The Geometry Processing Pipeline

Geometric Computer Vision

GCV v2021.1, Module 1

Alexey Artemov, Spring 2021

Lecture Outline

§1. The geometry processing pipeline [45 min]

1.1. Goals of 3D/geometric computer vision systems

1.2. Common stages of geometry processing

1.3. Scanning [next video]

1.4. Registration

1.5. Reconstruction and meshing

1.6. Postprocessing [next videos]

Lecture Outline

§2. 3D representations in computer vision/graphics [15 min, Friday]

2.1. Directly measurable: multiple-view images, range-images, point clouds, volumes

2.2. Derived: surface meshes, implicit functions

2.3. Higher-level: CAD, shape programs

§1. The geometry processing pipeline

Goals of 3D/geometric computer vision systems

1.1. Goals of 3D/geometric computer vision systems

§1. The geometry processing pipeline

- Two main aspects:
- **Construct** 3D geometry representations suitable for various tasks from raw data (range images, volumetric CT and MRI, LIDAR)
  - Usually involves multiple steps going from low to high level
  - As an intermediate tool, requires analysis, e.g. segmentation
  - E.g.: Points →meshes → parametrized patch layout
- **Manipulate and analyze** geometry:
  - Deformations, boolean operations, comparisons, physically-based deformations (related to CAE)

1.1. Goals of 3D/geometric computer vision systems

§1. The geometry processing pipeline

- **The geometry processing pipeline:** a highly modular sequence of interrelated stages for manipulations with 3D data, commonly for 3D reconstruction and understanding
  - Convenient concept of conversions between 3D representations
  - Flow: going from low-level to higher-level representations/properties
  - Modularity: injecting methods/models easier

1.1. Goals of 3D/geometric computer vision systems

§1. The geometry processing pipeline

- Why need to study an entire pipeline for 3D processing? Don't 3D scanners have it all?
  - Most hardware systems for 3D acquisition: standard/proprietary algorithms, no customization, limited conversion options
  - Being able to intervene at any stage: flexibility, "debugging", performance gains
- **Today: go over the "standard" reconstruction pipeline for 3d scanning**
- Consider two types of problems with existing techniques (how these can be addressed by ML-based methods?):
  - "Low-level": related to local surface properties, e.g., noise, normals, curvature, outliers,
  - "High-level": involve object semantic (e.g., high level part segmentation)

Common stages of geometry processing

1.2. Common stages of geometry processing

§1. The geometry processing pipeline

Scanning: results in range images ⇨ **Registration:** bring all range images to one coordinate system ⇨ **Stitching/ reconstruction:** Integration of scans into a single mesh ⇨ **Postprocess:**
- Topological and geometric filtering
- Remeshing
- Compression

1.2. Common stages of geometry processing

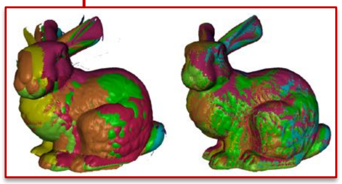
§1. The geometry processing pipeline

Scanning: results in range images ⇨ **Registration:** bring all range images to one coordinate system ⇨ **Stitching/ reconstruction:** Integration of scans into a single mesh ⇨ **Postprocess:**
- Topological and geometric filtering
- Remeshing
- Compression

1.2. Common stages of geometry processing

§1. The geometry processing pipeline

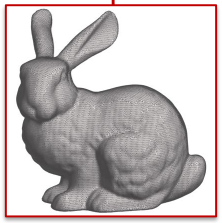
Scanning: results in range images ⇨ **Registration:** bring all range images to one coordinate system ⇨ **Stitching/ reconstruction:** Integration of scans into a single mesh ⇨ **Postprocess:**
- Topological and geometric filtering
- Remeshing
- Compression



1.2. Common stages of geometry processing

§1. The geometry processing pipeline

Scanning: results in range images ⇨ **Registration:** bring all range images to one coordinate system ⇨ **Stitching/ reconstruction:** Integration of scans into a single mesh ⇨ **Postprocess:**
- Topological and geometric filtering
- Remeshing
- Compression

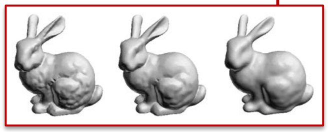


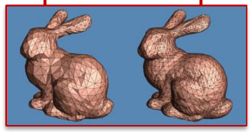
Slide Credit: Denis Zorin

1.2. Common stages of geometry processing

§1. The geometry processing pipeline

Scanning: results in range images ⇨ **Registration:** bring all range images to one coordinate system ⇨ **Stitching/ reconstruction:** Integration of scans into a single mesh ⇨ **Postprocess:**
- Topological and geometric filtering
- Remeshing
- Compression





Slide Credit: Denis Zorin

1.2. Common stages of geometry processing

§1. The geometry processing pipeline

1.2. Common stages of geometry processing

§1. The geometry processing pipeline

- The "standard" geometry pipeline for 3d scanning:
  - Scanning → registration → reconstruction → postprocessing
- From low-level to higher-level representations
- Natural modularity, allows extension/injection of stages

Skoltech
Skolkovo Institute of Science and Technology

Scanning

Skoltech
Skolkovo Institute of Science and Technology

1.3. Scanning

§1. The geometry processing pipeline

- Analyze a real-world object or environment to collect data on its shape/appearance
  - Many technologies: contact, optical, computed tomography, structured light…

- **Today: do not consider the first step in detail (obtaining depth data)**
- Assume depth images/range scans are available
- Focusing on the next steps
- **Next week:** detailed lecture about depth acquisition

Skoltech
Skolkovo Institute of Science and Technology

Registration

Skoltech
Skolkovo Institute of Science and Technology

1.2. Registration: context

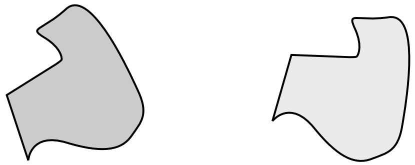
§1. The geometry processing pipeline

1.2. Registration: problem statement

Skoltech
Skolkovo Institute of Science and Technology

§1. The geometry processing pipeline

M_1 M_2



$$M_1 \approx T(M_2)$$

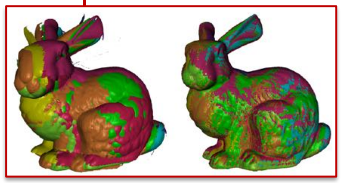
T: Translation + Rotation

1.2. Registration: context

Skoltech
Skolkovo Institute of Science and Technology

§1. The geometry processing pipeline

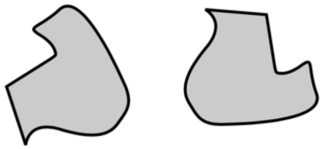
Scanning: results in range images ⇨ **Registration:** bring all range images to one coordinate system ⇨ **Stitching/ reconstruction:** Integration of scans into a single mesh ⇨ **Postprocess:**
- Topological and geometric filtering
- Remeshing
- Compression



1.2. Registration: local vs global

Skoltech
Skolkovo Institute of Science and Technology

§1. The geometry processing pipeline



Global Registration
Arbitrary Transformation

Local Registration
"Small" Transformation

Given $M_1, \ldots, M_n$, find $T_2, \ldots, T_n$ such that

$$M_1 \approx T_2(M_2) \cdots \approx T_n(M_n)$$

1.2. Registration: correspondences

Skoltech
Skolkovo Institute of Science and Technology

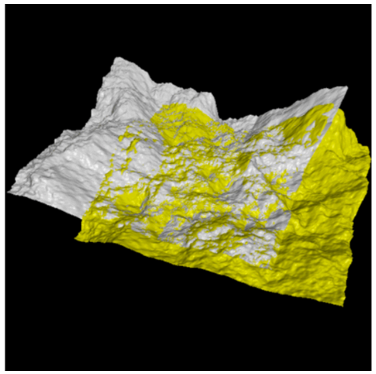
§1. The geometry processing pipeline

- How many points are needed to define a unique rigid transformation?
- The first problem is finding corresponding pairs!

Let $\mathbf{p}_i, \mathbf{q}_i$ define points on M_1 and M_2

$$\mathbf{p}_1 \rightarrow \mathbf{q}_1$$
$$\mathbf{p}_2 \rightarrow \mathbf{q}_2$$
$$\mathbf{p}_3 \rightarrow \mathbf{q}_3$$

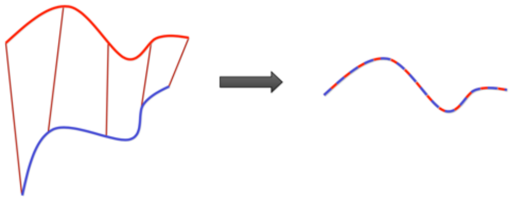
$$R\mathbf{p}_i + t \approx \mathbf{q}_i$$



1.2. Registration via ICP: Iterative Closest Point

§1. The geometry processing pipeline

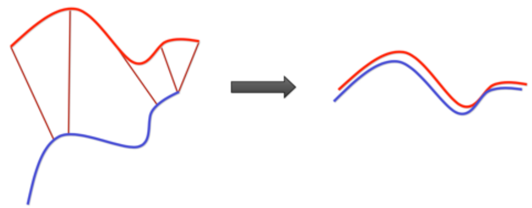
- Idea: Iteratively (1) find correspondences and (2) use them to find a transformation
- Intuition: If you have the right correspondences, then the problem is easy



1.2. Registration via ICP: Iterative Closest Point

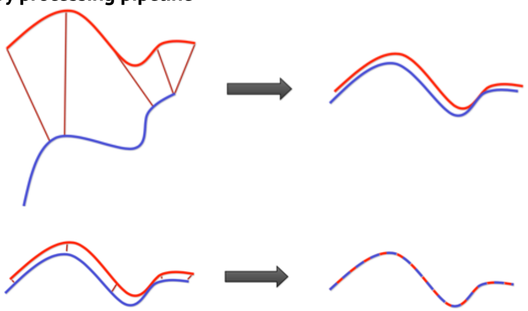
§1. The geometry processing pipeline

- Idea: Iteratively (1) find correspondences and (2) use them to find a transformation
- Intuition: If you don't have the right correspondences, you still can make progress



1.2. Registration via ICP: Iterative Closest Point

§1. The geometry processing pipeline



This algorithm converges to the correct solution only
if the starting scans are "close enough"

1.2. Registration via ICP: basic algorithm

§1. The geometry processing pipeline

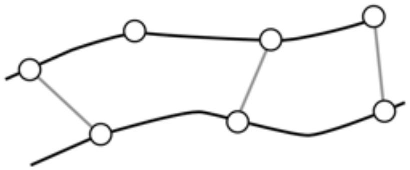
- Select (e.g., 1000) random points
- Match each to closest point on other scan, using data structure such as *k*-d tree
- Reject pairs with distance > *k* times median
- Construct error function:

$$E := \sum_i (R\mathbf{p}_i + t - \mathbf{q}_i)^2$$

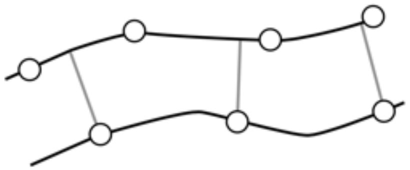
- Minimize (closed form solution
  comparison of four major algorithms", http://dl.acm.org/citation.cfm?id=250160)

1.2. Registration via ICP: important variant

§1. The geometry processing pipeline



Point-to-Point

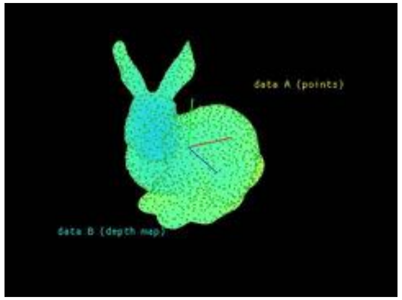


Point-to-Plane

See http://resources.mpi-inf.mpg.de/deformableShapeMatching/EG2012_Tutorial/ for details

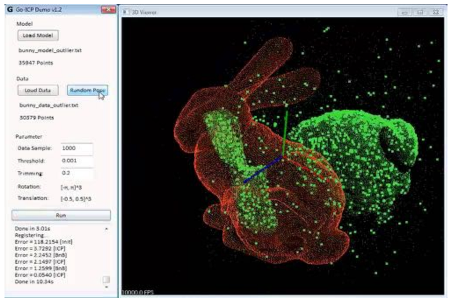
1.2. Registration via ICP: example impl.

§1. The geometry processing pipeline



1.2. Registration via ICP: example impl.

§1. The geometry processing pipeline



1.2. Registration via ICP: Related Work

§1. The geometry processing pipeline

- Original ICP:
  http://graphics.stanford.edu/courses/cs164-10-spring/Handouts/paper_icp.pdf
- Commonly used improvement:
  http://www8.cs.umu.se/research/ifor/dl/fasticp_paper.pdf
- Global registration, one of initial reliable methods:
  http://vecg.cs.ucl.ac.uk/Projects/SmartGeometry/global_registration/paper_docs/global_registration_sgp_05.pdf
- Recent paper, with a few refs:
  http://vladlen.info/publications/fast-global-registration/ (Talk by V. Koltun: http://videolectures.net/eccv2016_koltun_global_registration/)

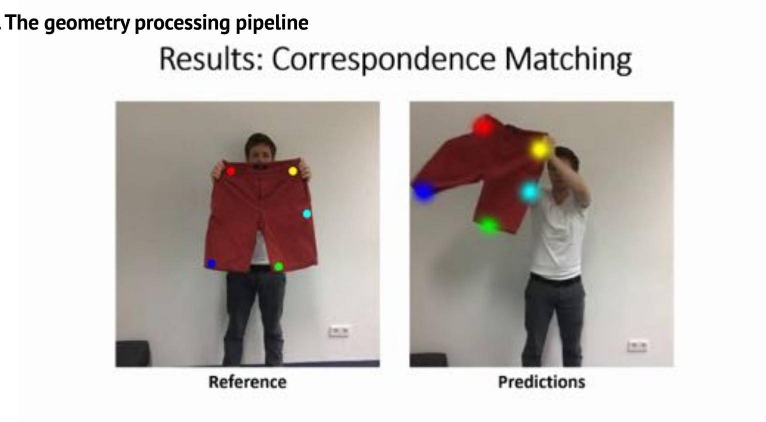
1.2. Registration via ICP: Problems

§1. The geometry processing pipeline

- In reality, registration needs to be non-rigid: e.g. range scans are usually warped
  - matters only for high quality
  - no direct ground truth may be available
  - data: for a set of objects, a collections of warped scans for each
  - learn an alignment/dewarping transformation
- Extension of ICP: http://gfx.cs.princeton.edu/pubs/Brown_2007_GNA/global_tps.pdf
- Related, more difficult (in general form) problem that received a lot of attention: **non-rigid reconstruction**

1.2. Registration: Non-Rigid

§1. The geometry processing pipeline



Reconstruction

1.5. Reconstruction: Digital Michelangelo Project

§1. The geometry processing pipeline

1G sample points → 8M triangles

4G sample points → 8M triangles

1.5. Reconstruction: Problem statement

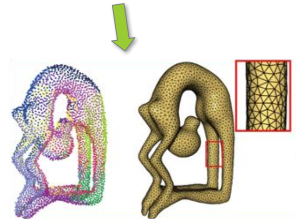
§1. The geometry processing pipeline

- Given **partial information of an unknown surface,** construct, to the extent possible, a **compact representation of the surface** (Hoppe et al., 1992)
  - Commonly (in this course): use multiple viewpoints and range data
- Surface: compact, connected, orientable 2D manifold, possibly with boundary, embedded in $\mathbb{R}^3$
  - *Closed surface:* a surface without a boundary, *bordered surface:* non-empty boundary
  - *Simplicial surface:* piecewise linear surface with triangular faces
- **Goal:** given samples $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ on or near an unknown surface M, recover $M' \approx M$

1.5. Reconstruction

§1. The geometry processing pipeline

Scanning: results in range images ⇨ **Registration:** bring all range images to one coordinate system ⇨ **Stitching/ reconstruction:** Integration of scans into a single mesh ⇨ **Postprocess:**
- Topological and geometric filtering
- Remeshing
- Compression



1.5. Reconstruction: Input to Process

§1. The geometry processing pipeline

- Input option 1:
  just a set of 3D points, irregularly spaced
  - Need to estimate normals

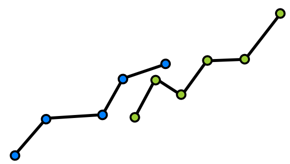
- Input option 2:
  normals come from the range scans

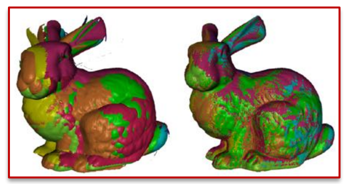
set of raw scans → reconstructed model

1.5. Reconstruction: How to Connect the Dots?

§1. The geometry processing pipeline

- **Explicit reconstruction:**
  stitch the range scans together



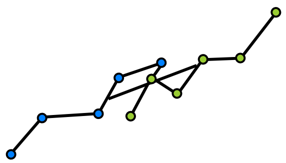


"Zippered Polygon Meshes from Range Images", Greg Turk and Marc Levoy, ACM SIGGRAPH 1994

1.5. Reconstruction: How to Connect the Dots?

§1. The geometry processing pipeline

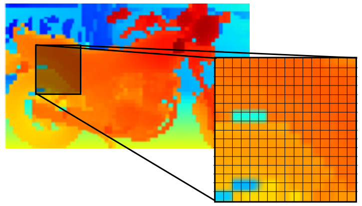
- **Explicit reconstruction:**
  stitch the range scans together



- Connect sample points by triangles
- Exact interpolation of sample points
- Bad for noisy or misaligned data
- Can lead to holes or non-manifold situations

1.5. Reconstruction: Range-Image to Mesh

§1. The geometry processing pipeline



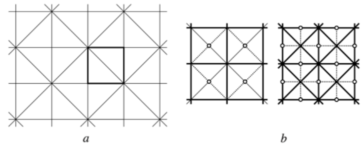


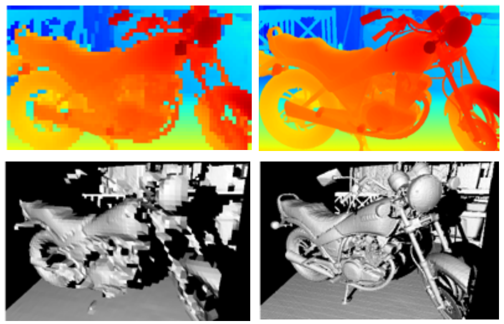
Fig. 1. a. Laves [4.8²] tiling with one of the basic blocks outlined. b. Two bisection refinement steps are equivalent to a face split. Vertices inserted at each step are shown as circles, new edges are shown as dotted lines.

Velho, Luiz, and Denis Zorin. "4–8 Subdivision." *Computer Aided Geometric Design* 18.5 (2001): 397-427.

1.5. Reconstruction: Range-Image to Mesh

§1. The geometry processing pipeline

Low resolution High resolution



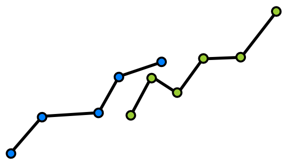
Range-images

Explicit reconstruction

1.5. Reconstruction: How to Connect the Dots?

§1. The geometry processing pipeline

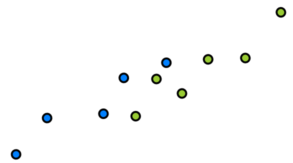
- **Implicit reconstruction:** estimate a signed distance function (SDF); extract 0-level set



1.5. Reconstruction: How to Connect the Dots?

§1. The geometry processing pipeline

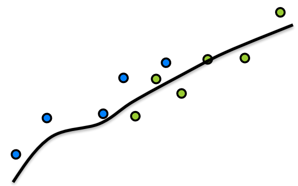
- **Implicit reconstruction:** estimate a signed distance function (SDF); extract 0-level set



1.5. Reconstruction: How to Connect the Dots?

§1. The geometry processing pipeline

- **Implicit reconstruction:** estimate a signed distance function (SDF); extract 0-level set

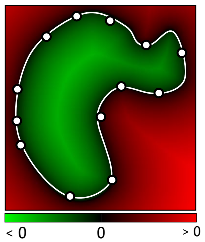


- Approximation of input points
- Watertight manifold results by construction

1.5. Reconstruction: How to Connect the Dots?

§1. The geometry processing pipeline

- **Implicit reconstruction:** estimate a **signed distance function (SDF)**; extract 0-level set



- Assumes the existence of a function

$$f : \mathbb{R}^3 \rightarrow \mathbb{R}$$

with value > 0 outside the shape and < 0 inside

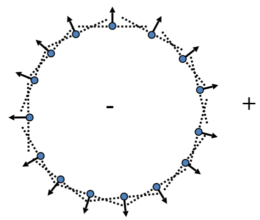
- Extract zero-level set

$$\{\mathbf{x} : f(\mathbf{x}) = 0\}$$

1.5. Reconstruction: SDF from Points and Normals

§1. The geometry processing pipeline

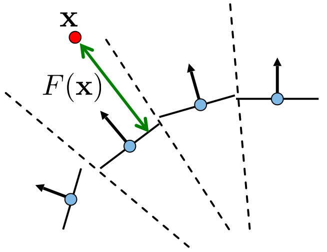
- Compute **signed distance function (SDF)** to the tangent plane of the closest point
- Normals help to distinguish between inside and outside
- "Surface reconstruction from unorganized points", Hoppe et al., ACM SIGGRAPH 1992 http://research.microsoft.com/en-us/um/people/hoppe/proj/recon/



1.5. Reconstruction: SDF from Points and Normals

§1. The geometry processing pipeline

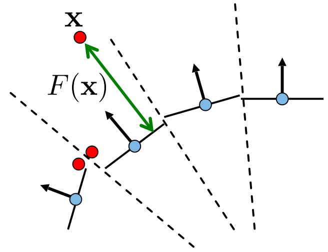
- Compute **signed distance function (SDF)** to the tangent plane of the closest point
- Problem??



1.5. Reconstruction: SDF from Points and Normals

§1. The geometry processing pipeline

- Compute **signed distance function (SDF)** to the tangent plane* of the closest point
- The function will be discontinuous

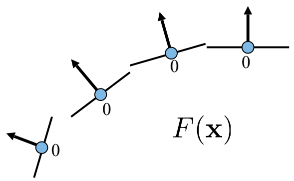


* The Hoppe92 paper computes the tangent planes slightly differently (by PCA on k-nearest-neighbors of each data point, see next class), but the consequences are still the same.

1.5. Reconstruction: Smooth SDF

§1. The geometry processing pipeline

- Instead find a smooth formulation for F.
- Scattered data interpolation:
  - $F(\mathbf{p}_i) = 0$
  - F is smooth
  - Avoid trivial $F \equiv 0$

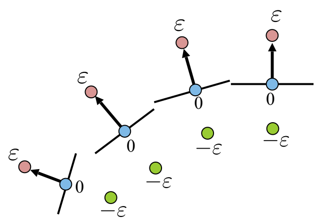


"Reconstruction and representation of 3D objects with radial basis functions", Carr et al., ACM SIGGRAPH 2001

1.5. Reconstruction: Smooth SDF

§1. The geometry processing pipeline

- Scattered data interpolation:
  - $F(\mathbf{p}_i) = 0$
  - F is smooth
  - Avoid trivial $F \equiv 0$
- Add off-surface constraints



$$F(\mathbf{p}_i + \varepsilon \mathbf{n}_i) = \varepsilon$$
$$F(\mathbf{p}_i - \varepsilon \mathbf{n}_i) = -\varepsilon$$

"Reconstruction and representation of 3D objects with radial basis functions", Carr et al., ACM SIGGRAPH 2001

1.5. Reconstruction: Radial Basis Function Interpolation

§1. The geometry processing pipeline

Spline $\varphi(r) = r^2 \log r$

Gaussian $\varphi(r) = \exp\{-cr^2\}$

- **RBF**: Weighted sum of shifted, smooth kernels

$$F(\mathbf{x}) = \sum_{i=0}^{N-1} w_i \, \varphi(\|\mathbf{x} - \mathbf{c}_i\|)$$

Scalar weights
Unknowns

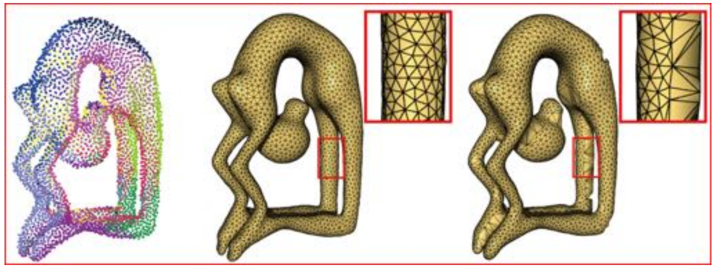
Smooth kernels (basis functions) centered at constrained points.
For example:
$\varphi(r) = r^3$

$N = 3n$

Skoltech
Skolkovo Institute of Science and Technology

1.5. Reconstruction: Implicit vs. Explicit

§1. The geometry processing pipeline



Input Implicit Explicit

Skoltech
Skolkovo Institute of Science and Technology

1.5. Reconstruction: Summary

§1. The geometry processing pipeline

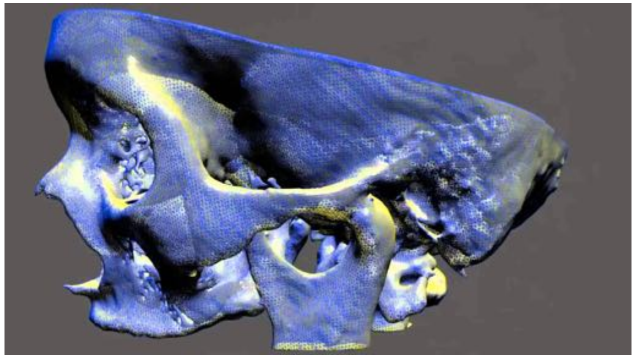
- **Surface reconstruction:** create a surface representation from sparse input points
  - **Explicit:** directly create connectivity by linking close points together
  - **Implicit:** recover a signed distance function (SDF) with values < 0 inside the shape and > 0 outside, then extract level set (next section)
  - State-of-the-art reconstruction algorithm: Poisson Surface Reconstruction (more in ~Lecture 6)

Skoltech
Skolkovo Institute of Science and Technology

Meshing

1.5. Meshing: Motivation

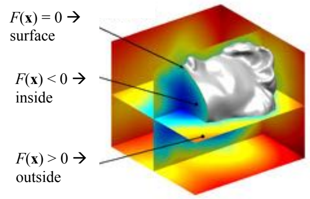
§1. The geometry processing pipeline



1.5. Meshing: Extracting the Surface

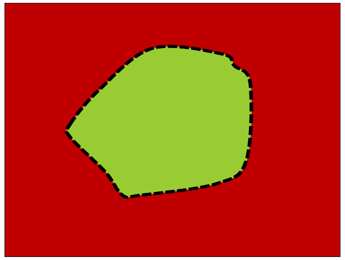
§1. The geometry processing pipeline

- Wish to compute a manifold mesh of the level set
- **Mesh:** a subdivision of a continuous geometric space into discrete geometric and topological cells (often simplicial surface is constructed)
- **Meshing:** implicit surface → simplicial surface



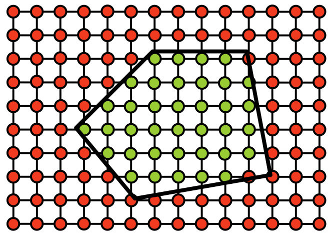
1.5. Meshing: Sample the SDF

§1. The geometry processing pipeline



1.5. Meshing: Sample the SDF

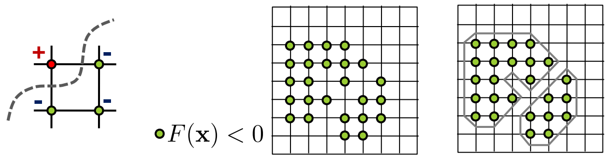
§1. The geometry processing pipeline



1.5. Meshing: Tessellation

§1. The geometry processing pipeline

- Want to approximate an implicit surface with a mesh
- Can't explicitly compute all the roots
  - Sampling the level set is difficult (root finding)
- Solution: find approximate roots by trapping the implicit surface in a grid (lattice)

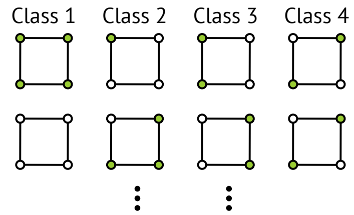


Skoltech
Skolkovo Institute of Science and Technology

1.5. Meshing: Marching Squares

§1. The geometry processing pipeline

- 16 different configurations in 2D
- 4 equivalence classes (up to rotational and reflection symmetry + complement)

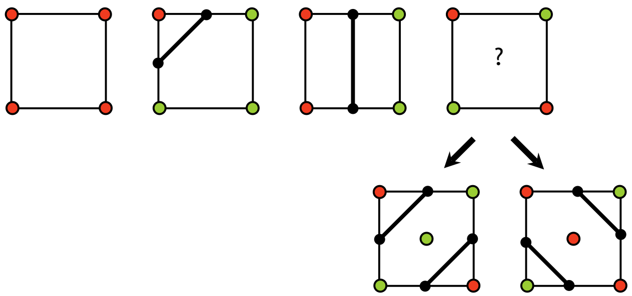


Skoltech
Skolkovo Institute of Science and Technology

1.5. Meshing: Tessellation in 2D

§1. The geometry processing pipeline

- 4 equivalence classes (up to rotational and reflection symmetry + complement)

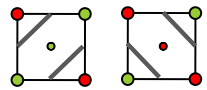


Skoltech
Skolkovo Institute of Science and Technology

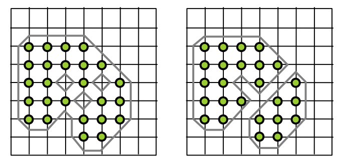
1.5. Meshing: Tessellation in 2D

§1. The geometry processing pipeline

- Case 4 is ambiguous:



- Always pick consistently to avoid problems with the resulting mesh

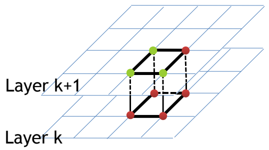


1.5. Meshing: 3D Marching Cubes

Skoltech
Skolkovo Institute of Science and Technology

§1. The geometry processing pipeline

- Marching Cubes (Lorensen and Cline 1987)
  1. Load 4 layers of the grid into memory
  2. Create a cube whose vertices lie on the two middle layers
  3. Classify the vertices of the cube according to the implicit function (inside, outside or on the surface)

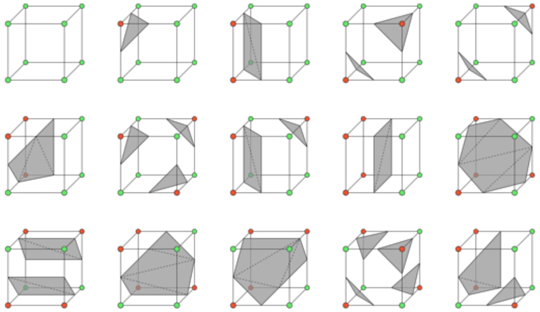


1.5. Meshing: 3D Marching Cubes

Skoltech
Skolkovo Institute of Science and Technology

§1. The geometry processing pipeline

- Unique cases (by rotation, reflection and complement)

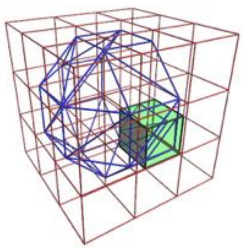


1.5. Meshing: 3D Marching Cubes

Skoltech
Skolkovo Institute of Science and Technology

§1. The geometry processing pipeline

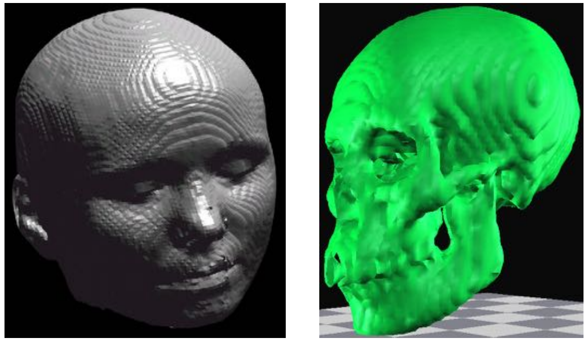
Implementation



1.5. Meshing: Marching Cubes – Problems

Skoltech
Skolkovo Institute of Science and Technology

§1. The geometry processing pipeline



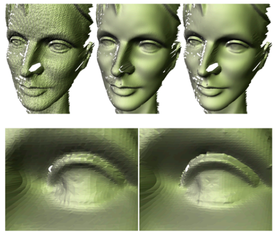
Output aliasing artefacts

Postprocessing

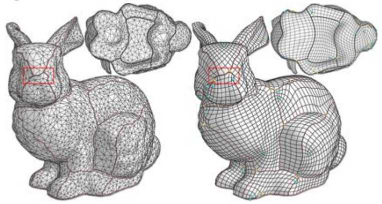
1.6. Postprocessing

§1. The geometry processing pipeline

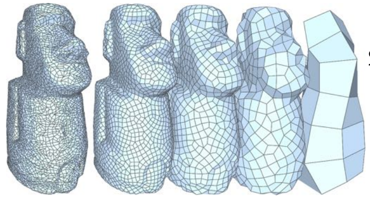
- Highly application-dependent



Mesh smoothing/de-noising



Re-meshing/quadrangulation



Simplification, compression

§2. 3D representations in vision and graphics [Friday Tutorial]

References

1. Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., & Lévy, B. (2010). *Polygon mesh processing*. CRC press.