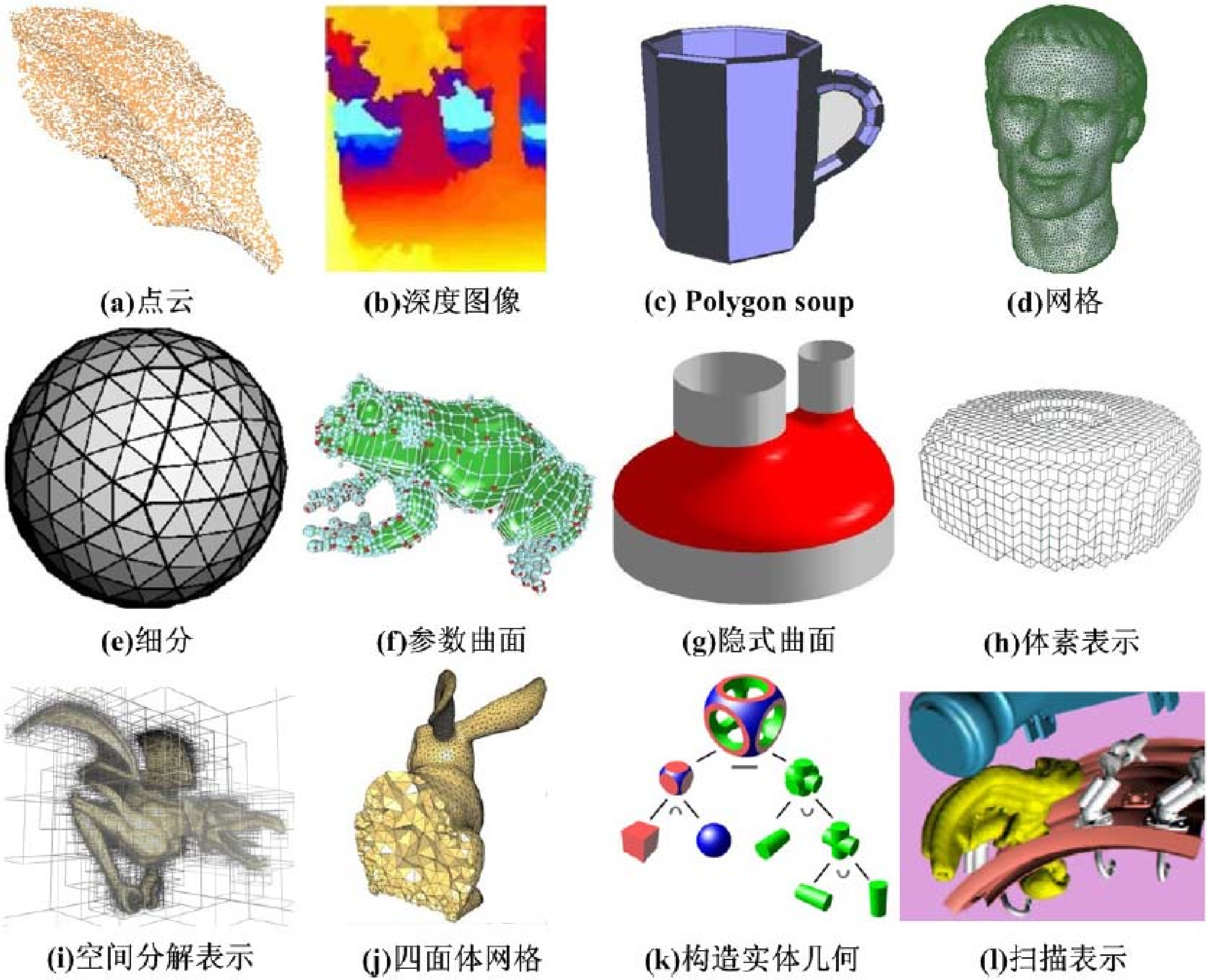
Shape Representation

JJCAO

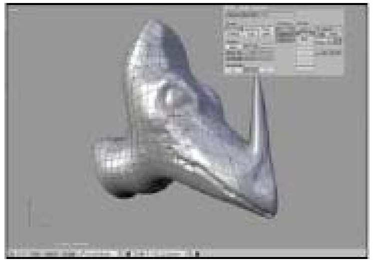
Do not hesitate to do everything that should be done!

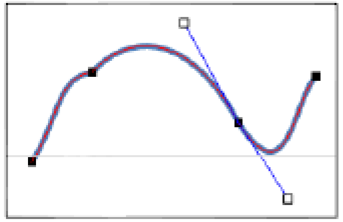
Shape Representation



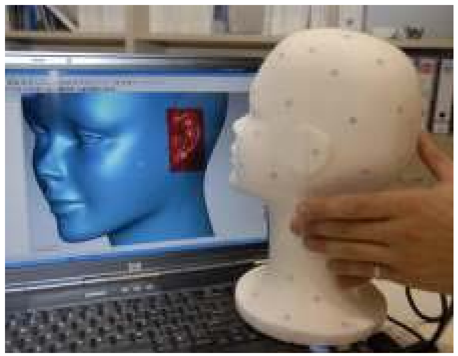
Shape representation

Where does the shape come from?





- Modeling "by hand"
  - Higher-level representations,
  - Amenable to modification, control
- Acquired real-world objects
  - Discrete sampling
  - Points, meshes

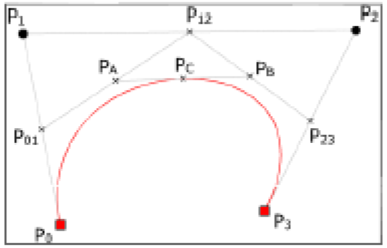


Shape Representation

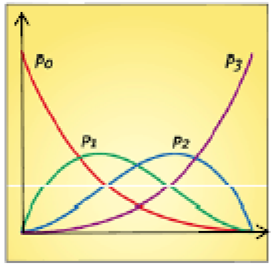
- Parametric surface
- Implicits
- Points
- Polygon mesh

Parametric Curves and Surfaces

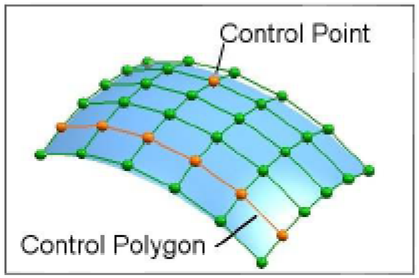
- Curves



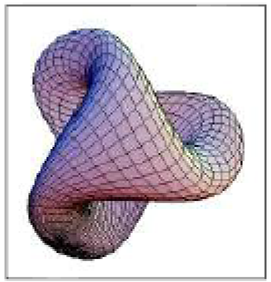
$$S(t) = \mathbf{x}_t$$



- Surfaces



$$S(x, y) = \mathbf{x}_t$$



Surface Representations

- Parametric
  - Represent a surface as (continuous) injective function from a domain $\Omega \subset \boldsymbol{R}^2$ to $S \subset \boldsymbol{R}^3$

$$\mathbf{q} : R^2 \to R^d, d = 1,2,3,\ldots$$

$$(u,v) \mapsto \mathbf{q}(u,v) = \left(x(u,v), y(u,v), z(u,v)\right)$$

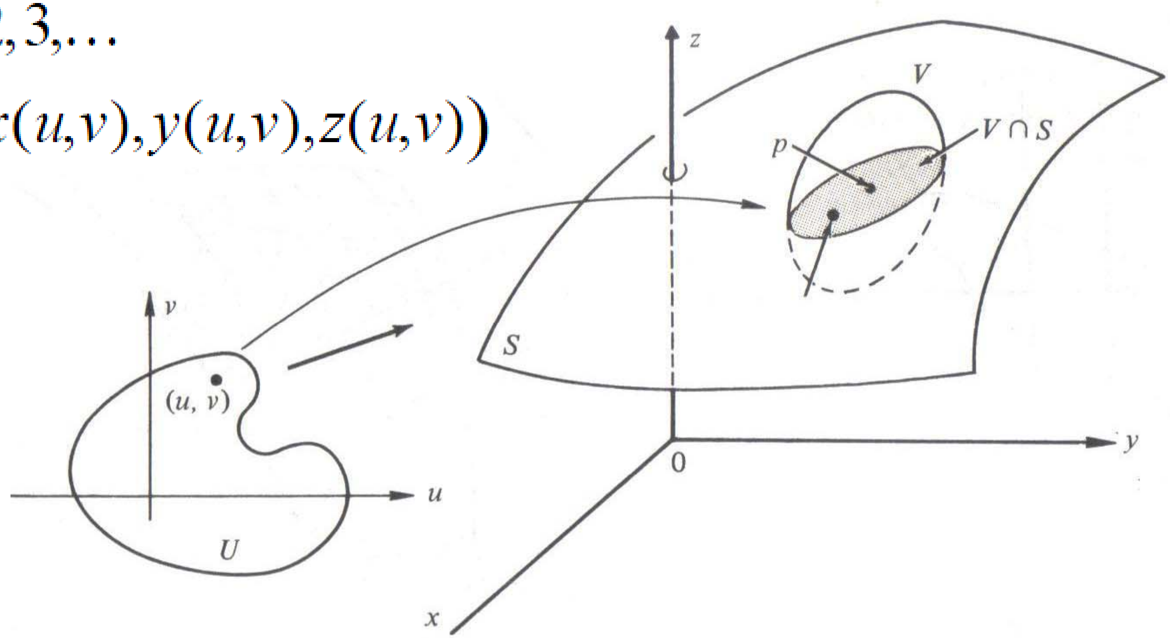


Figure 2-1

Mesh Representations

- Parametric
  - Represent a surface as (continuous) injective function from a domain $\Omega \subset \boldsymbol{R}^2$ to $S \subset \boldsymbol{R}^3$
- In practice, it's not easy to find a single function that parameterizes the surface.
- So instead, we represent a surface as a collection of functions (charts) from (simple) 2D domains into 3D.

Mesh Representations

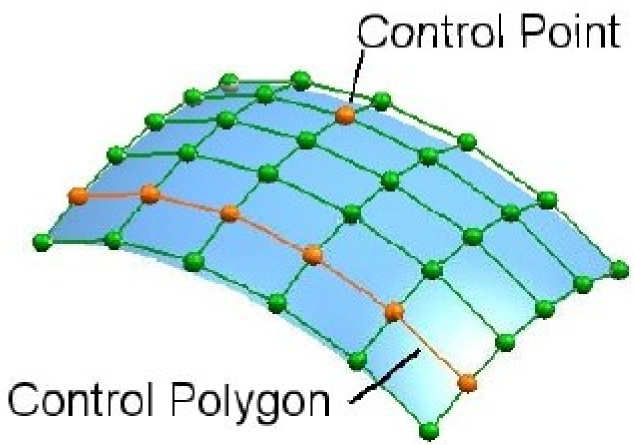
- Parametric
  - Represent a surface as (continuous) injective function from a domain $\Omega \subset \boldsymbol{R}^2$ to $S \subset \boldsymbol{R}^3$

Given a set of charts, we say that the manifold S is "smooth" if for any two charts $\phi_1 : \Omega_1 \to S$ and $\phi_2 : \Omega_2 \to S$, the map $\phi_2^{-1} \circ \phi_1$ is smooth.

Spline & NURBS

- Extract analytical rep.
- Support interactive shape editing
- Compact rep.

- Major modeling techniques in CAD



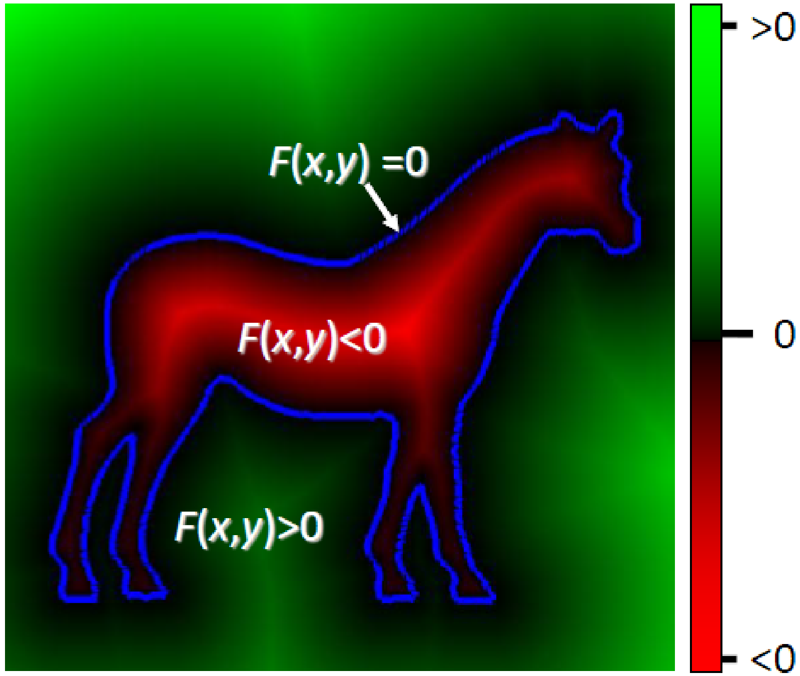
Shape Representation

- Parametric surface
- Implicits
- Points
- Polygon mesh

Shape Representations

- Parametric
  - Represent a surface as (continuous) injective function from a domain $\Omega \subset \boldsymbol{R}^2$ to $S \subset \boldsymbol{R}^3$
- Implicit
  - Represent a surface as the zero set of a (regular) function defined in R^3.

$$K = g^{-1}(0) = \{\mathbf{p} \in \mathrm{R}^3 : g(\mathbf{p}) = 0\}$$



Implicit Surfaces

Gradient

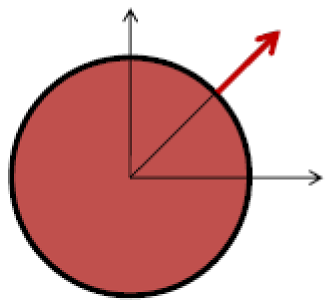
- Represent a surface as the zero set of a (regular) function defined in R^3.
- The normal vector to the surface is given by the gradient of the (scalar) implicit function

$$\nabla g(x,y,z) = \left(\frac{\partial g}{\partial x}, \frac{\partial g}{\partial y}, \frac{\partial g}{\partial z} \right)^T$$

  – Example

$$g(x,y,z) = x^2 + y^2 + z^2 - r^2$$

$$\nabla g(x,y,z) = (2x, 2y, 2z)^T$$



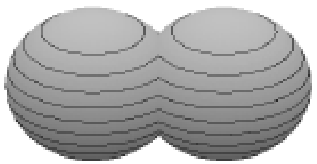
$$\nabla g(x,y,z) = (2,2,0)^T$$

- Why is the condition that the function be regular (i.e. have non-vanishing derivative) necessary?
- How smooth is the surface?

Implicit Surfaces

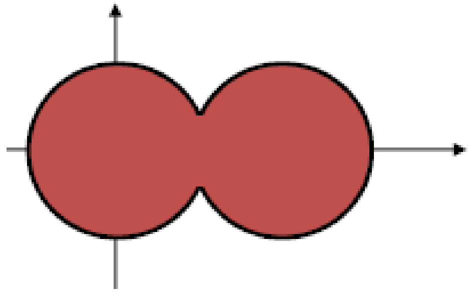
Smooth set operation

- Standard operations: union and intersection



$$\bigcup_i g_i(\mathbf{p}) = \min g_i(\mathbf{p})$$

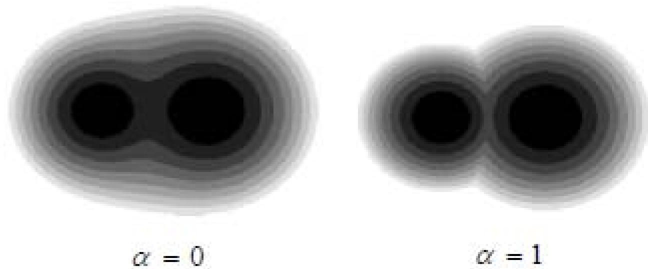
$$\bigcap_i g_i(\mathbf{p}) = \max g_i(\mathbf{p})$$



- In many cases, smooth blending is desired
  - Pasko and Savchenko [1994]

$$g \cup f = \tfrac{1}{1+\alpha}\left(g + f - \sqrt{g^2 + f^2 - 2\alpha g f}\right)$$

$$g \cap f = \tfrac{1}{1+\alpha}\left(g + f + \sqrt{g^2 + f^2 - 2\alpha g f}\right)$$



  - alpha

$$\lim_{\alpha\to 1} g \cup f = \tfrac{1}{2}\left(g + f - \sqrt{(g-f)^2}\right) = \frac{g+f}{2} - \frac{|g-f|}{2} = \min(g, f)$$

$$\lim_{\alpha\to 1} g \cap f = \tfrac{1}{2}\left(g + f + \sqrt{(g-f)^2}\right) = \frac{g+f}{2} + \frac{|g-f|}{2} = \max(g, f)$$

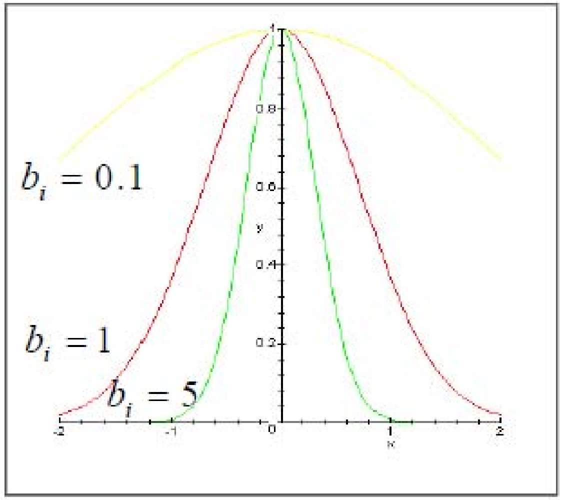
Implicit Surfaces

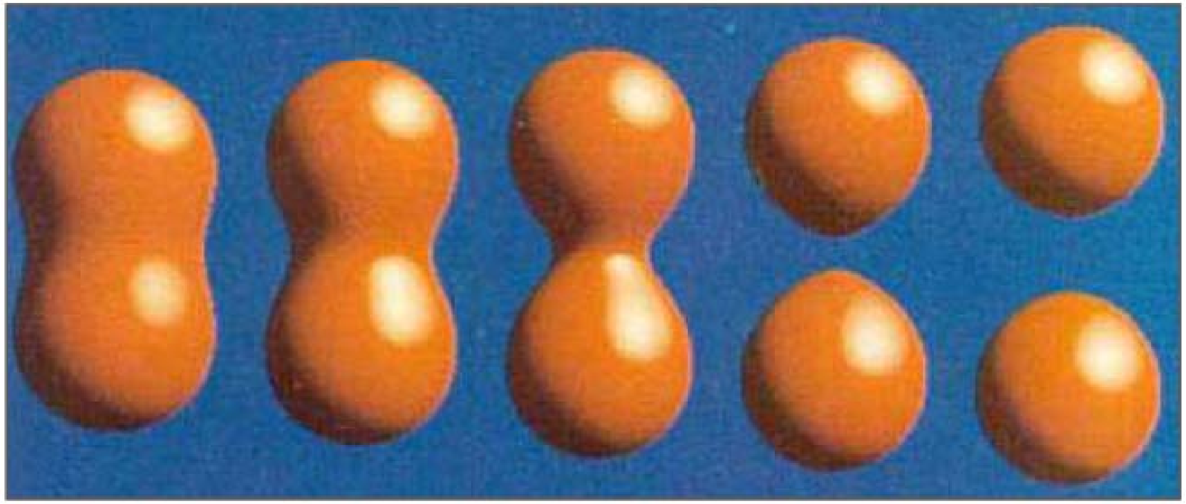
Blobs

- Suggested by Blinn [1982]
  - Defined implicitly by a potential function around a point pi: $\mathrm{F(x,y,z)} = \sum_i g_i(p) - T$

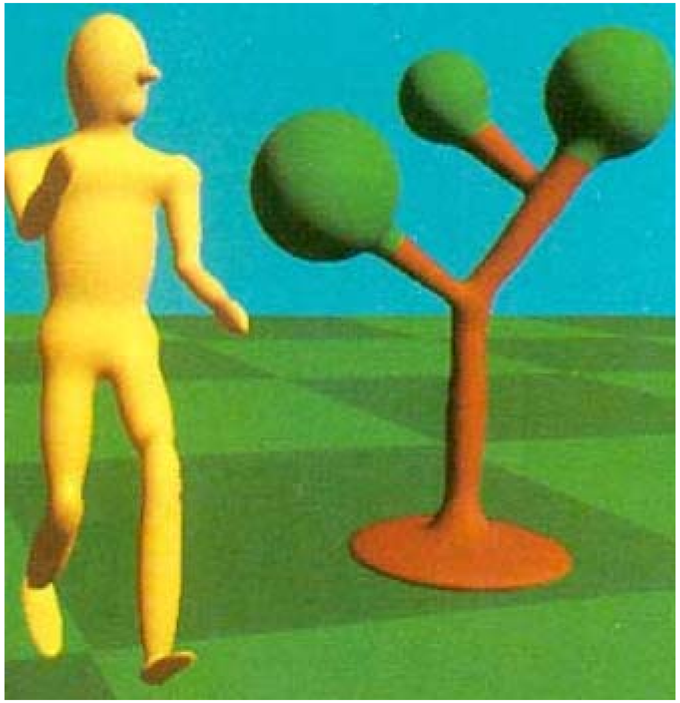
$$g_i(\mathbf{p}) = a_i e^{-b_i \|\mathbf{p}-\mathbf{p}_i\|^2}$$

- Set operations by simple addition/subtraction

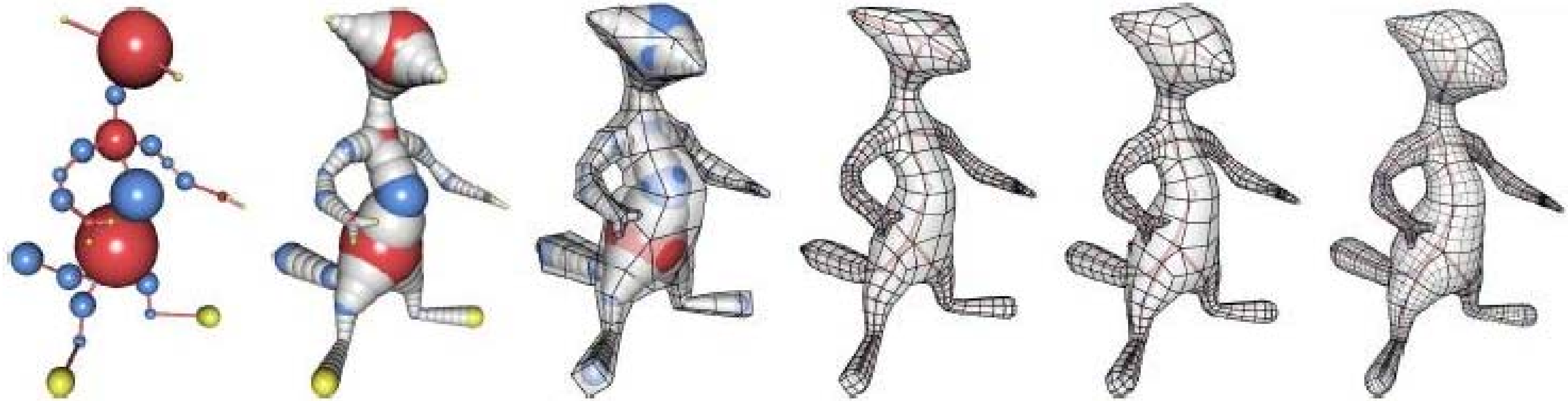




Blinn [1982]: A Generalization Drawing of Algebraic Surface



pg10_B-Mesh: A Fast Modeling System for Base Meshes of 3D Articulated Shapes



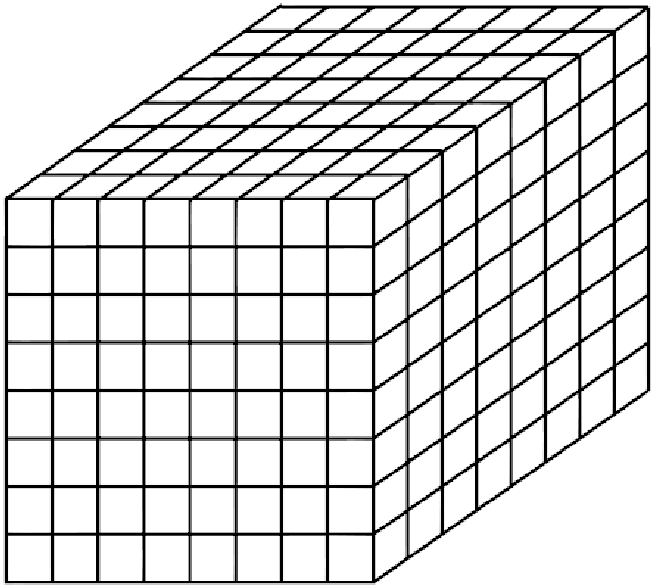
Parametric v.s. Implicit

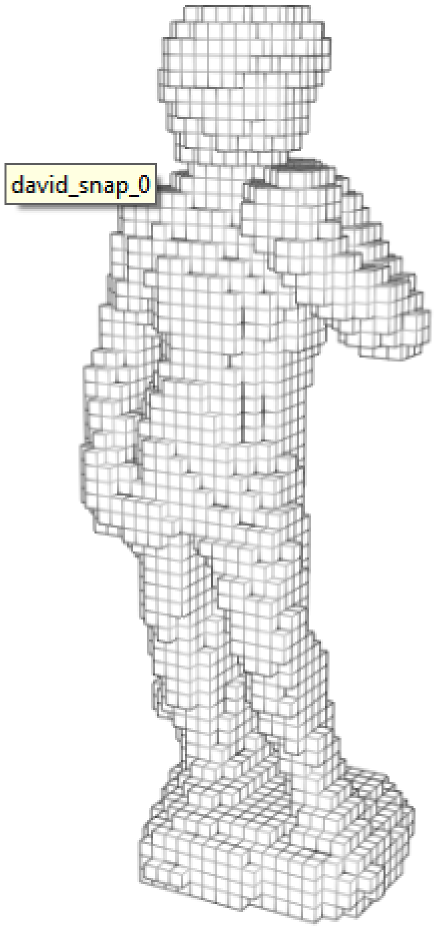
- Parametric
  - Easy to enumerate points on the surface
  - Easy to find neighbors
  - Hard to determine inside/outside
  - Hard to determine if a point is on the surface
- Implicit
  - Easy to determine if you are inside or outside
  - Easy to determine if a point is on the surface
  - Easy to modify the topology of the surface
    - Simulation, Reconstruction (Hole-filing)
  - Usually compact
  - Hard to generate points on the surface
  - Does not lend itself to (real-time) rendering

Implicit Representations

Voxel Grids:

Represented by the values of the function on regular grid.





Implicit Representations

Voxel Grids:

Represented by the values of the function on regular grid.

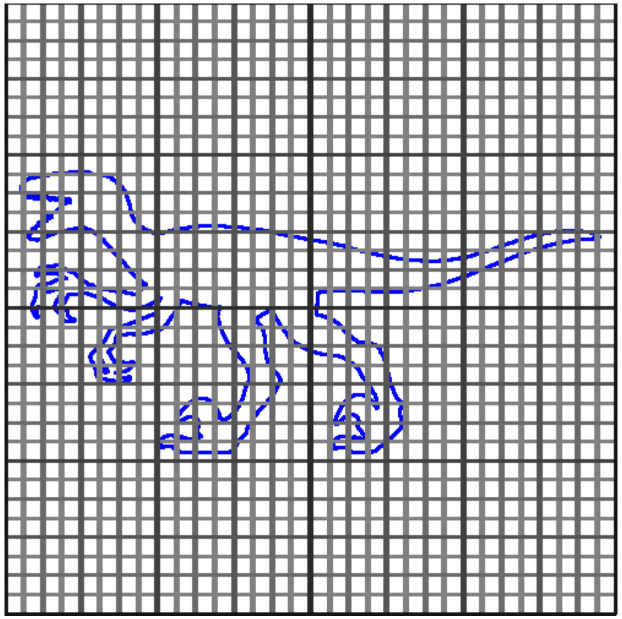
Though binary voxel grids are simplest, often represent the (signed) Euclidean Distance Transform:

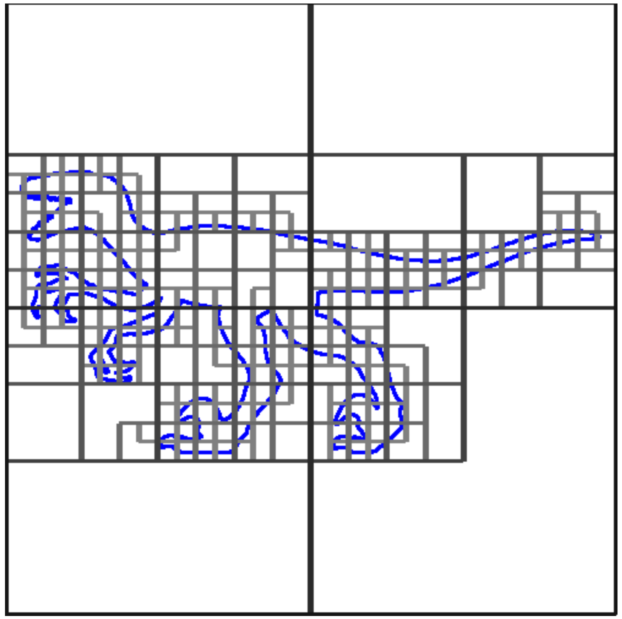
$$EDT^2(p) = \min_{q \in S} \| p - q \|^2$$

Implicit Representations

Adaptive Grids: Quadtree

In practice, we may only need a high-precision representation of the implicit function near the surface, so represent the function over an adaptive grid.





Shape Representation

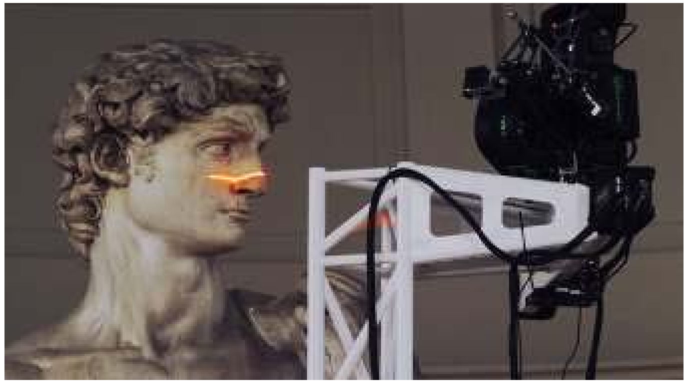
- Parametric surface
- Implicits
- Points
- Polygon mesh

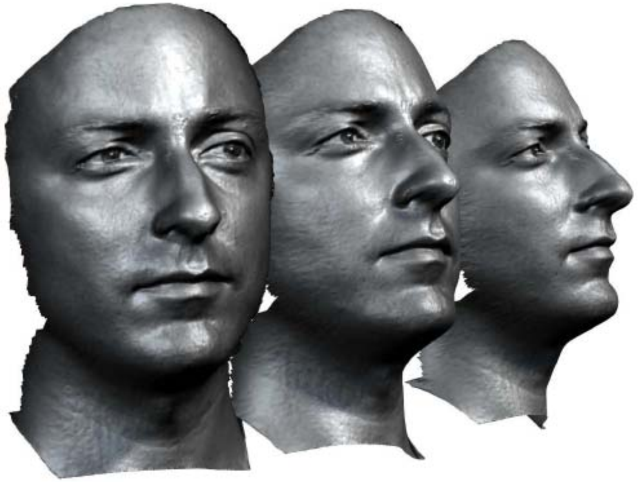
Shape Acquisition

Sampling of real world objects

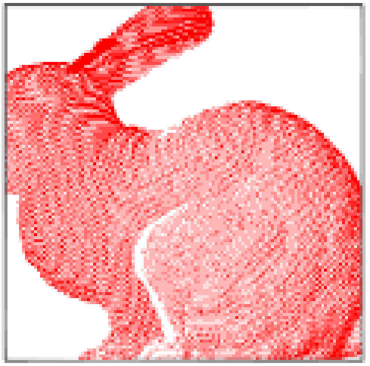
Points

- Scanners
  - Laser
  - Depth imaging
- Properties & Operations
  - Potentially noisy, with outliers
  - Registration of multiple images
  - Non-uniform sampling, sparse, holes

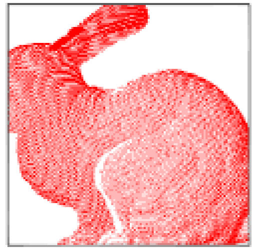




180 frames per second (From David Gu)



Points

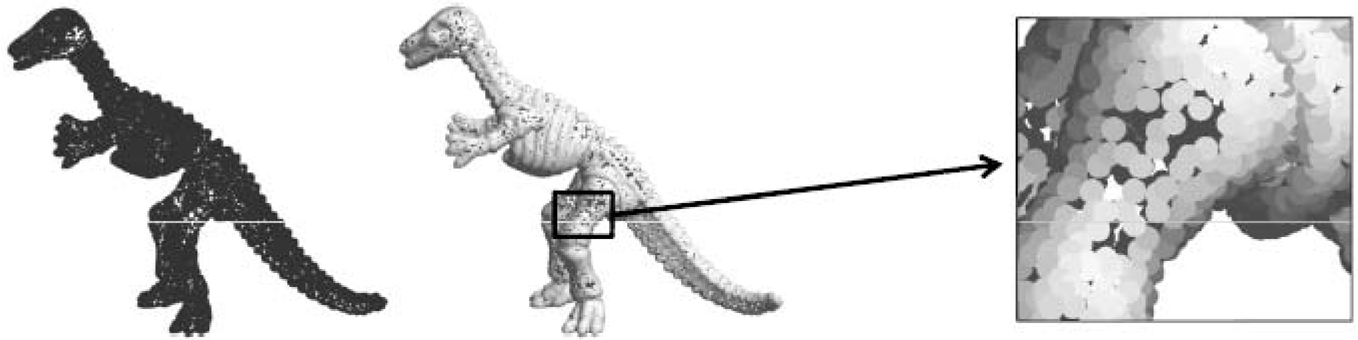


- Points = unordered set of 3-tuples
- Often converted to other reps
  - Easier to process, edit & render
  - Meshes, implicits, parametric surfaces
- Efficient point processing & modeling requires a spatial partitioning data structure
  - To figure out **neighborhoods**

Points

Neighborhood information

- Why do we need neighbors?



need normals (for shading)

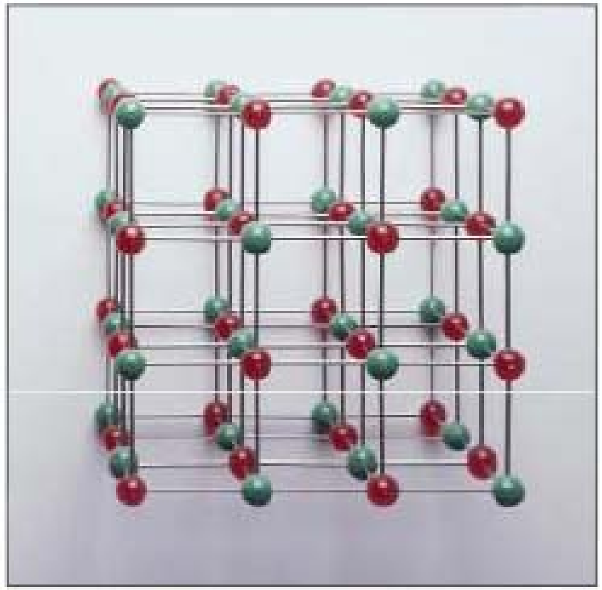
upsampling – need to count density

- Need sub-linear implementations of
  - K-nearest neighbors to point x (knn)
  - In radius search

Spatial Data Structures

Commonly used for point processing

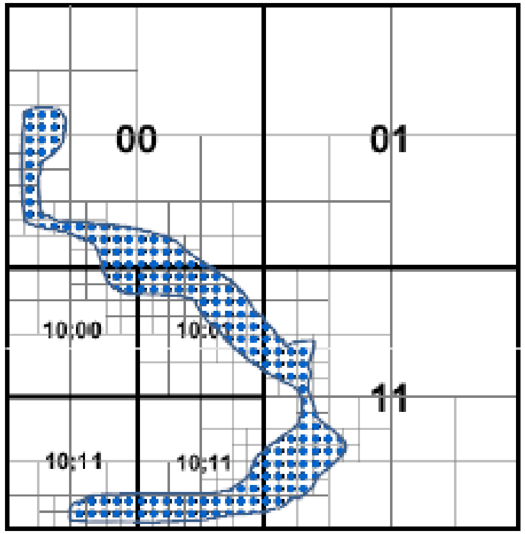
- Regular uniform 3D lattice
  - Simple proximity queries by searching neighboring cells
  - Determining lattice parameters (i.e. cell dimensions) is non-trivial
  - Generally unbalanced, i.e. many empty cells



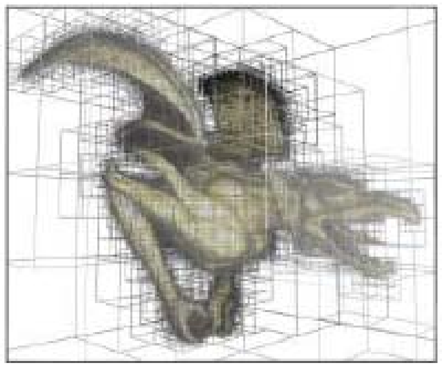
Spatial Data Structures

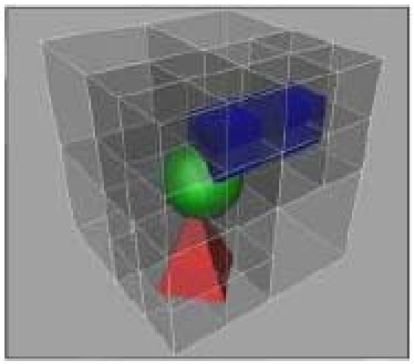
Commonly used for point processing

- Octree
  - A hierarchical tree build by sequential subdivision (8) of a occupied cells.
  - Adaptive, i.e. only splits when too many points in cell
  - Proximity search by (recursive) tree traversal and distance to neighboring cells
  - Tree might not be balanced
  - Widely used for complicated scenes that need faster processing and lower accuracy



E.g. Collision detection in real-time simulation or animation

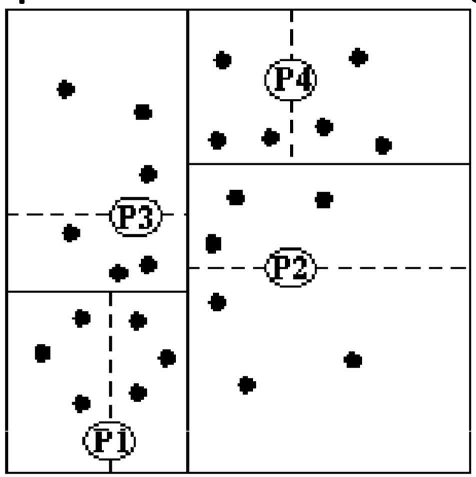


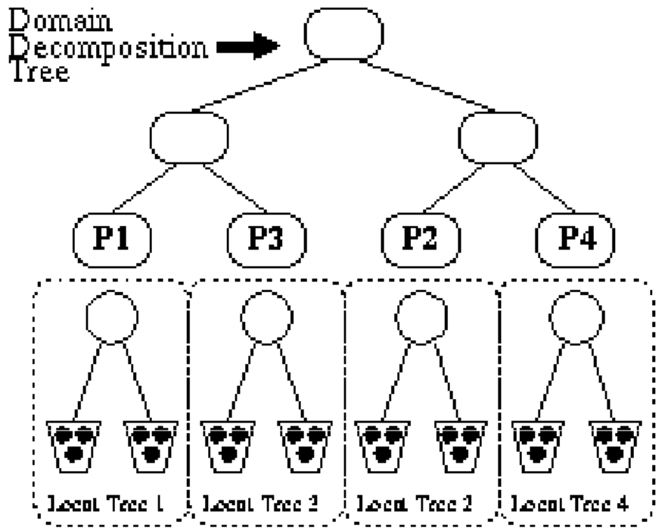


Spatial Data Structures

Commonly used for point processing

- Kd-Tree
  - Each cell is individually split along the median into two cells
  - Same amount of points in cells
  - Perfectly balanced tree
  - Proximity search similar to the recursive search in an Octree.
  - More data storage required for inhomogeneous cell dimensions





Shape Representation

- Parametric surface
- Implicits
- Points
- Polygon mesh

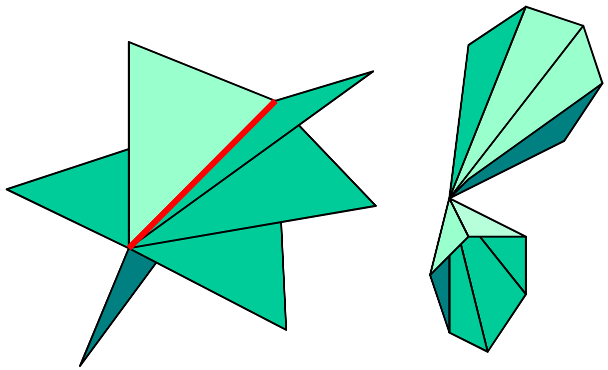
Polygonal Meshes

- Topology
  - Simplicial Complex, Combinatorics
  - connectivity of the vertices
- Geometry
  - Conformal Structure – Corner angles (and other variant definitions)
  - Riemannian metrics – Edge lengths
  - Embedding – Vertex coordinates

Triangle Mesh

Definition (Mesh)

- A triangle mesh is a oriented two dimensional simplical complex, generally embedded in R^3.
- Non-manifold



Triangle mesh is represented as $M = \{V, E, F\}$

$$V = \{\mathbf{v}_i = (x_i, y_i, z_i)^T, i = 1, 2, \ldots, n\}$$

$$E = \{(i, j) \mid \mathbf{v}_i, \mathbf{v}_j \ \textit{are linked by an edge}\}$$

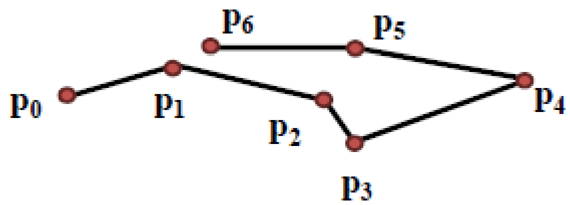
$$F = \{(i, j, k) \mid \mathbf{v}_i, \mathbf{v}_j, \mathbf{v}_k \ \textit{are the vertices of a face}\}$$

Definitions

Geometric graph

- A **Graph** is a pair G=(V,E)
- The **degree** or **valence** of a vertex describes the number of edges incident to this vertex.

A geometric graph Q=(V,E) with $V=\{\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_{n-1}\} \subset \mathbb{R}^d$ with $d \geq 2$ and $E=\{(\mathbf{p}_0, \mathbf{p}_1),(\mathbf{p}_1, \mathbf{p}_2),\ldots,(\mathbf{p}_{n-2}, \mathbf{p}_{n-1})\}$ is a *polygon*

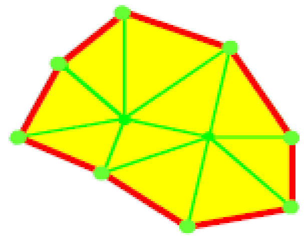


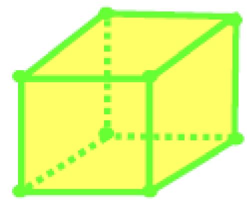
- A polygon is
- Planar, if all vertices lie on a plane
- Closed, if p0=p_{n-1}
- Simple, if it does not self-intersect

Definitions

Polygonal mesh

- The set of all edges that belong to only one polygon is termed the ***boundary*** of the polygonal mesh, and is either empty or forms closed loops
- The **polygonal** mesh is closed if it has no **boundary**.

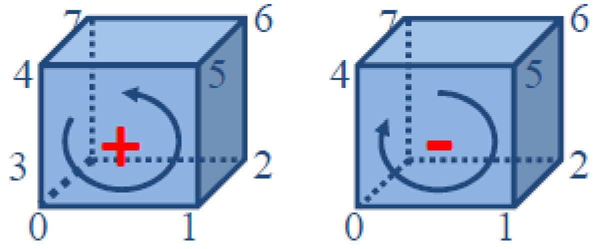




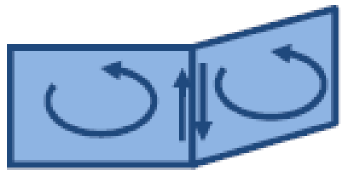
Definitions

Orientation

- Every face of a polygonal mesh is orientable
  - By defining “counterclockwise” or “colockwise”



  - Defines the sign of the surface normal
  - Two neighboring facets are equally oriented, if the edge directions of the shared edge (induced by the face orientation) are opposing



- A polygonal mesh is orientable, if the incident faces to every edge can be equally oriented.

Euler-Poincaré Formula

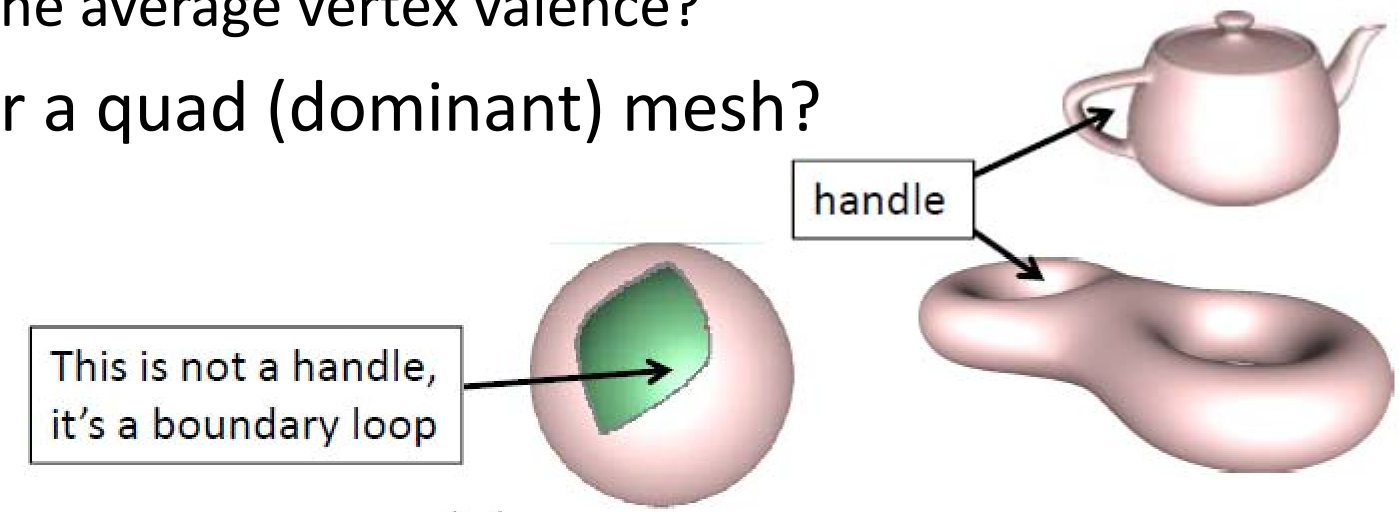
For a closed, connected, water-tight mesh with genus g, the number of the vertices (V), edges (E), & faces (F) satisfy:

$$V-E+F=2-2g=\chi$$, **Euler characteristic**

For a triangle mesh:

- What is the ratio of triangles to vertices?
- What is the ratio of edges to vertices?
- What is the average vertex valence?

How about for a quad (dominant) mesh?



Euler-Poincaré Formula

Generalization

Theorem: Let

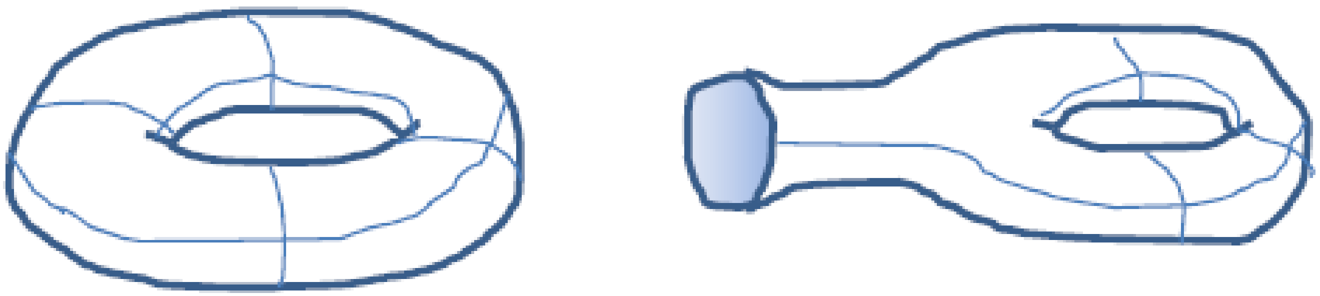
h - # boundary loops

c - # connected components

G - # handles (genus)

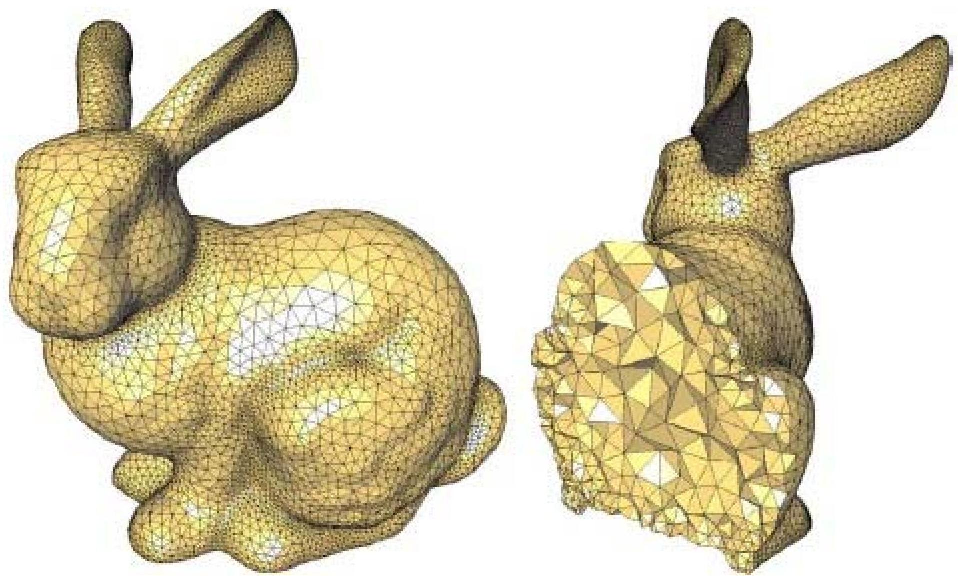
Then:

$$v - e + f - h = 2\,(c - g)$$

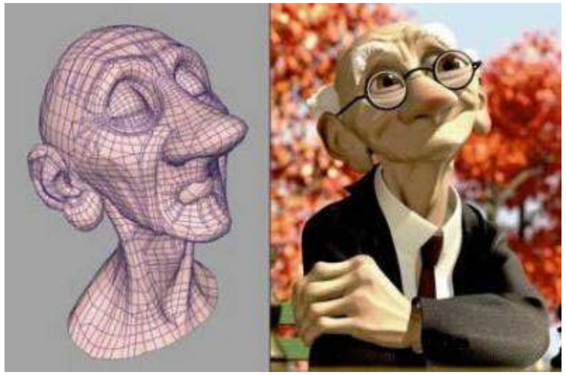


Tetrahedral Mesh

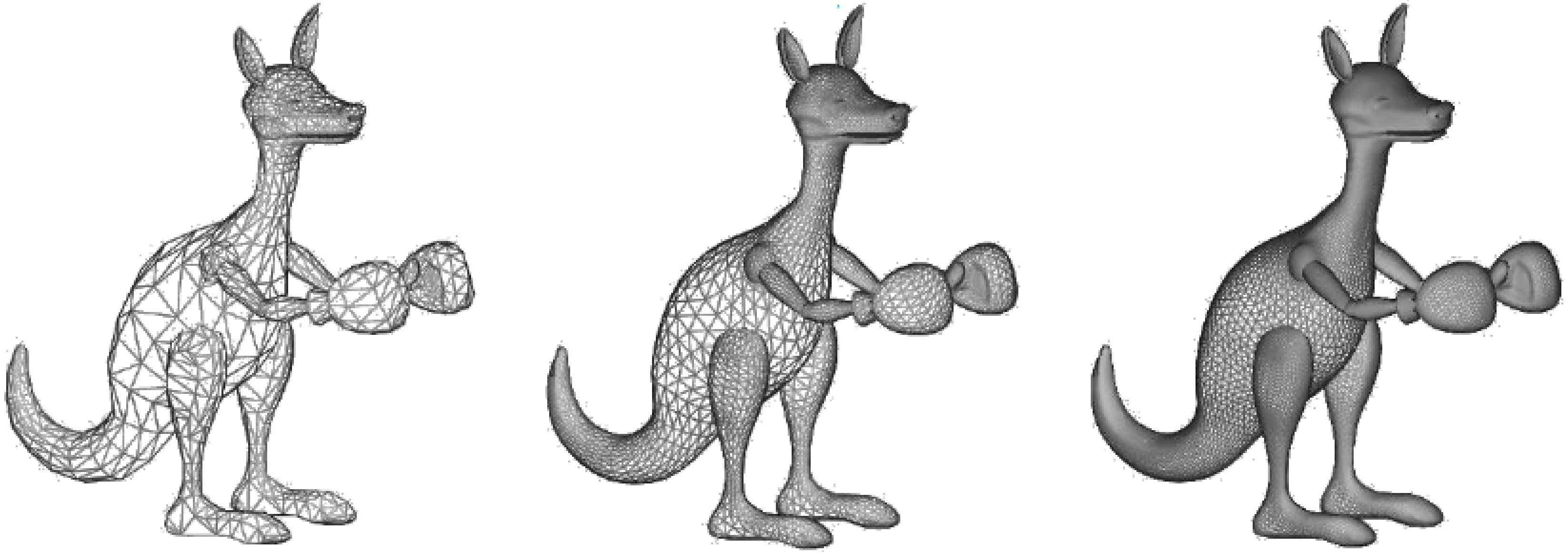
- Solid shapes can be tetrahedralized



Mesh Representation



- Subdivision Surfaces
  - Given a base (triangle) mesh & a set of rules for refining the geometry.
  - Repeated subdivision results in a surface with provable smoothness properties.



Shape Representation

- Parametric surface
- Implicits
- Points
- Polygon mesh
- Transition Between Representations
- Others

Transitioning Between Representations

Parametric to Implicit

Assign distance values to points on a voxel grid:

Naïve:

- For each voxel, find the closet point on the surface & use that to set the voxel's value.

Efficient:

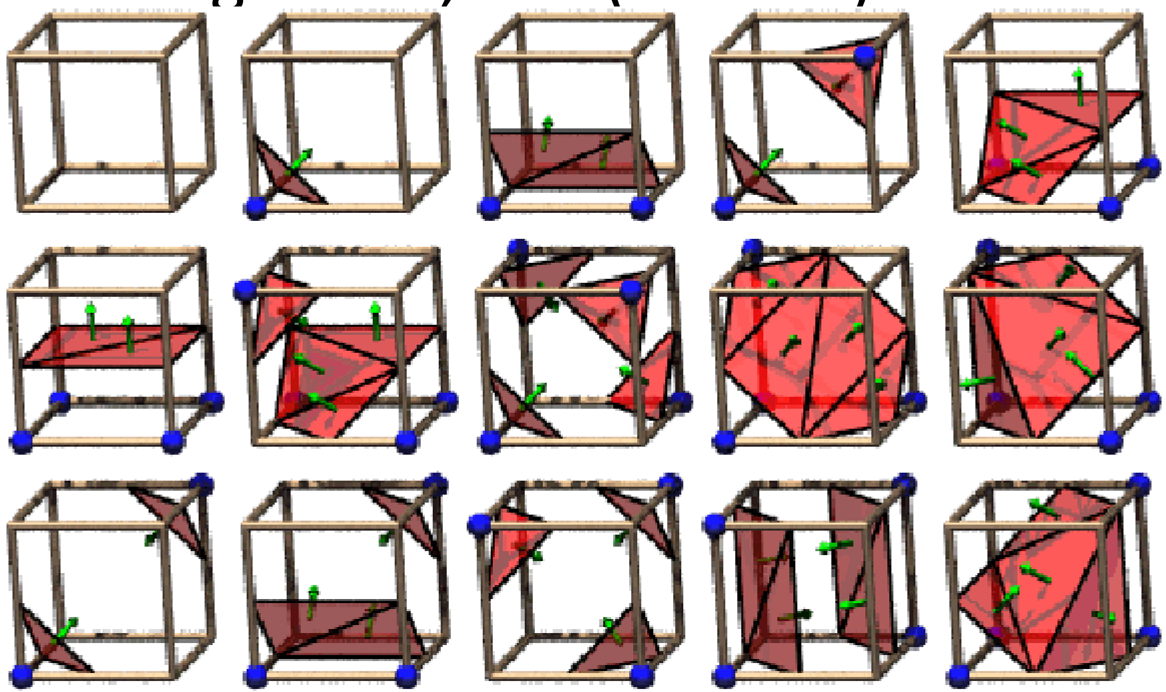
- For each voxel near the surface, find the closet point on the surface (using a kd-tree) and use that to set voxel's value.
- Use the fast marching method to define distance values away from the surface

Transitioning Between Representations

Implicit to Parametric

Extract the zero-set of the implicit function:

- Marching cubes (for voxel grids)
- Dual marching cubes, etc. (octrees)



The 15 Cube Combinations

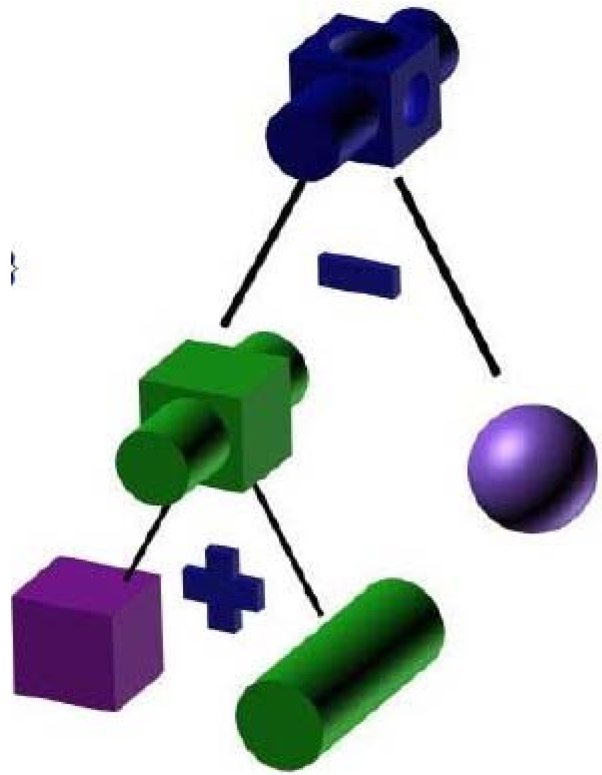
Shape Representation

- Parametric surface
- Implicits
- Points
- Polygon mesh
- Transition Between Representations
- Others

CSG Representation

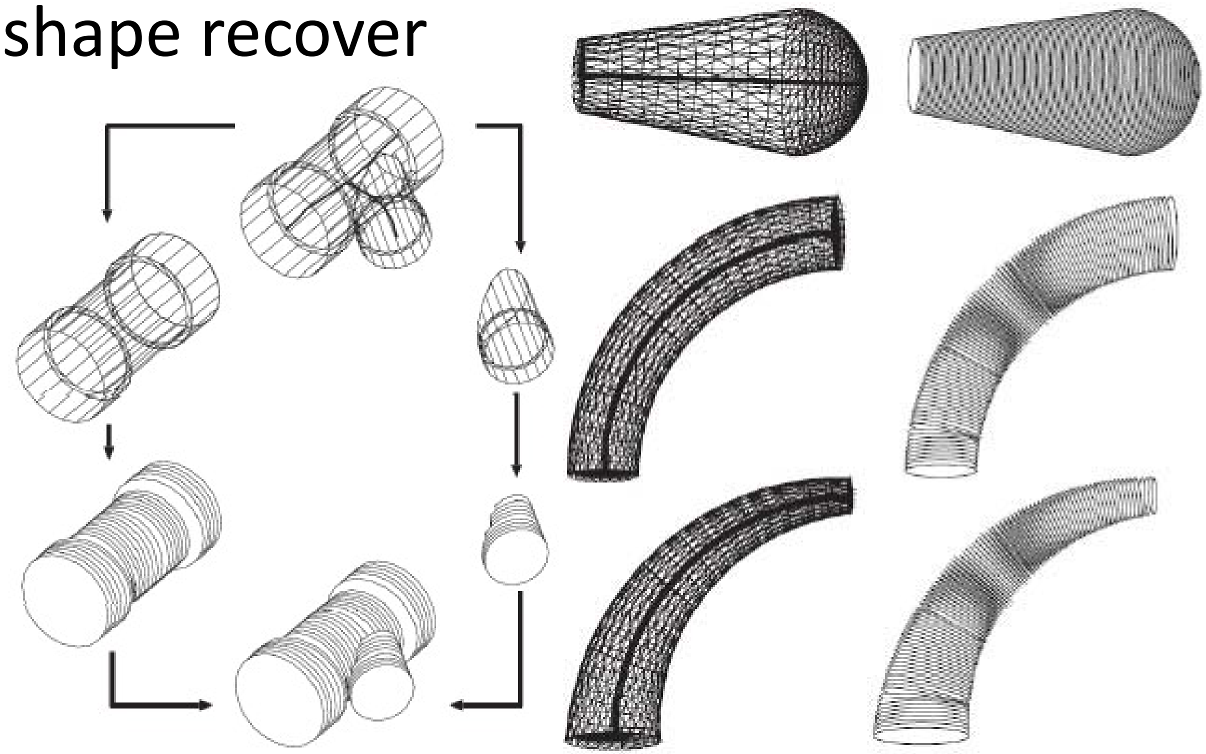
- Polygonal Mesh -> machine-oriented representation
- CSG -> user-oriented representation
  - Store the “logic of the shape”
- A CSG modeling system = {building blocks, Boolean operations}

- Widely used in 3DMax, Maya…
  - Support user-intervention
  - Good for simple shapes



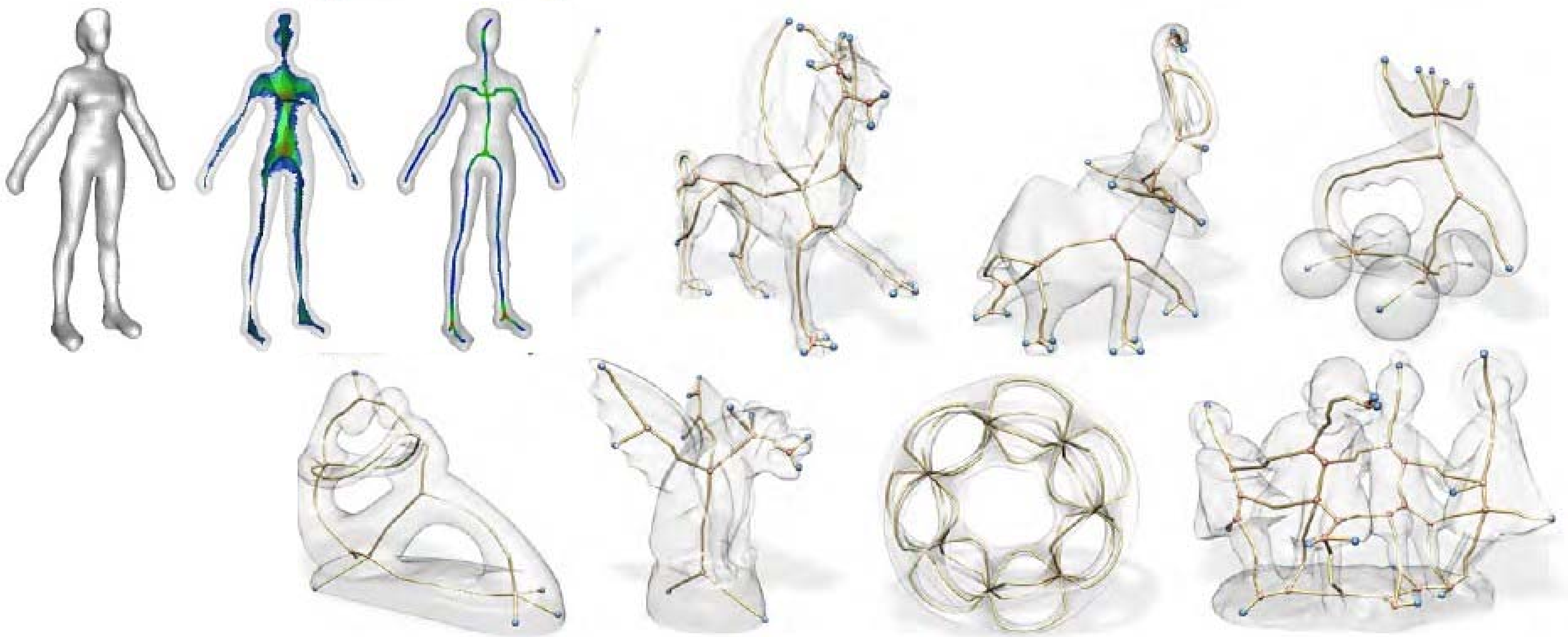
Generalized cylinder rep.

- A shape = {axis, a cross-section curve, a scaling function}
- Good for symmetric shapes with few local details and with clear skeletal structure
- Widely used in vision community for shape recognition, and shape recover

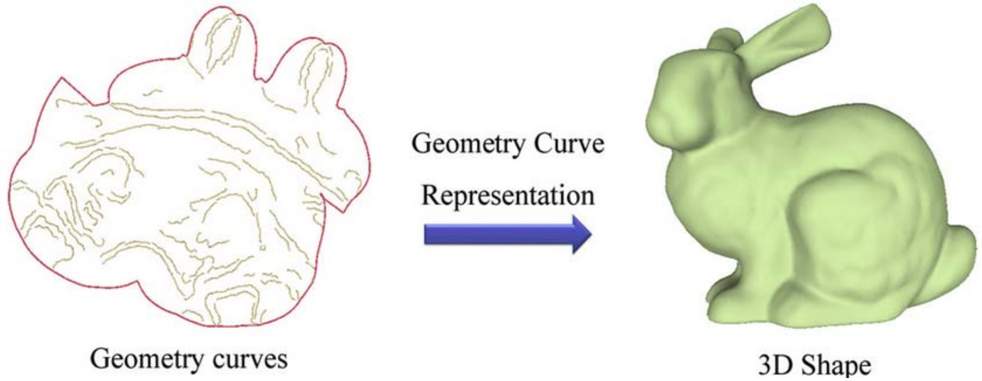


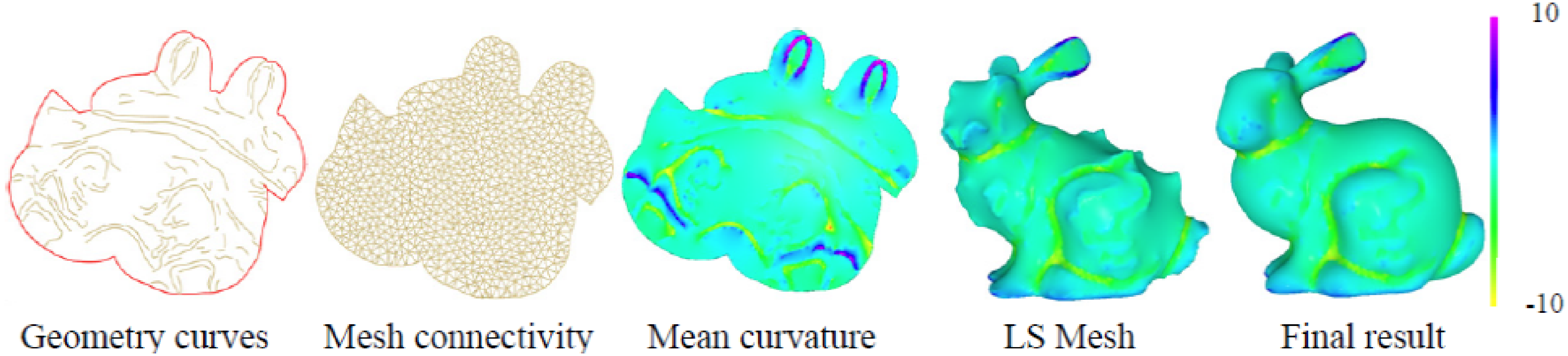
Skeleton Rep.

- A thin 1D or 2D representation of 2D/3D objects
- A (hierarchical) set of bones + attached skins
- Widely used in animation, matching, object recognition



Gm13_Geometry Curves: A Compact Representation for 3D Shapes





Resources

- read & display a mesh: jjcao_plot/eg_trisurf.m
- Read & display a huge point set (100k to 1 million points)
  - PC_processing_1.0
  - jjcao_code/tools/pcd_viewer

References

- **Michael Misha Kazhdan: Advanced Topics in Computer Graphics: Mesh Processing (600.657)**
- **Andrew Nealen: CS 523: Computer Graphics : Shape Modeling**
- David Gu: Computational Conformal Geometry 2010