01 - Introduction and Overview

Acknowledgements: Olga Sorkine-Hornung
Who Am I?

- Assistant Professor of Computer Science since February 2016
Graphics @ Courant

Denis Zorin
Fluid and Solid Simulation
Microstructures
Scientific Computing

Kenneth Perlin
Virtual Reality
Computer/Human Interface
Animation

Daniele Panozzo
Geometry Processing
Digital Fabrication
Architectural Geometry
Course Goals

• Learn how to design, program and analyze algorithms for **interactive 3D shape modeling** and **digital geometry processing**

• Theory and applications of 3D mesh processing

• Hands-on experience with shape modeling and geometry processing algorithms
Geometric Modeling and Processing

• The shape of an object is an important characteristic (not the only one…)

• Geometry processing: computerized modeling of 2D/3D geometry
Applications

- Product design and prototyping
- Medicine, prosthetics
- Architecture
- Cultural heritage
Applications

Geographical systems

Manufacturing, 3D Printing

[Bickel et al., ACM SIGGRAPH 2010]
Fabrication

- Modern scanning and 3D printing technologies allow replication and much more
Digital Geometry Processing (DGP)

- Processing of discrete (polygonal mesh) models
- Why discrete?
  - Simplicity – ease of description
  - Efficiently rendered by graphics hardware
  - Output of most acquisition tools (CT, MRI, LIDAR, Kinect…)
  - Input to most simulation/analysis tools (FE solvers)
Interactive Shape Modeling

• Tools for design, editing and animation of digital shapes
  • Interactive means fast algorithms
  • Intuitive – convenient interface and predictable outcome

http://youtu.be/EMx6yNe23ug
Digital Shape Modeling

- How do shapes find their way into computers?
- Geometric modeling is difficult

Humans have no direct “video out”

“Translation” from 2D to 3D is hard
Digital Shape Modeling

• How do shapes find their way into computers?
• Geometric modeling is difficult

Use computation to compensate for lack of direct ability to convey visual information
Computer-Aided Geometric Design

- Traditional pipeline for modeling shapes from scratch

User defines a layout of surface patches and control points
Computer-Aided Geometric Design

• Traditional pipeline for modeling shapes from scratch

User defines a layout of surface patches and control points
Computer-Aided Geometric Design

• Traditional pipeline for modeling shapes from scratch

User defines a layout of surface patches and control points

Editing is performed by moving control points and/or prescribing tangents
Computer-Aided Geometric Design

- Traditional pipeline for modeling shapes from scratch

User defines a layout of surface patches and control points

Editing is performed by moving control points and/or prescribing tangents
Computer-Aided Geometric Design

- Traditional pipeline for modeling shapes from scratch

User defines a layout of surface patches and control points

Editing is performed by moving control points and/or prescribing tangents

Patch-based construction of a surface
Blender Demo
Computer-Aided Geometric Design

- High-quality surfaces
- Constrained modeling
- Requires a specific idea of the object first
  - Not easy to experiment and explore alternatives
- Requires training, skill and tedious work

CATIA, Dassault Systemes
http://youtu.be/gTC5zMktMr0
Modern Geometry Acquisition Pipeline

Physical object → scanning → 3D point samples, range images

Digital, discrete shape representation: unstructured mesh
Unstructured Digital Shapes

• How to **edit** and **animate**?

• How to convert to a **structured representation**?

• Computational challenge: very large amounts of data, yet modeling has to remain interactive

Thai statue, 10M triangles, Stanford 3D Scanning Repository
Traditional CAD vs Modern Mesh Modeling

**Traditional CAD**

- Control Point
- Control Polygon

\[ x(u, v) = \sum_{i, j} p_{i,j} B_i(u) B_j(v) \]

**Modern mesh modeling**

\[ \min_x \int_S E(x) \quad s.t. \quad x|_{c} = x_{fixed} \]

User has more freedom!
Select and manipulate arbitrary regions.
Tools?

• Use techniques from both CS & Math
  • PDEs
  • Discrete differential geometry
  • Numerical linear algebra
  • Graph theory
  • ...

• ...combined with intuition and creativity ...

• work on real data = write/use code
Prerequisites

- Linear Algebra
  - We will not cover the concepts that you need. If you are not familiar with basis, points, vectors, matrices and linear systems, the course will be difficult to follow.

- C++
  - We will not review the basic concepts of C++. There will be plenty of examples given, it should be easy to catch up if you never used C++ before. Keep this reference at hand [http://www.cppreference.com](http://www.cppreference.com)

- Why C++?

- Git
  - It will be used to distribute material and to deliver homework
Organization

• Communication through the course repository/website: https://github.com/danielepanozzo/gp

• Mailing list (if you did not receive the test email, let me know)

• Weekly lecture: Thursdays 5.10-7 PM - 60 Fifth Ave C10

• Office hours:
  • Daniele: Thursday 3PM-4PM - 60 Fifth Ave - 5th floor
  • Zhongshi: TBD
Lectures

• I will upload the slides on the website before the class, so that you can directly annotate them

• For every class, I will always add references in the end to the textbook and/or external resources

• You are encourage to take a look at the material before I present it in class
Lectures

- Please interrupt me at any time to ask questions
Final Coding Project

• Individual project, we will publish the rules later but you are essentially free to do whatever you want, as long as it requires geometry processing

• The project will be presented in a fast-forward session at the end of semester (3-5 minutes per project)
Material

Polygon Mesh Processing

https://libigl.github.io/libigl/tutorial/tutorial.html

https://www.wikipedia.org
Grading

<table>
<thead>
<tr>
<th>Assignment</th>
<th>Topic</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Implicit Surfaces</td>
<td>17.5</td>
</tr>
<tr>
<td>2</td>
<td>Parametrization</td>
<td>17.5</td>
</tr>
<tr>
<td>3</td>
<td>Deformation</td>
<td>17.5</td>
</tr>
<tr>
<td>4</td>
<td>Final Project</td>
<td>27.5</td>
</tr>
</tbody>
</table>

Total Exercises: 80%

- **Final Oral Exam: 20%** (you must get at least 10% in the final to pass)

- You **must** pass the final to pass the class

- The oral exam will be recorded (if you don’t want to be recorded send me a private email before the end of the week)

- There will be optional tasks, that will allow you to recover points lost in the assignments
Policy

• You are encouraged to consult with your classmates/friends but collaboration in the assignments is **NOT** allowed.

• You are **NOT** allowed to copy code online or use external libraries (except those provided in the class) for the first 3 assignments.

• We will use plagiarism tools to validate all homework. Plagiarism will be punished with a zero-tolerance policy: the minimal penalty will be a score of 0 points on the assignment, the reduction of 1 letter grade in the final score, and a permanent ban on NYU-related jobs.

• If you are stuck in an assignment post on github or send an email to me or to the assistant.

• Ask questions on github, we will usually answer in less than 24 hours.
Course Topics

- Overview of shape representations
  - Parametric curves/surfaces
  - Implicits
  - Polygonal meshes

Pixar
Course Topics

• Shape acquisition
  • Scanning/imaging
  • Reconstruction
Course Topics

- Differential geometry
- Continuous and (mostly) discrete
- Powerful tool to analyze and model shapes

© 2002 Encyclopedia Britannica, Inc.
Course Topics

- Digital geometry processing
  - Denoising, smoothing, simplification, remeshing, parameterization, compression
Course Topics

• Parameterization

3D space \((x,y,z)\)

2D parameter domain \((u,v)\)

boundary

boundary
Course Topics

- Parameterization
Course Topics

- Shape creation and editing

http://youtu.be/W0XGkS7zebo
Course Topics

- Skinning, animation

http://youtu.be/P9fqm8vgdB8

http://youtu.be/Pjg33pH9Rk0
Course Topics

• Architectural geometry and structure-aware modeling

Panozzo et al., SIGGRAPH 2013
Course Topics

- Architectural geometry and structure-aware modeling

[Potmann 08]
http://www.dmg.tuwien.ac.at/pottmann/2008/panels08/panels.html

[Whiting 09]
http://www.inf.ethz.ch/personal/whitinge/projects/siggasia09.html

earthquake simulation
Course Topics

• 3D fabrication-aware shape modeling
Course Topics

• 3D fabrication-aware shape modeling
Break

- What do you prefer?
- Let me know after this 10 minutes break!
Shape Representations
Shape Representation: Origin- and Application-Dependent

- Acquired real-world objects:
  - Discrete sampling
  - Points, meshes

- Modeling “by hand”:
  - Higher-level representations, amendable to modification, control
  - Parametric surfaces, subdivision surfaces, implicits

- Procedural modeling
  - Algorithms, grammars
Similar to the 2D Image Domain

- Acquired digital images:
  - Discrete sampling
  - Pixels on a grid

- Painting “by hand”:
  - Strokes + color/shading
  - Vector graphics
  - Controls for editing
Similar to the 2D Image Domain

- Acquired digital images:
  - Discrete sampling
  - Pixels on a grid

- Painting “by hand”:
  - Strokes + color/shading
  - Vector graphics
  - Controls for editing
Representation Considerations

• How should we represent geometry?
  • Needs to be stored in the computer
  • Creation of new shapes
    • Input metaphors, interfaces…
  • What operations do we apply?
    • Editing, simplification, smoothing, filtering, repair…
• How to render it?
  • Rasterization, raytracing…

Variational Shape Approximation
Shape Representations

- Points
- Polygonal meshes
Shape Representations

- Parametric surfaces
- Subdivision surfaces
- Implicit functions
Points
Output of Acquisition
Points

- Standard 3D data from a variety of sources
  - Often results from scanners
  - Potentially noisy

- Depth imaging
- Registration of multiple images
Points

- points = unordered set of 3-tuples
- Often converted to other reps
  - Meshes, implicits, parametric surfaces
  - Easier to process, edit and/or render
- Efficient point processing and modeling requires a spatial partitioning data structure
  - To figure out neighborhoods
Points: Neighborhood information

• Why do we need neighbors?

• Need sub-linear-time implementations of
  • k-nearest neighbors to point $x$
  • In-radius search $\| \mathbf{p}_i - \mathbf{x} \| < \varepsilon$

need normals (for shading)

upsampling - need to count density
Parametric Curves and Surfaces
Parametric Representation

- Range of a function
  \[ f : X \to Y, X \subseteq \mathbb{R}^m, Y \subseteq \mathbb{R}^n \]

- Planar curve: \( m = 1, n = 2 \)
  \[ s(t) = (x(t), y(t)) \]

- Space curve: \( m = 1, n = 3 \)
  \[ s(t) = (x(t), y(t), z(t)) \]
Parametric Representation

\[ f : X \rightarrow Y, X \subseteq \mathbb{R}^m, Y \subseteq \mathbb{R}^n \]

- Range of a function

- Surface in 3D:

\[ s(u, v) = (x(u, v), y(u, v), z(u, v)) \]
Parametric Curves

• Explicit curve/circle in 2D

\[ p : \mathbb{R} \rightarrow \mathbb{R}^2 \]
\[ t \mapsto p(t) = (x(t), y(t)) \]
\[ p(t) = r \left( \cos(t), \sin(t) \right) \]
\[ t \in [0, 2\pi) \]
Parametric Curves

- Bezier curves, splines

\[ s(t) = \sum_{i=0}^{n} p_i B_i^n(t) \]

\[ B_i^n(t) = \binom{n}{i} t^i (1 - t)^{n-i} \]
Parametric Surfaces

• Sphere in 3D

\[ s : \mathbb{R}^2 \rightarrow \mathbb{R}^3 \]

\[ s(u, v) = r \left( \cos(u) \cos(v), \sin(u) \cos(v), \sin(v) \right) \]

\[ (u, v) \in [0, 2\pi) \times [-\pi/2, \pi/2] \]
Parametric Surfaces

• Curve swept by another curve

\[ s(u, v) = \sum_{i,j} p_{i,j} B_i(u) B_j(v) \]

• Bezier surface:

\[ s(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} p_{i,j} B_i^m(u) B_j^n(v) \]
Tangents and Normal

\[ s_u = \frac{\partial s(u, v)}{\partial u} \]
\[ s_v = \frac{\partial s(u, v)}{\partial v} \]
\[ \mathbf{n} = \frac{s_u \times s_v}{\| s_u \times s_v \|} \]

Tangent plane is normal to \( \mathbf{n} \)
Parametric Curves and Surfaces

• Advantages
  • Easy to generate points on the curve/surface
  • Separates x/y/z components

• Disadvantages
  • Hard to determine inside/outside
  • Hard to determine if a point is on the curve/surface
Implicit Curves and Surfaces
Implicit Curves and Surfaces

- Kernel of a scalar function
  \[ f : \mathbb{R}^m \rightarrow \mathbb{R} \]
- Curve in 2D:
  \[ S = \{ x \in \mathbb{R}^2 | f(x) = 0 \} \]
- Surface in 3D:
  \[ S = \{ x \in \mathbb{R}^3 | f(x) = 0 \} \]

- Space partitioning
  \[ \{ x \in \mathbb{R}^m | f(x) > 0 \} \text{ Outside} \]
  \[ \{ x \in \mathbb{R}^m | f(x) = 0 \} \text{ Curve/Surface} \]
  \[ \{ x \in \mathbb{R}^m | f(x) < 0 \} \text{ Inside} \]
Implicit Curves and Surfaces

- Kernel of a scalar function
  \[ f : \mathbb{R}^m \rightarrow \mathbb{R} \]
  \[ S = \{ x \in \mathbb{R}^2 | f(x) = 0 \} \]
  \[ S = \{ x \in \mathbb{R}^3 | f(x) = 0 \} \]

- Curve in 2D:
- Surface in 3D:

- Zero level set of signed distance function
Implicit Curves and Surfaces

• Implicit circle and sphere

\[ f(x, y) = x^2 + y^2 - r^2 \quad f(x, y, z) = x^2 + y^2 + z^2 - r^2 \]
Implicit Curves and Surfaces

• The normal direction to the surface (curve) is given by the gradient of the implicit function

\[ \nabla f(x, y, z) = \left( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \right)^T \]

• Example

\[ f(x, y, z) = x^2 + y^2 + z^2 - r^2 \]
\[ \nabla f(x, y, z) = (2x, 2y, 2z)^T \]
Boolean Set Operations

- **Union:**
  \[ \bigcup_{i} f_i(x) = \min f_i(x) \]

- **Intersection:**
  \[ \bigcap_{i} f_i(x) = \max f_i(x) \]
Boolean Set Operations

- Positive = outside, negative = inside
- Boolean subtraction:
  \[ h = \max(f, -g) \]

<table>
<thead>
<tr>
<th></th>
<th>( f &gt; 0 )</th>
<th>( f &lt; 0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( g &gt; 0 )</td>
<td>( h &gt; 0 )</td>
<td>( h &lt; 0 )</td>
</tr>
<tr>
<td>( g &lt; 0 )</td>
<td>( h &gt; 0 )</td>
<td>( h &gt; 0 )</td>
</tr>
</tbody>
</table>

- Much easier than for parametric surfaces!
Smooth Set Operations

• In many cases, smooth blending is desired
  • Pasko and Savchenko, Blending operations for the functionally based constructive geometry [1994]

\[
f \cup g = \frac{1}{1+\alpha} \left( f + g - \sqrt{f^2 + g^2 - 2\alpha fg} \right)
\]
\[
f \cap g = \frac{1}{1+\alpha} \left( f + g + \sqrt{f^2 + g^2 - 2\alpha fg} \right)
\]
Smooth Set Operations

- Examples

\[ \alpha = 0 \quad \alpha = 1 \]
Designing with Implicit Surfaces

• Zero set (or level set) of a function:

\[ f(\mathbf{p}) = ||\mathbf{p}||^2 - r^2 \]

• But also a level set at value \( e^{-1} \) of this function:

\[ f(\mathbf{p}) = e^{-||\mathbf{p}||^2/r^2} \text{ at } e^{-1} \]
Designing with Implicit Surfaces

• With smooth falloff functions, adding implicit functions generates a blend:

\[ f(p) = e^{-\|p-p_1\|^2} + e^{-\|p-p_2\|^2} \]

• Called “Metaballs” or “Blobs”
Blobs

- Suggested by Blinn [1982]
  - Defined implicitly by a potential function around a point $\mathbf{p}_i$:
    \[
    f(\mathbf{p}) = a_i e^{-b_i \lVert \mathbf{p} - \mathbf{p}_i \rVert^2}
    \]
- Set operations by simple addition/subtraction

Blobs


Sketch-Based Implicit Blending

Baptiste Angles¹,², Marco Tarini³, Brian Wyvill¹, Loïc Barthe², Andrea Tagliasacchi¹

1. University of Victoria  2. Université de Toulouse, IRIT/CNRS  3. Università dell’Insubria, ISTI / CNR
Implicit Curves and Surfaces

- Advantages
  - Easy to determine inside/outside
  - Easy to determine if a point is **on** the curve/surface

- Disadvantages
  - Hard to generate points on the curve/surface
  - Does not lend itself to (real-time) rendering
### Summary

<table>
<thead>
<tr>
<th>Parametric</th>
<th>Implicit</th>
<th>Discrete/Sampled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Splines, tensor-product surfaces</td>
<td>Metaballs/blobs</td>
<td>Meshes</td>
</tr>
<tr>
<td>Subdivision surfaces</td>
<td>Distance fields</td>
<td>Point set surfaces</td>
</tr>
</tbody>
</table>

**Parametric**
- Splines, tensor-product surfaces
- Subdivision surfaces

**Implicit**
- Metaballs/blobs
- Distance fields

**Discrete/Sampled**
- Meshes
- Point set surfaces
Next Week

• A bit about geometry acquisition

• All About Meshes
In the Next Lectures

• How to get a nice, watertight surface mesh from a sampled point set

• The most popular way:
  points ➔ implicit function ➔ surface mesh
Thank you