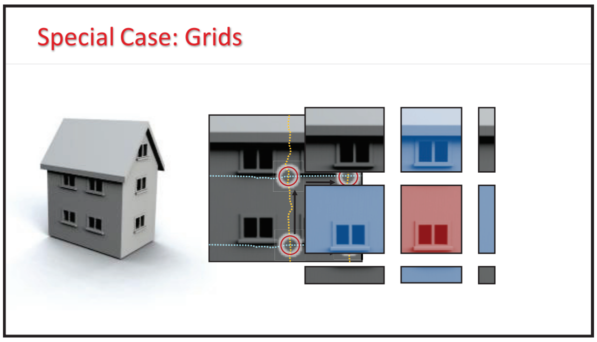
Dependencies: General Case

- Conflicting docking sites

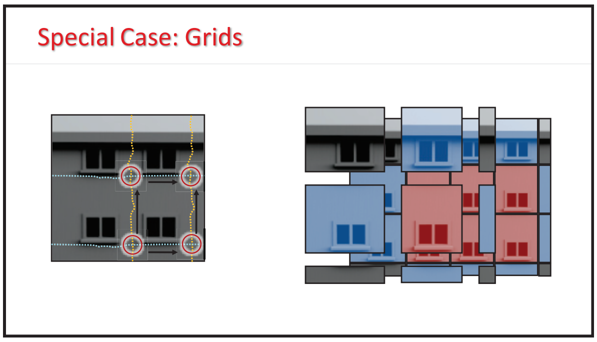
$S \rightarrow g_c C$
$C \rightarrow g_a A \mid g_b B$
$A \rightarrow \ldots$
$B \rightarrow \ldots$



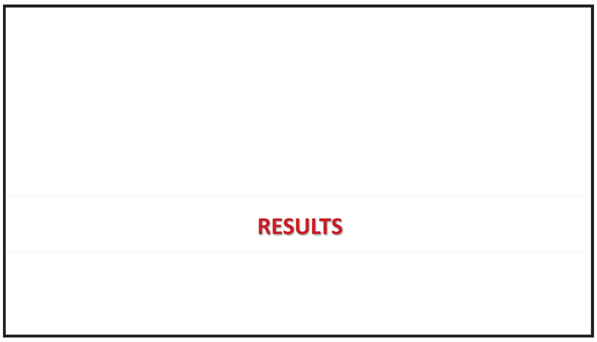
Special Case: Grids



Special Case: Grids

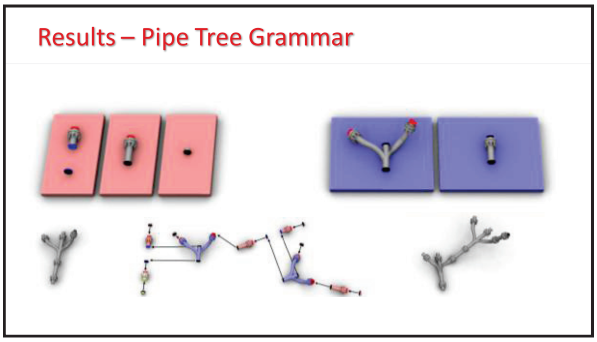


Special Case: Grids

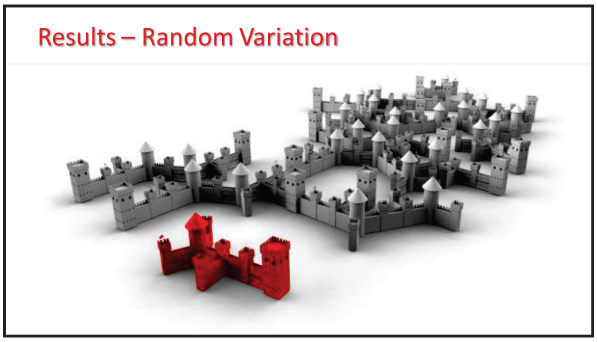


RESULTS

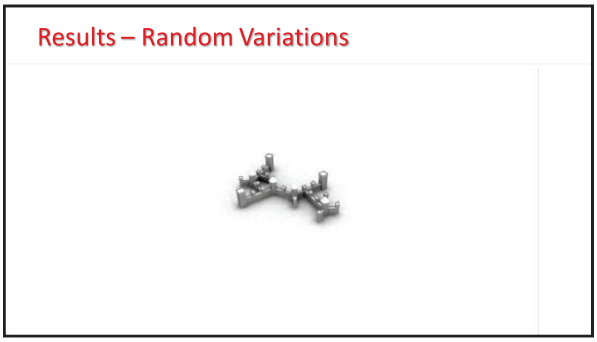
Results – Pipe Tree



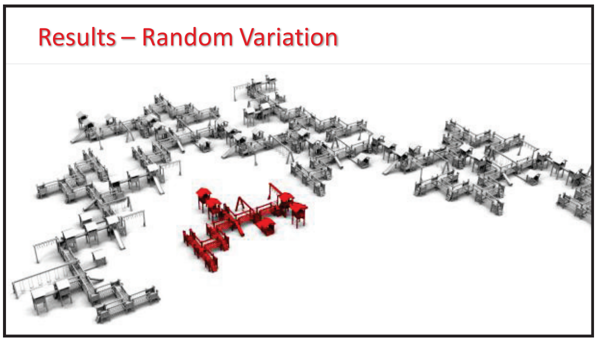
Results – Pipe Tree Grammar



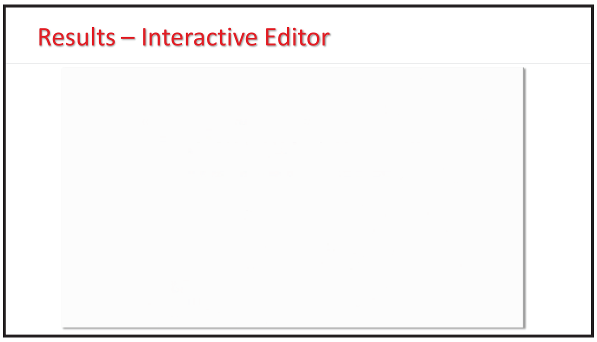
Results – Random Variation



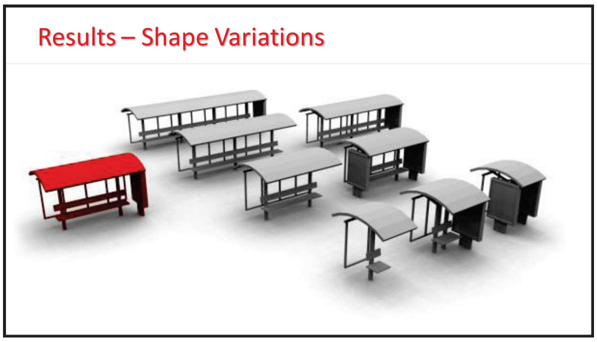
Results – Random Variations



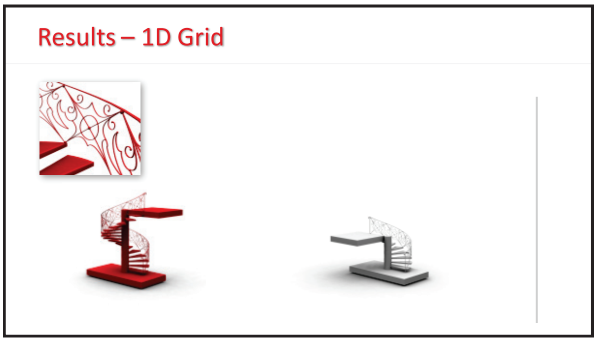
Results – Random Variation



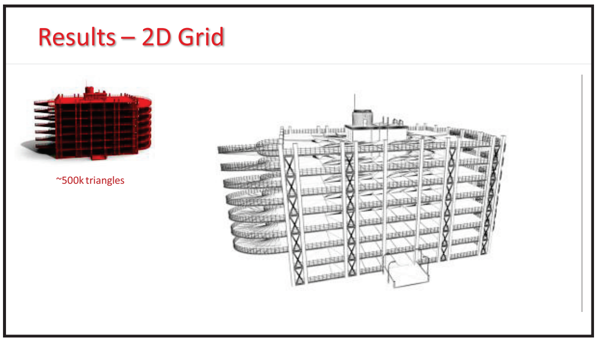
Results – Interactive Editor



Results – Shape Variations

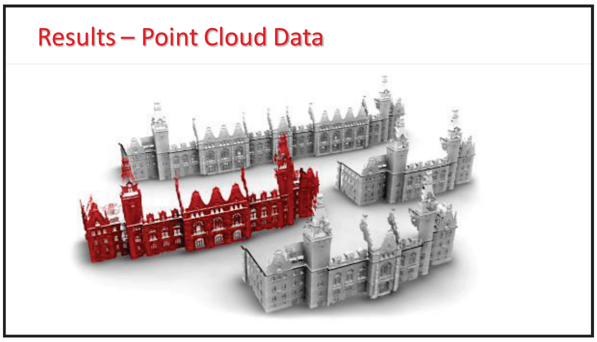


Results – 1D Grid

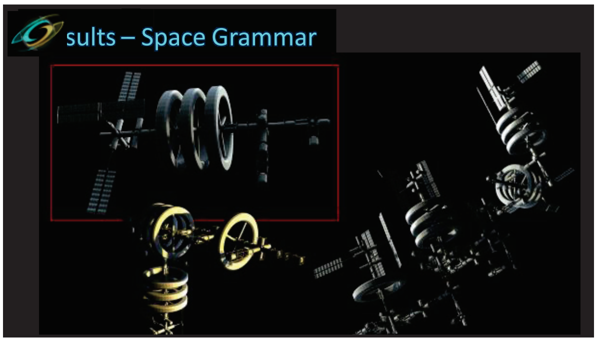


Results – 2D Grid

~500k triangles

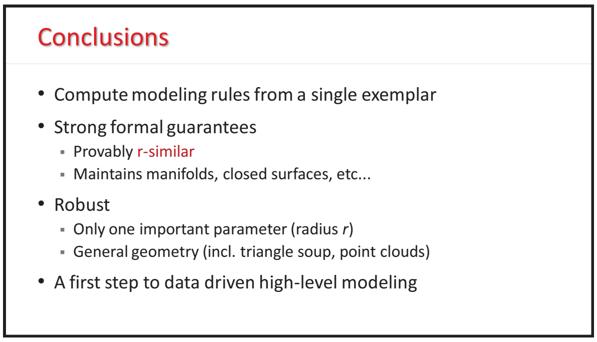


Results – Point Cloud Data



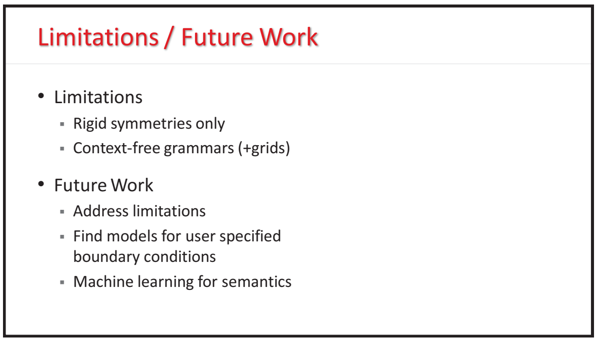
sults – Space Grammar

CONCLUSIONS / FUTURE WORK



Conclusions

- Compute modeling rules from a single exemplar
- Strong formal guarantees
  - Provably r-similar
  - Maintains manifolds, closed surfaces, etc...
- Robust
  - Only one important parameter (radius *r*)
  - General geometry (incl. triangle soup, point clouds)
- A first step to data driven high-level modeling



Limitations / Future Work

- Limitations
  - Rigid symmetries only
  - Context-free grammars (+grids)
- Future Work
  - Address limitations
  - Find models for user specified boundary conditions
  - Machine learning for semantics

Acknowledgements

- We would like to thank
  - Alexander Berner, Qi-Xing Huang, Matthias Hullin, Leonidas Guibas, Gerd Wolf, Ivo Ihrke, Tobias Ritschel, Thorsten Thormählen, Qing Fang, and the anonymous reviewers
  - This work has been partially supported by the Cluster of Excellence “Multi-Modal Computing and Interaction”.

Thank you for your attention!