Geometric Modeling 2
More Geometric Modeling Techniques

• Why not just use a triangulated surface and spline modeling for everything?
• We’ll address this today...
Implicit surfaces
Implicit Surfaces

- Implicit surfaces represent a surface with a function $\phi(x)$ defined over the entire 3D space
- The inside region $\Omega^-$, the outside region is $\Omega^+$, and the surface $\partial \Omega$ are all defined by the function $\phi(x)$
  - $\phi(x) < 0$ inside
  - $\phi(x) > 0$ outside
  - $\phi(x) = 0$ surface
- Easy to check if a point is inside an object
- Efficient to make topology changes to an object
- Efficient Boolean operations: Union, Difference, Intersection

\[ \phi = x^2 + y^2 - 1 = 0 \]
Implicit Surfaces

• Implicit surfaces are good for handling complicated surfaces like water

• Triangle mesh representations struggle with merging and pinching, overturning waves, etc.
Topological Changes
Constructive Solid Geometry (CSG)

- Create complex objects using boolean operators on simple objects

- CSG objects can be represented by binary trees
  - $\cup$ for Union
  - $\setminus$ for Difference
  - $\cap$ for Intersection

- CSG operations are readily applied to implicit surfaces

http://en.wikipedia.org/wiki/Constructive_solid_geometry
Blobbies, Metaballs & Soft Objects

- Surfaces created from straight lines and planes can appear very rigid looking
- Sometimes want more organic “squishy” looking objects
- Each blob is defined as a density function around a particle
- To create the aggregate surface:
  - Iterate through each pixel on the screen
  - Iterate through each blob in the world
  - Calculate the blob’s function for the current pixel, and add it to the pixel’s current value
  - Kernels can be 2D ellipses, 2D diamonds, 3D spheres, etc.
Metaballs & Soft Objects

- Around the same time blobbies were created
  - Metaballs developed in Japan
  - Soft objects developed in Canada and New Zealand
- Similar, just slightly different density kernel functions
- Metaballs and soft objects have a finite influence around each particle, so each pixel only needs to query nearby density functions (computationally cheaper)
Blobby Modeling
Blobby Modeling

(a) $N = 1$
(b) $N = 2$
(c) $N = 20$
(d) $N = 60$

(e) $N = 120$
(f) $N = 451$
Topological changes
Question 1 (short form)

Give TWO good reasons that someone might use an implicit surface, instead of a triangle mesh, for geometric modeling.
Question 1 (long form)

Explain the strengths and weaknesses of implicit surfaces versus explicit surfaces like triangles and NURBS. (probably around 250 words)
Marching Cubes

- Turns an implicit surface into triangles
- Define the implicit surface on a 3D grid
- Then for each grid cell determine the topology of the volume in order to reconstruct the surface with triangles
- Often used for medical data...
Computer vision
Range Scanning

- A range scanner senses 3D positions on an object’s surface and returns an \( m \times n \) grid of distances (range points) that describe the surface – \( m \) points per laser sheet, \( n \) laser sheets. This grid is called a range image.
- In case of multiple range images, a rigid transformation is found for each image to align them together.
- This transformation is found using a modified version of the iterative closest point alignment (ICP), which minimizes the least squared distance between nearest points in two range images.
Range Scanning

- Each sample point in the $m \times n$ range image is a potential vertex in the triangle mesh. Special care is taken to avoid inadvertently joining portions of the surface together that are separated by depth discontinuities (Figure 3).
- Zero, one or two triangles can be created from four points of a range image that are in adjacent rows and columns.
- Shortest of the two diagonals between the points is used to identify the two triplets of points that may become triangles.
- Each of these point triples is made into a triangle if the edge lengths fall below a distance threshold.
- These meshes corresponding to each range image are combined using the zippering algorithm\(^1\) given the rigid transformation between each image.

\(^1\) Turk, Greg and Marc Levoy “Zippered Polygon Meshes from Range Images” SIGGRAPH 1994
Mobile 3D Scanning

Structure Sensor for iPad

Autodesk 123D Catch
Voxel Carving

- Construct a voxelized 3D model given multiple images of an object, from calibrated cameras, taken from different directions
- A silhouette is computed for each image
- The silhouette for each image is back projected onto the grid
- The voxels that lie in the back-projection of every image correspond to the final 3D model
- Once the model is acquired, colors can also be back projected

Discretized scene volume, to be assigned RGBA values

Color the voxel gray if silhouette is in every image
Voxel Carving

Original image

Extracted silhouettes

Carved out voxels

Back projecting the colors
Reconstruction from large photo collections

- Construct a 3D model from large number of photos (say from google images).
- Computer vision algorithms can be used to predict the relative camera positions and orientations for each image, and at the same time obtain a sparse point cloud representation of the object. See for example Bundler.
- The position of a point that is visible in multiple images can be determined.
- Many dense reconstruction algorithms can be used to get denser points given the camera parameters and this initial point cloud. See for example CMVS.

See The Visual Turing Test for Scene Reconstruction, Shan et. al. and the references therein.
Trees, Drones, Etc... (Ed Quigley)
Trees, Drones, Etc...  (Ed Quigley)
Trees, Drones, Etc...  (Ed Quigley)
Question 2 (short form)

List TWO of the main drawbacks, when modeling geometry using these computer vision techniques.
Question 2 (long form)

Discuss some of the strengths and weaknesses of modeling geometry using computer vision techniques. (around 250 words)
De-noising

- Given a noisy input mesh (scanned or otherwise), generate a smooth output mesh
- The goal is to filter out the high frequency noise
Laplacian Smoothing

\[ L(p_i) = \frac{1}{2}(p_{i+1} - p_i) + \frac{1}{2}(p_{i-1} - p_i) \]

\[ p_i \gets p_i + \lambda L(p_i), \quad 0 < \lambda < 1 \]

- Repeat several iterations
- For smoothing a surface, take into account all the one-ring incident vertices of \( p_i \) when computing \( L(p_i) \)
Taubin Smoothing

- Laplacian smoothing eventually shrinks a closed curve/surface to a single point.

- Taubin smoothing performs an extra inflation step to counteract shrinkage:
  \[ p_i \leftarrow p_i + \lambda L(p_i), \quad 0 < \lambda < 1 \]
  \[ p_i \leftarrow p_i + \mu L(p_i), \quad \mu < 0 \]

- See more details in “A signal processing approach to fair surface design”
Aside: Adding Noise

- Sometimes, it is desirable to add noise into geometry in order to create some randomness for added realism.
- Many different noise functions: Perlin noise, Worley noise, etc. (more on this when we talk about textures...)

(From: GPUGems3)
Procedural Geometry

• Use an **algorithm** for model creation
• Typically used for complex and tedious models
  • Terrain
  • Plants, foliage
  • Buildings, cities
• Easy to make variations on a model

http://bigquix.deviantart.com/art/Procedural-City-200300796
Procedural Geometry

• Start with a small set of data or rules to describe high level properties of the desired models
  • Tree: branching properties + leaf shape
  • Building: room subdivision + door/window placement

• The rest of the model is algorithmically constructed
  • add randomness to the base model
  • use recursive algorithms

http://procworld.blogspot.com.es/2012/03/building-rooms.html
L-systems

• Typically used to model plants
• Developed by biologist Lindenmayer to study algae growth
• A recursive formal grammar:
  • An alphabet of symbols (terminal and non-terminal)
  • A collection of production rules
    • Non-terminal symbols create new symbols or sequences of symbols recursively
• The process starts with an initial string (axiom) to which the production rules are applied
• Finally, a translator turns both terminals and non-terminals into geometric structures
L-systems

- Nonterminals:
  - A, B: both mean “draw forward”
- Terminals:
  - +: Turn right by 60 degrees
  - -: Turn left by 60 degrees
- Initial Axiom: A
- Rules:
  - $A \rightarrow B + A + B$
  - $B \rightarrow A - B - A$

Sierpinski Triangle
L-system + Stack = Branches

• Nonterminals:
  • \(X\): (no action)  \(F\): draw forward

• Terminals:
  • +: Turn right by 25 degrees
  • −: Turn left by 25 degrees
  • [ : store current state on the stack
  • ] : load state from stack

• Initial Axiom: \(X\)

• Rules:
  • \(X \rightarrow F - [[X]+X] + F [+FX] - X\)
  • \(F \rightarrow FF\)
L-systems

• Easily extended to 3D
  • Model the trunks and branches as cylinders
  • As recursion proceeds:
    • Shrink cylinder size
    • Vary color from brown to green

• Add more variety with a stochastic L-system
  • Multiple rules for each symbol
  • At each symbol, randomly choose one rule to replace

• L-system is a relatively abstract specification
  • Requires experience to model a specific and given form
  • (great place to think about machine learning and data...)

Algorithmic Beauty of Plants by Przemyslaw, Prusinkiewicz, and Lindenmayer http://www.algorithmicbotany.org/papers/#abop

http://web.comhem.se/solgrop/3dtree.htm
Fractals

- Initiator: start with a shape
- Generator: replace subparts with scaled copy of original
- Apply generator repeatedly
Statistical Fractal Generator

- Add randomness in fractal generation
- Can be used to model an irregular “random” silhouette or terrain
- Random midpoint displacement

Start with single horizontal line segment. Repeat for sufficiently large number of times
{
  Repeat over each line segment in scene
  {
    Find midpoint of line segment.
    Displace midpoint in Y by random amount.
    Reduce range for random numbers.
  }
}

2D Silhouette
Generating Height-fields

• Start with a 2D fractal (or any 2D grey-scale image)
• Lay a rectangular grid on the ground
• For each vertex of the grid, vary height based on pixel intensity

http://www.travelnotes.de/rays/grandcan/grandcan.gif
Generating 3D Landscapes

• General procedure
  • Initiator: start with shape
  • Generator: random subparts with a self-similar random pattern

• Use this to generate entire terrains

• Similar to subdivision, but with much more interesting rules for setting vertex positions
Fractal Worlds

http://classes.yale.edu/fractals/panorama/art/mountainssim/romantic/romantic.html
Fractal Worlds

http://www.cruzine.com/2013/04/16/futuristic-artworks-mark-brady/
https://www.behance.net/gallery/4674237/Fractal-Worlds
Question 3 (short form)

What do you think are the PROS and CONS of procedural modeling?
Question 3 (long form)

Investigate and write about some of the PROS and CONS of procedural modeling? (about 250-500 words)
Machine Learning
Machine Learning

Interactive Example-Based Terrain Authoring with Conditional Generative Adversarial Networks, Siggraph Asia 2017

Real Time Live Capture