Implicit Surfaces

Misha Kazhdan

CS598b

Definition:

Given a function *F* the implicit surface *S* generated by this function is the set of zero points of *F*:

$$S = \{p \,|\, F(p) = 0\}$$

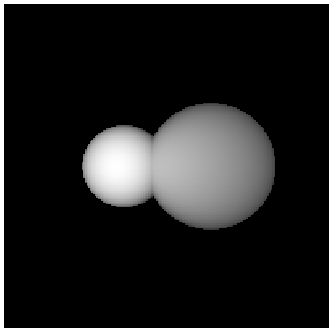
The interior *I* is the set of points on which *F* is negative:

$$I = \{p \,|\, F(p) < 0\}$$

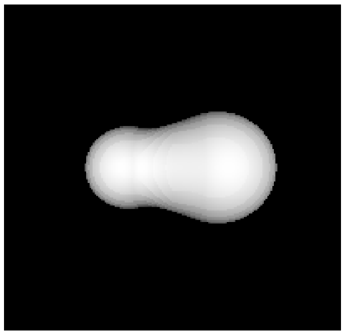
Examples:

- Algebraic surfaces: Given by polynomial equations.
  - Quadric Surfaces (easy eigenvalue analysis)
  - Higher Degree Surfaces
- Skeletons (minimal distance to surface)
- Metaballs/Blobbies/Soft-Objects (point charge model; field functions)
- Convolution Surfaces

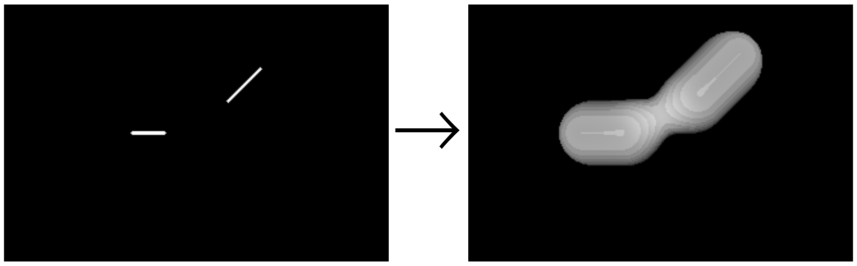
Skeleton



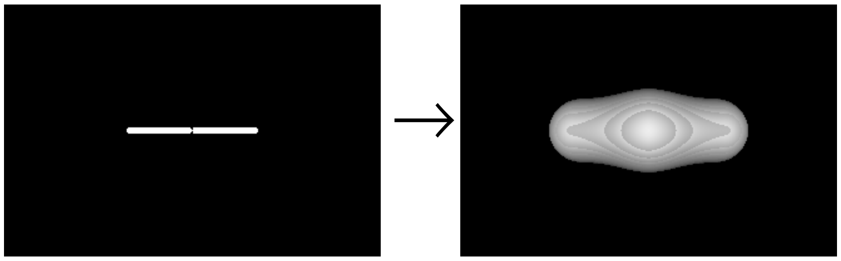
Blended



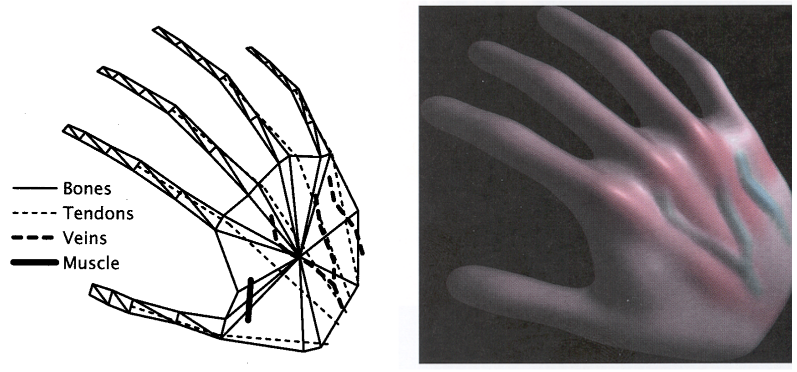
Blending Arbitrary Skeletons:



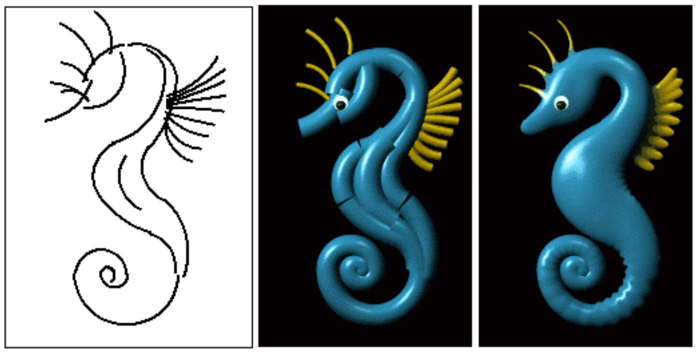
The Bulge Problem:



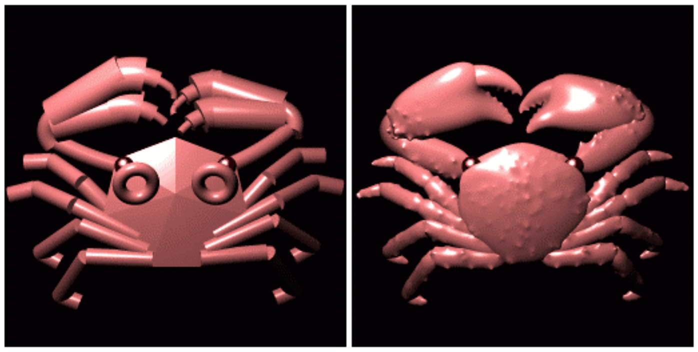
[Bloomenthal. *Introduction to Implicit Surfaces*. Figure 7.16]



[http://www.ugcs.caltech.edu/~andrei/papers/seafood/horse.gif]



[http://www.ugcs.caltech.edu/~andrei/papers/seafood/



Arguably, the most complicated convolution surface out there. (At least that's what they claim.)

Reconstruction Methods

- Point Charge Model
  - **Iteratively find new points by splitting old ones**
  - Generate a list possible points and iteratively choose from the list
- Local reconstruction of the distance function

[Muraki. *Volumetric Description of Range Data using "Blobby Model"*. Computer Graphics, Volume 25, Number 4,July 1991]

Problem:

Given a collection of range data points $\{q_i\}$ and "some" normal information, define an implicit function interpolating these data points.

Idea of the Solution:

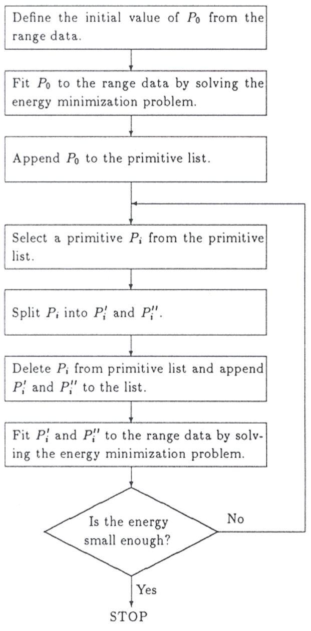
To iteratively find a set of points $\{p_i\}$ with associated field functions $\{f_i\}$ such that the implicit function:

$$F(x) = 1 - \sum_i f_i\left(\|x - q_i\|\right)$$

interpolates the datapoints and conforms with the normal information.

At each iteration, some point is split in two, and the location of the two points, and their associated field functions are optimized to minimize an energy function.

[Muraki]


Define the initial value of P_0 from the range data.

Fit P_0 to the range data by solving the energy minimization problem.

Append P_0 to the primitive list.

Select a primitive P_i from the primitive list.

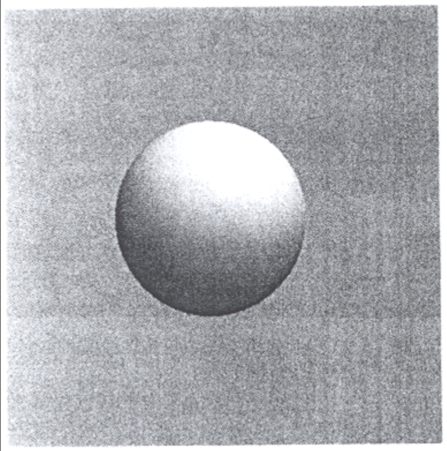
Split P_i into P_i' and P_i''.

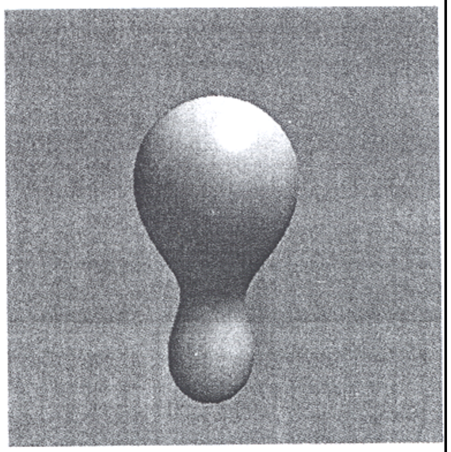
Delete P_i from primitive list and append P_i' and P_i'' to the list.

Fit P_i' and P_i'' to the range data by solving the energy minimization problem.

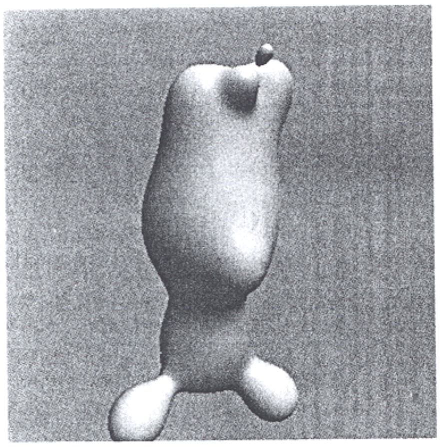
Is the energy small enough?

<u>Face 1</u>:

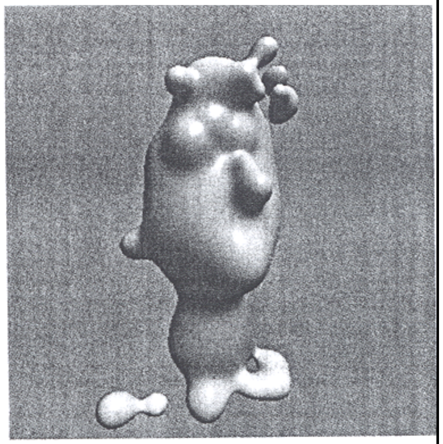

(a) $N = 1$


(b) $N = 2$

Face 2:

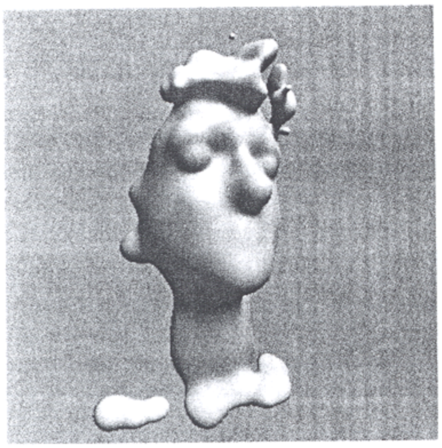


(c) $N = 10$

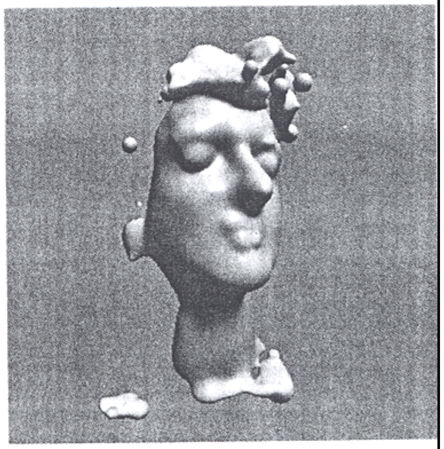


(d) $N = 35$

Face 3:

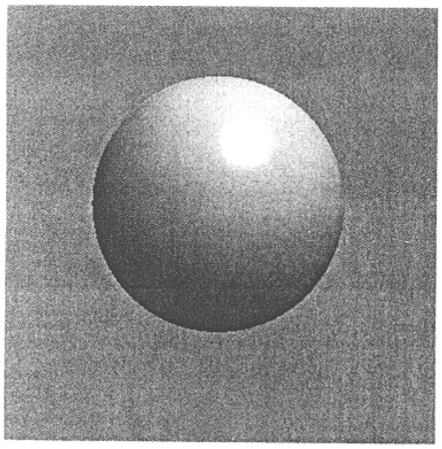


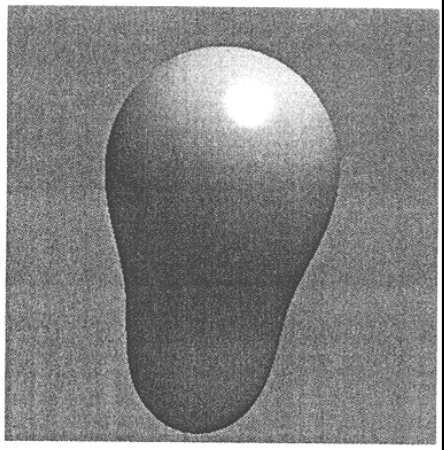
(e) $N = 70$



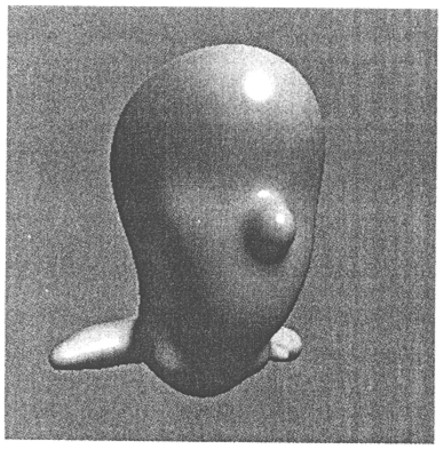
(f) $N = 243$

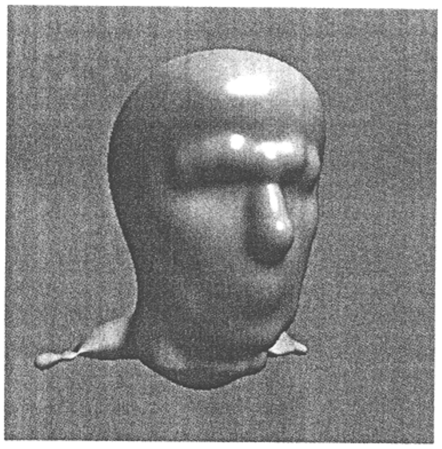
Head 1:


(a) $N = 1$

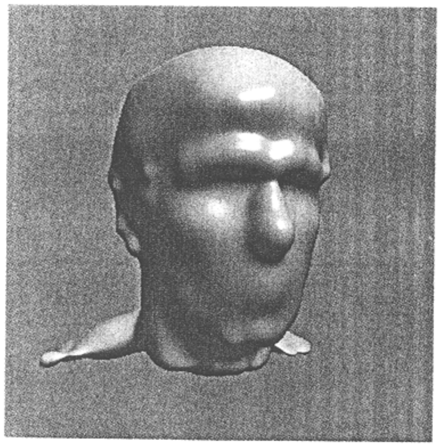

(b) $N = 2$

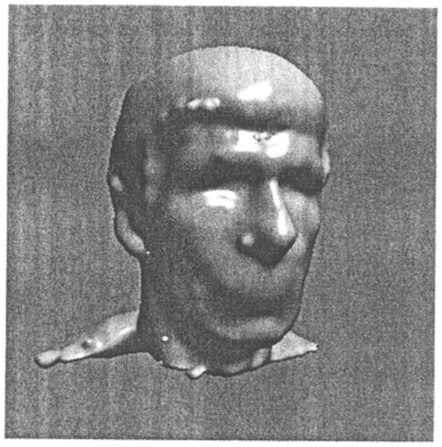
Head 2:


(c) $N = 20$


(d) $N = 60$

Head 3:


(e) $N = 120$


(f) $N = 451$

Energy Function:

The energy function is a weighted sum of three components corresponding to three different properties minimized:

- "Distance" of the data points $\{q_i\}$ from the implicit surface.
- Discrepancy of the point normal and the surface gradient.
- The overall strength of the charge.

- Face: 256x256 data points (It took several days to get the N=243 model)
- Head: 5334 data points.

Problems:

- Computationally very slow. (Especially since a full optimization is run for each point in the list before selecting the one split.)
- The new point chosen is restricted to starting off from an existing point.
- Not usable in the case that there is no normal data (as you might be modeling the outside instead of the inside.)
- The field functions do not have compact support so this method does not really allow for local control.

Reconstruction Methods

- Point Charge Model
  - Iteratively find new points by splitting old ones
  - **Generate a list possible points and iteratively choose from the list**
- Local reconstruction of the distance function

[Bittar, Tsingos and Gascuel. *Automatic Reconstruction of Unstructured 3D Data:Combining Medial Axis and Implicit Surfaces*]

Problem:

Given a collection of data points $\{q_i\}$ to locate the interior of the surface and define an implicit function interpolating the data points.

(This method seeks to improve on Muraki's method by initially generating a list of potential skeleton points and using a field function with compact support.)

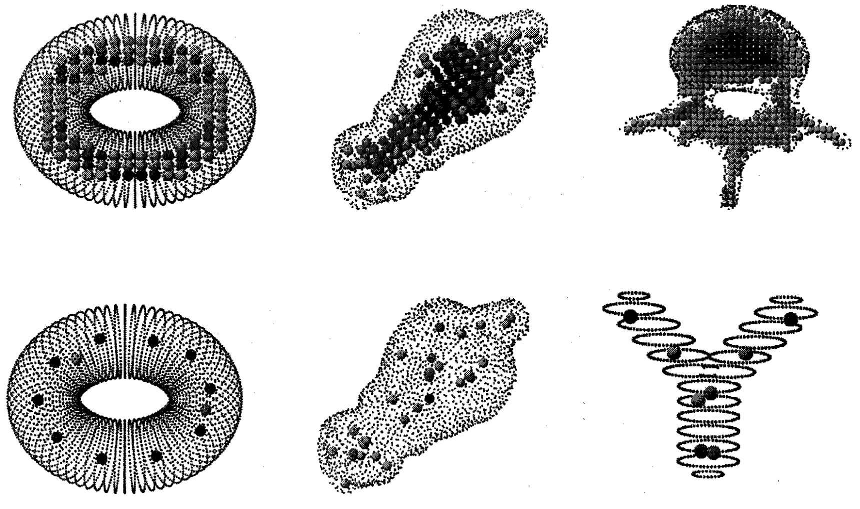
Idea of the Solution:

- Voxelize at appropriate resolution.
- Propagate external point information.
- A subset of interior points is chosen as candidate skeleton points.
- From this set, points are chosen iteratively.

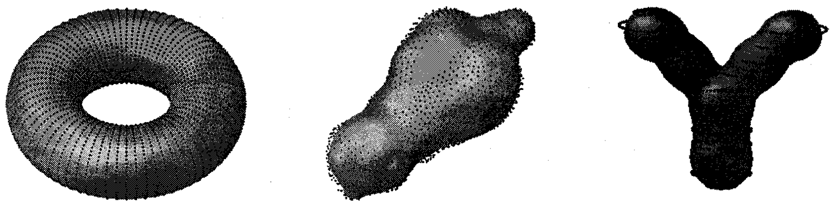
Potential Skeleton List Generation:

- Each interior voxel has a maximal radius.
- If the maximal sphere about the voxel is not contained in the maximal sphere about any other voxel, this voxel is added to the candidate list.

[Bittar, Tsingos, and Gascuel]

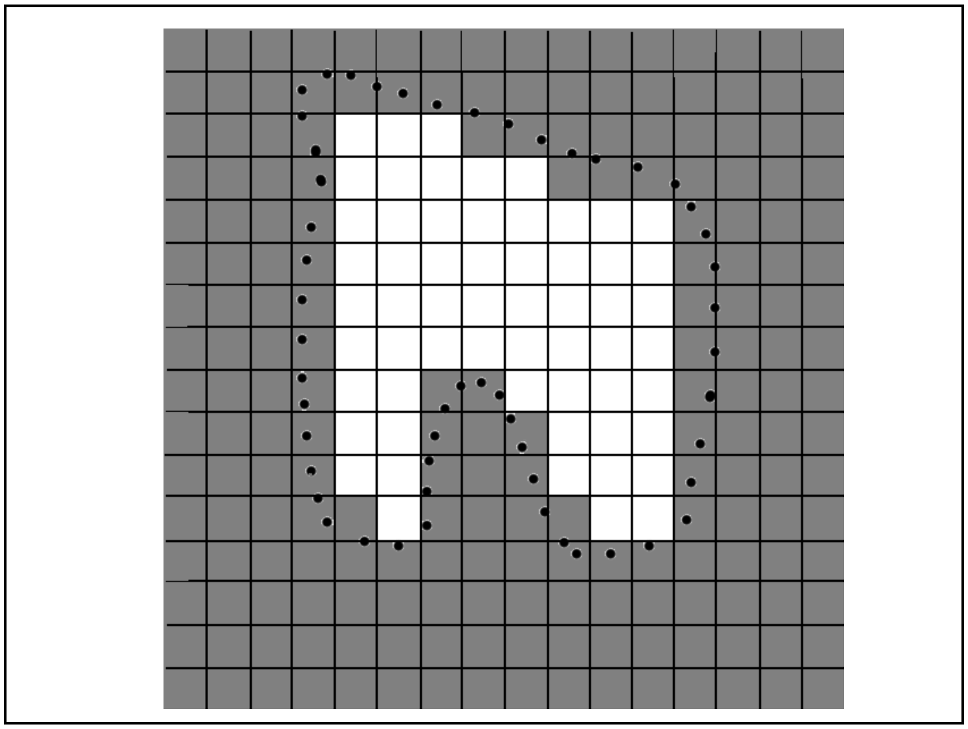


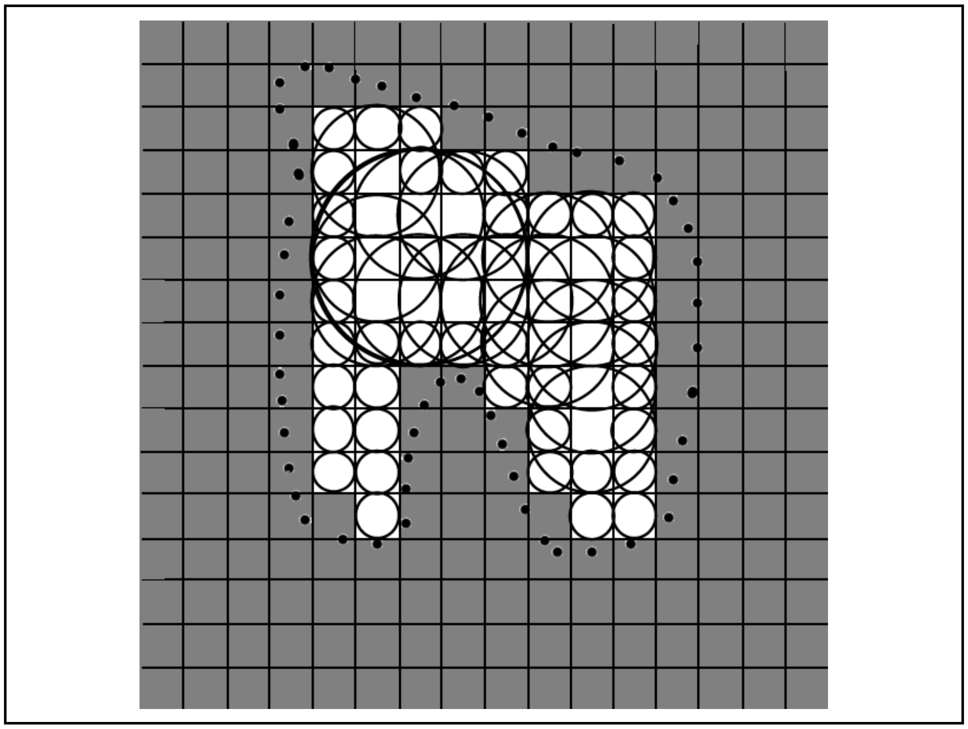
[Bittar, Tsingos, and Gascuel]

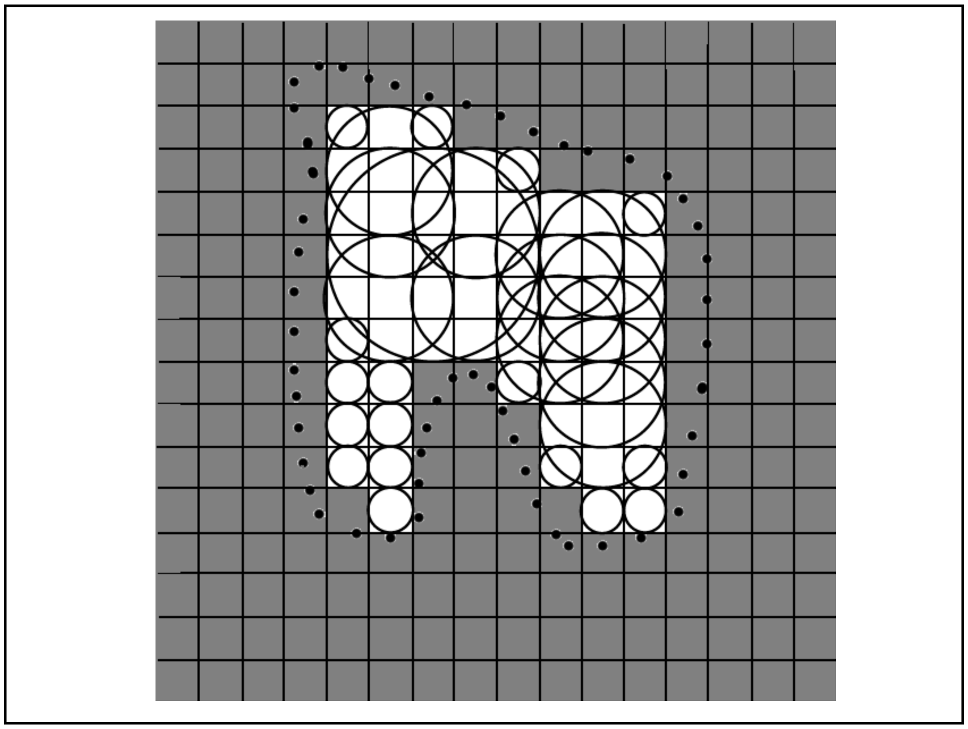


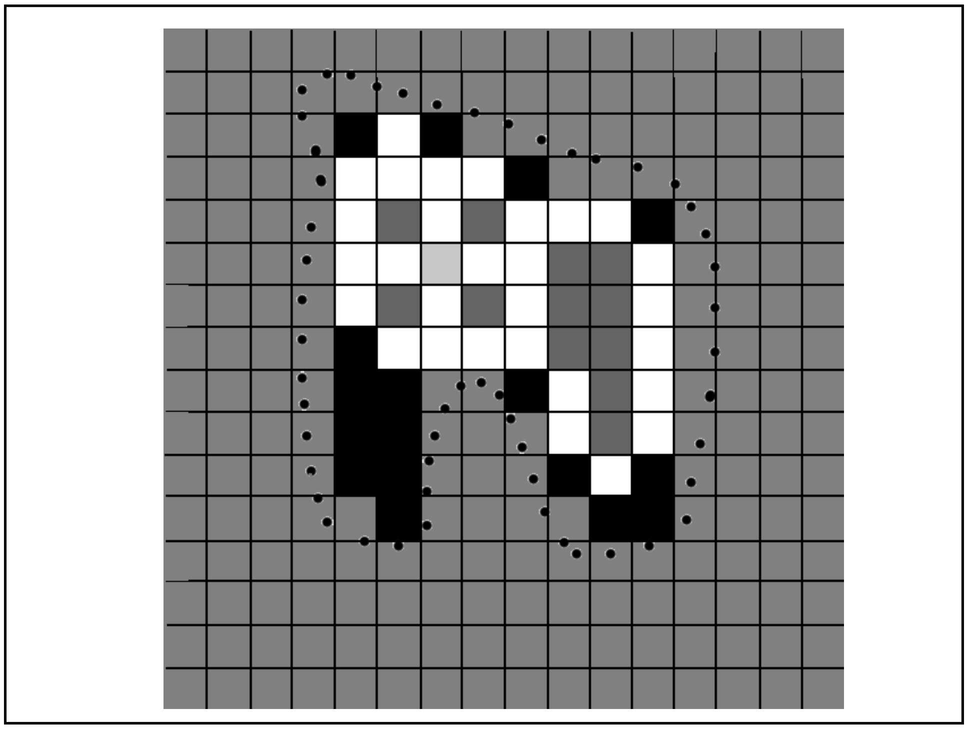
Potential Skeleton List Generation:

- Each interior voxel has a maximal radius.
- If the maximal sphere about the voxel is not contained in the maximal sphere about any other voxel, this voxel is added to the candidate list.



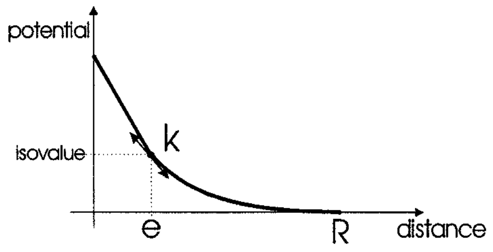






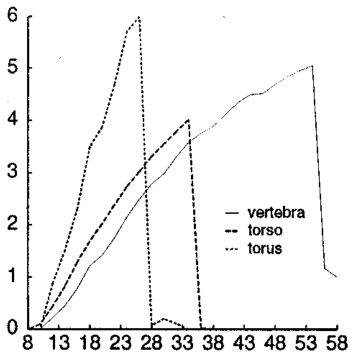
The field function used is:

$$f(r) = \begin{cases} -kr + ke + 1 & \text{if } r \in [0, e] \\ \dfrac{ke(e-R) + 3e - R}{(e-R)^3}(r-R)^2 & \text{if } r \in [e, R] \\ 0 & \text{otherwise} \end{cases}$$



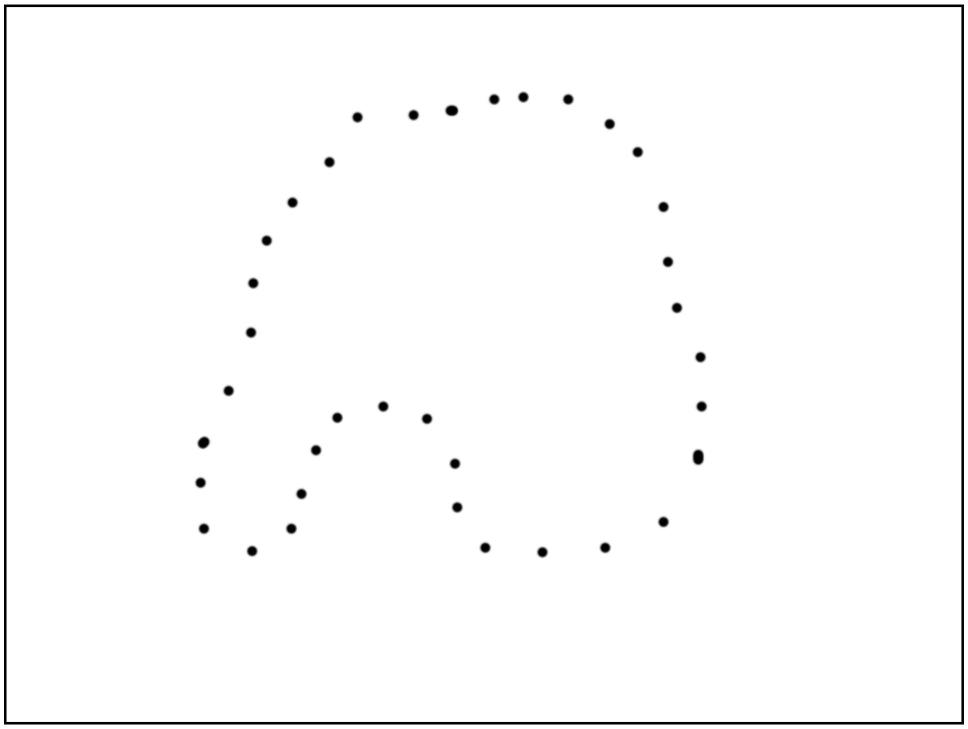
Resolution Specification:

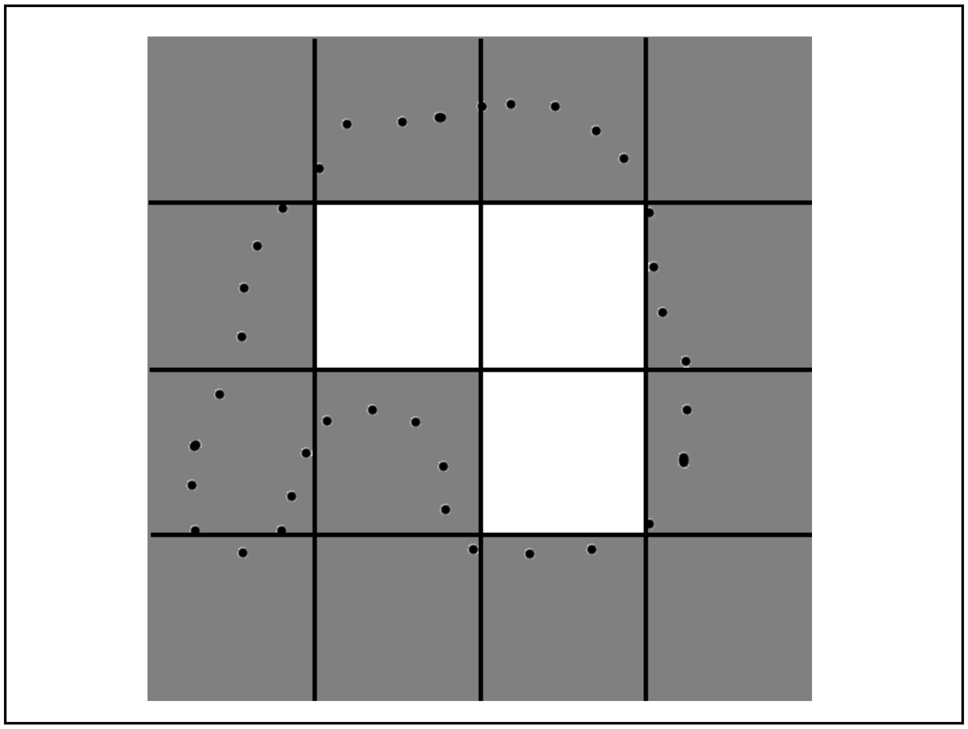
Let *g(p)* be the function giving the number of candidate skeleton points for a specified resolution:

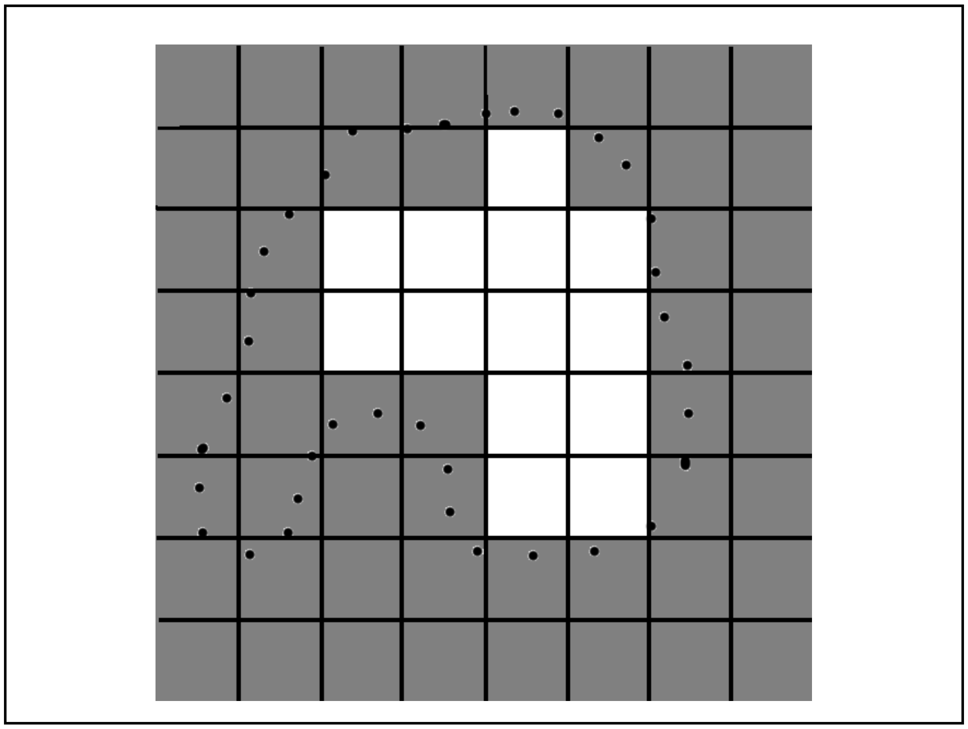


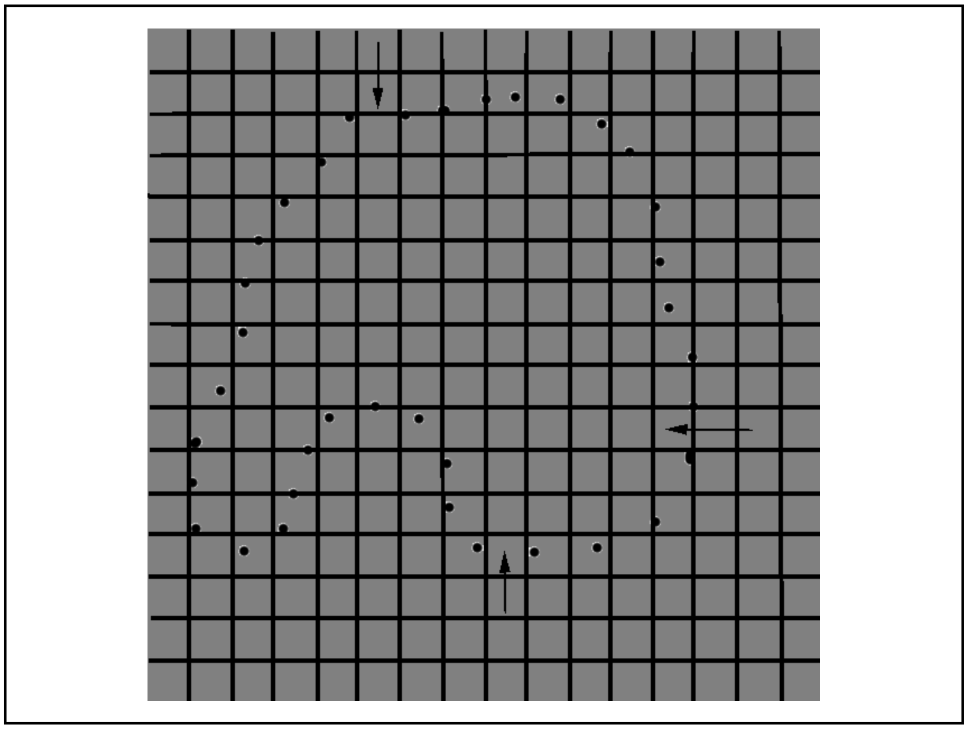
[Bittar, Tsingos, and Gascuel]

The resolution chosen is half the resolution at which this function attains its maximum.

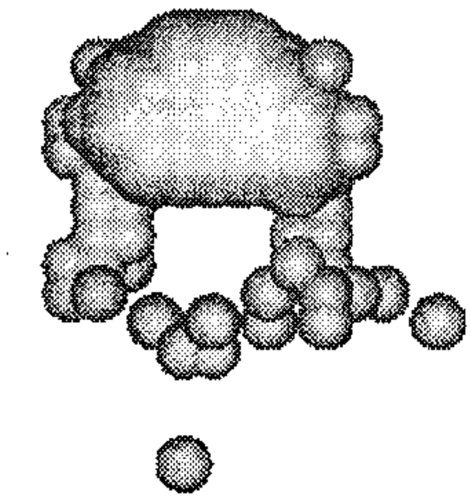






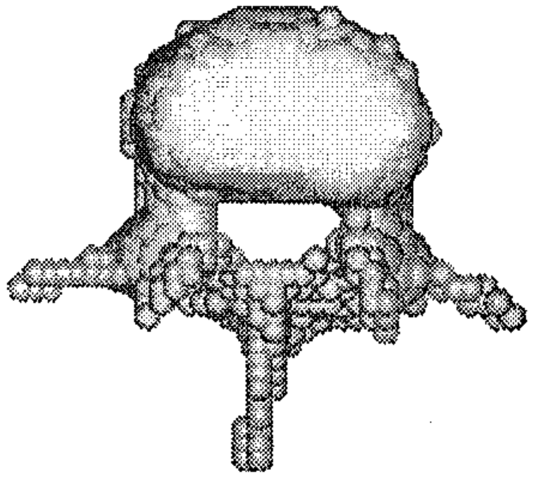


[Bittar, Tsingos, and Gascuel]



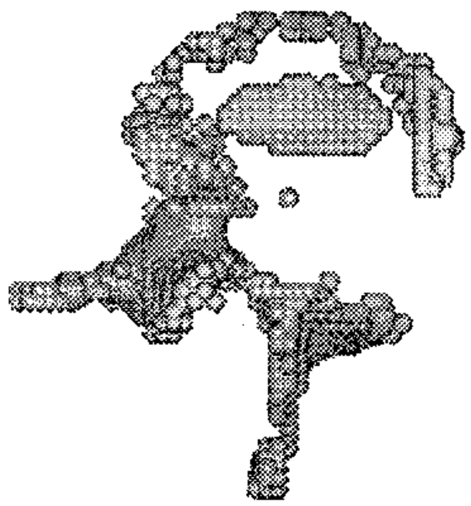
(a) 124 spheres

[Bittar, Tsingos, and Gascuel]



(b) 1050 spheres

[Bittar, Tsingos, and Gascuel]


(c) 1191 spheres

The Iterative Process:

Each potential skeleton point has a “select” flag associated to it. (Specifying if this point has been included in the final skeleton list.)

Each data point has a “select” flag associated to. (Specifying if this point has been fit in this step of the iteration.)

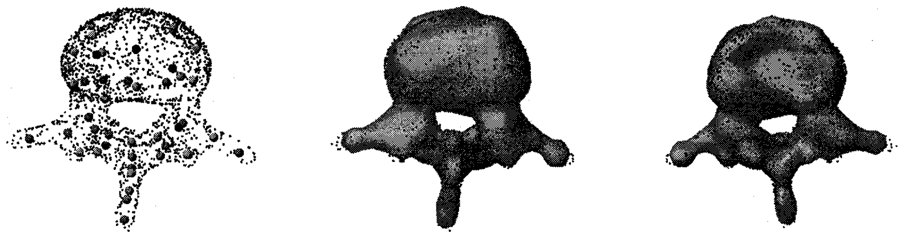
Upon each iteration:

- Set all of the data point “select” flags to “off”.
- While there are data points with “select” flag set to off:
  - Calculate the error term for each potential skeleton point with “select” flag set to off.
  - For potential skeleton point with highest error term:
    - Set the “select” flag for this potential skeleton point to “on”.
    - Add the the skeleton point and the associated function to the list
    - Set the “select” flag to “on” for all data points within the area of influence of the skeleton points.

- Holding the position of the newly added points constant, optimize their associated field function.
- Optimize all parameters of all variables.

Advantages:

- This method defines a quick way to select new primitives.
- More than one primitive can be added at each iteration resulting in substantially fewer iterations.
- Each skeleton points only contributes to the implicit function within a bounded region.



[Bittar, Tsingos and Gascuel]

object	number of data points	medial axis resolution	n. of medial axis spheres	number of skeletons	final Energy E	calculation time (sec)	number of passes
torus	4176	22	172	12	$5.46e^{-4}$	163	1
torso	2027	26	181	42	$3.78e^{-3}$	843	3
Y	871	14	16	10	$2.74e^{-3}$	194	1
vertebra	19837	42	1050				
	2037			46	$3.14e^{-2}$	3135	1

Reconstruction Methods

- Point Charge Model
  - Iteratively find new points by splitting old ones
  - Generate a list possible points and iteratively choose from the list
- **Local reconstruction of the distance function**

[Turk, O’Brien. *Variational Implicit Surfaces*]

[Yngve, Turk. *Creating Smooth Implicit Surfaces from Polygonal Meshes*]

Problem:

Given a mesh representation of a solid object, to define an implicit function describing the object.

Idea of the Solution:

To create a function which approximates the signed distance function near the surface of the object (and does not introduce zeros away from it) using radial basis functions and a linear term.

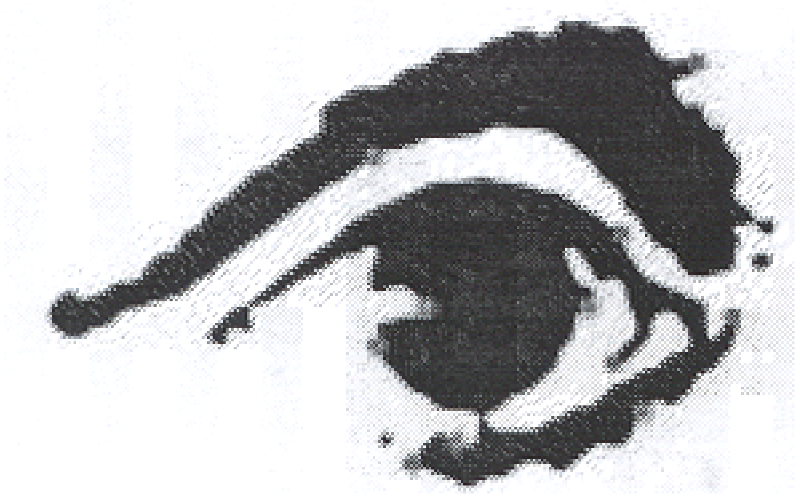
Upon each iteration, new points are added to the list of interpolated points with associated values. These points are:

- Points on the mesh which fall far from the implicit surface previously defined (with an associated zero value).
- Points on the implicit surface previously defined which fall from the triangle mesh (with an associated value equal to the signed distance of the point from the mesh).

Distance is measured as real distance divided by the radius of regularity to punish errors around regions of high curvature with greater severity.

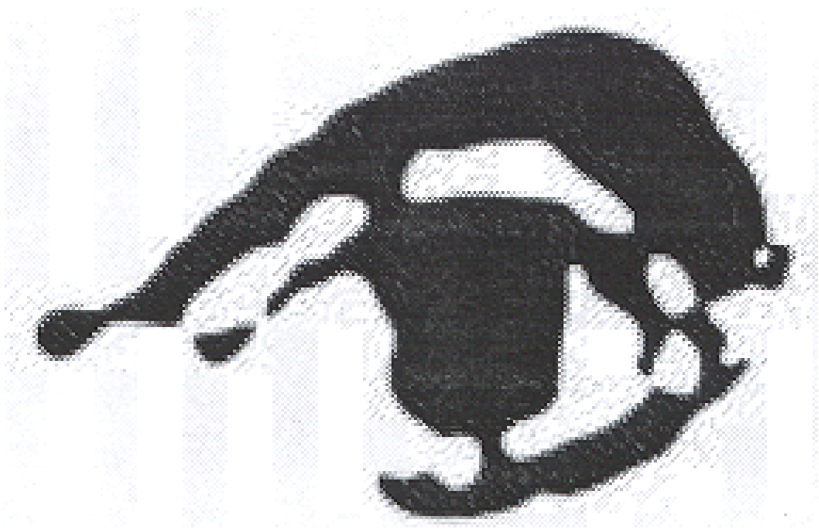
Original Surface

[Yngve and Turk]

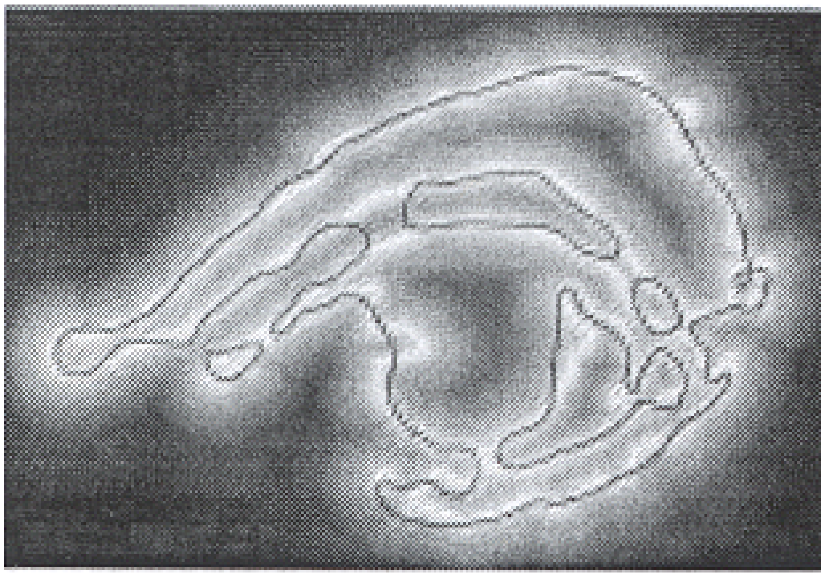


Implicit Surface

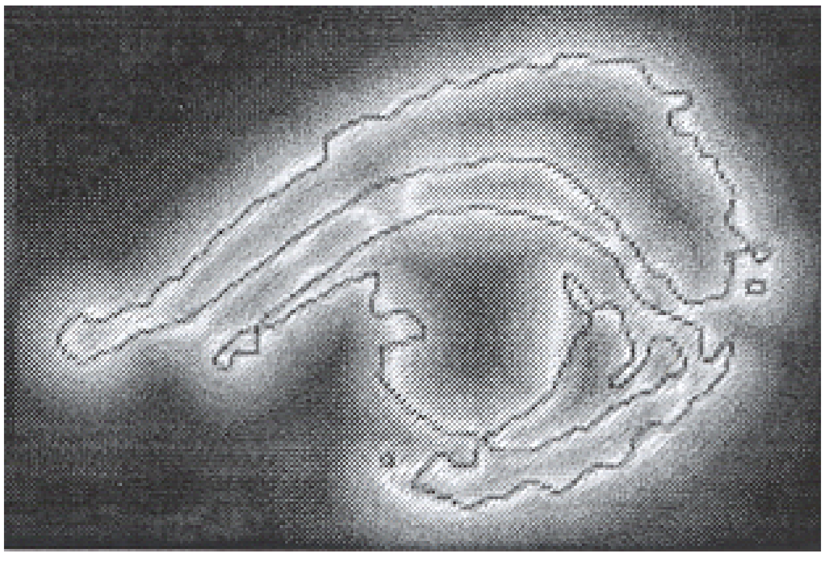
[Yngve and Turk]



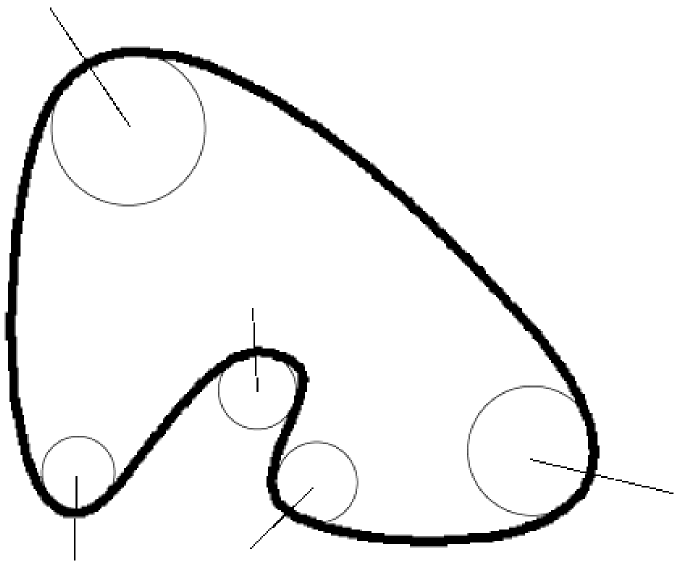
[Yngve and Turk]



[Yngve and Turk]



Radius of Regularity



Given a radial basis function Φ and a collection of points $\{q_i\}$ with associated values $\{c_i\}$, solve for values of $\{d_i\}$ and $\{p_0,p_1,p_2,p_3\}$ such that:

$$F(q) = \sum_j d_j \Phi(q - q_j) + p_0 + p_1 q_x + p_2 q_y + p_3 q_z$$

Setting:

$$\Phi_{ij} = \Phi(q_i - q_j)$$

this amounts to solving:

$$\begin{bmatrix} \Phi_{11} & \Phi_{12} & \cdots & \Phi_{1k} & 1 & q_1^x & q_1^y & q_1^z \\ \Phi_{21} & \Phi_{22} & \cdots & \Phi_{2k} & 1 & q_2^x & q_2^y & q_2^z \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots \\ \Phi_{k1} & \Phi_{2k} & \cdots & \Phi_{kk} & 1 & q_k^x & q_k^y & q_k^z \\ 1 & 1 & \cdots & 1 & 0 & 0 & 0 & 0 \\ q_1^x & q_2^x & \cdots & q_k^x & 0 & 0 & 0 & 0 \\ q_1^z & q_2^y & \cdots & q_k^y & 0 & 0 & 0 & 0 \\ q_1^z & q_2^z & \cdots & q_k^z & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_k \\ p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

It is possible to use pretty much any radial function. The authors choose one that guarantees minimization of overall curvature of the implicit surface.

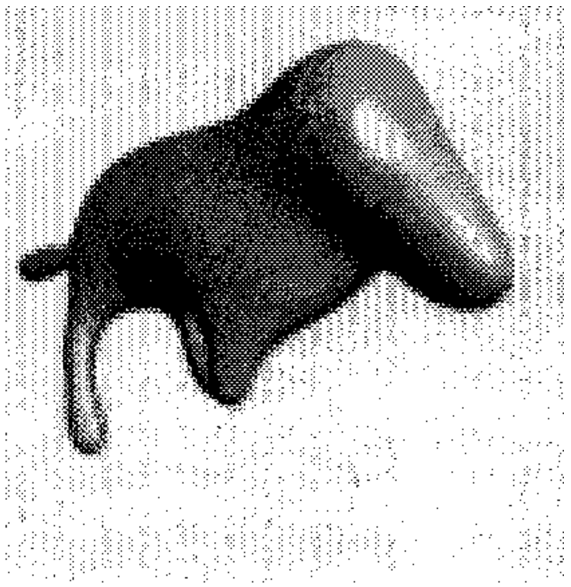
For the 2D case, this function is:

$$\Phi(x) = \|x\|^2 \log(\|x\|)$$

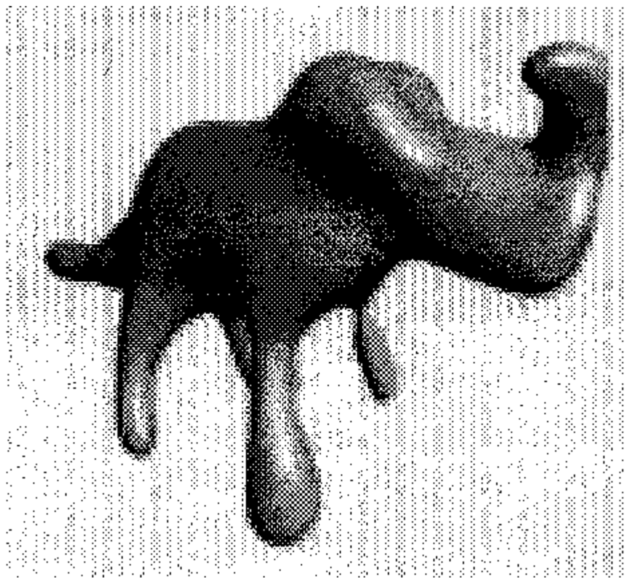
For the 3D case, this function is:

$$\Phi(x) = \|x\|^3$$

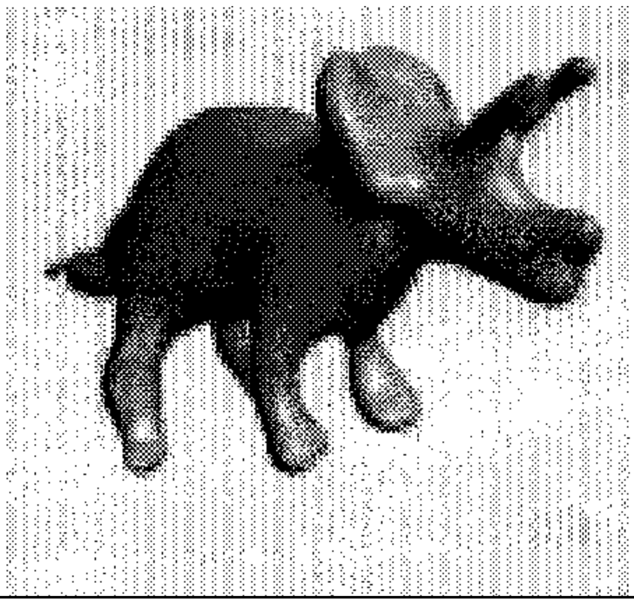
Initial attempt. [Yngve and Turk]



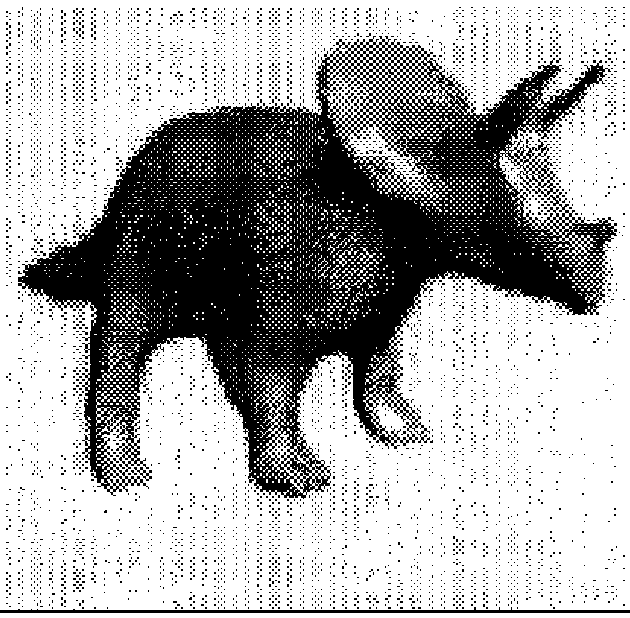
Fourth Iteration. [Yngve and Turk]



Tenth Iteration. [Yngve and Turk]



Eighteenth Iteration. [Yngve and Turk]



Number of polygons:	2834
Iterations to Finish:	22
Constraints (z/e/i) :	327/93/83
Total Time:	~83 minutes

Discussion

Muraki's Method:

- Very slow
- Field functions do not have compact support
- Initial starting point may be bad
+ Does not depend on resolution density
+ Gives a hierarchy of points (as only new points are optimized)
+ Inherently describes point distribution.

Bittar et al's Method:

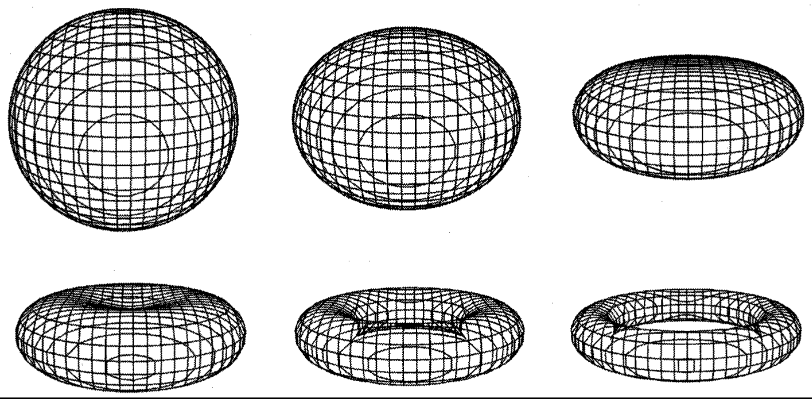
- Highly dependent on resolution density
- Does not give a hierarchy of points
+ Fast
+ Inherently describes point distribution

Turk et al's Method:

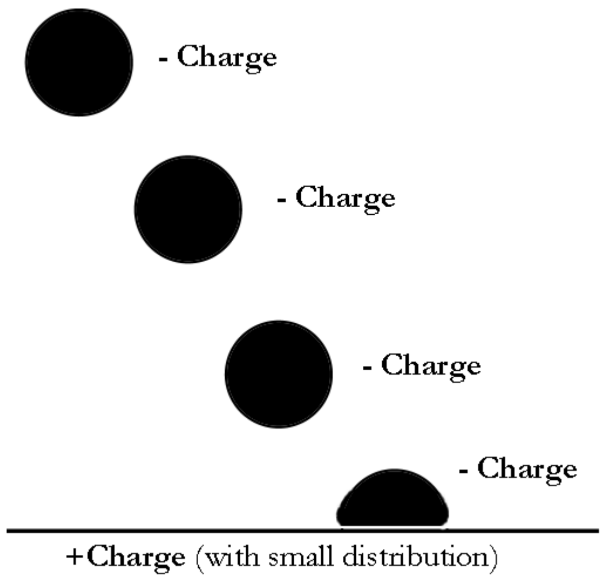
- Does not carry information about the points
- Very laborious (generating distance functions on each iteration)
+ Takes curvature into account
+ Seems capable of modeling rather complex surfaces.

Why should we care about implicit surfaces?

- They are natural objects to use for the description of surfaces via skeletons. (The skeleton of a surface being a simplification, it might be substantially easier to compare skeletons.)
- Allows for nice warping between objects of different topologies:



- Allows for nice simulation of physical properties:


- Charge
- Charge
- Charge
- Charge
+Charge (with small distribution)