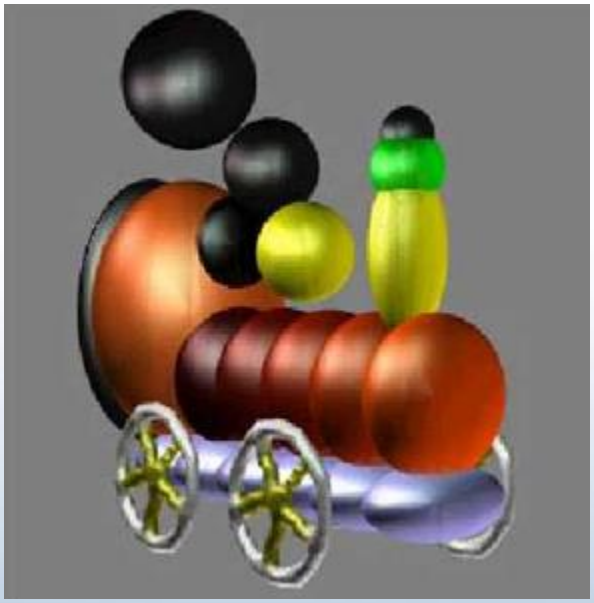
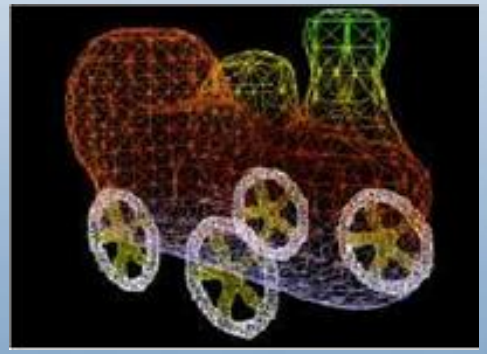


Skeletal Primitives in HyperFun



Images by Brian Wyvill

HyperFun Library

Primitives

Algebraic primitives:

hfSphere, hfEllipsoid, hfCylinder, hfEllCylinder, hfEllCone, hfTorus, hfSuperel, hfBlock

Skeletal objects:

hfBlobby, hfMetaball, hfSoft

Convolution objects:

hfConvPoint, hfConvLine, hfConvArc, hfConvTriangle, hfConvCurve, hfConvMesh

Procedural objects*:*

hfNoiseG

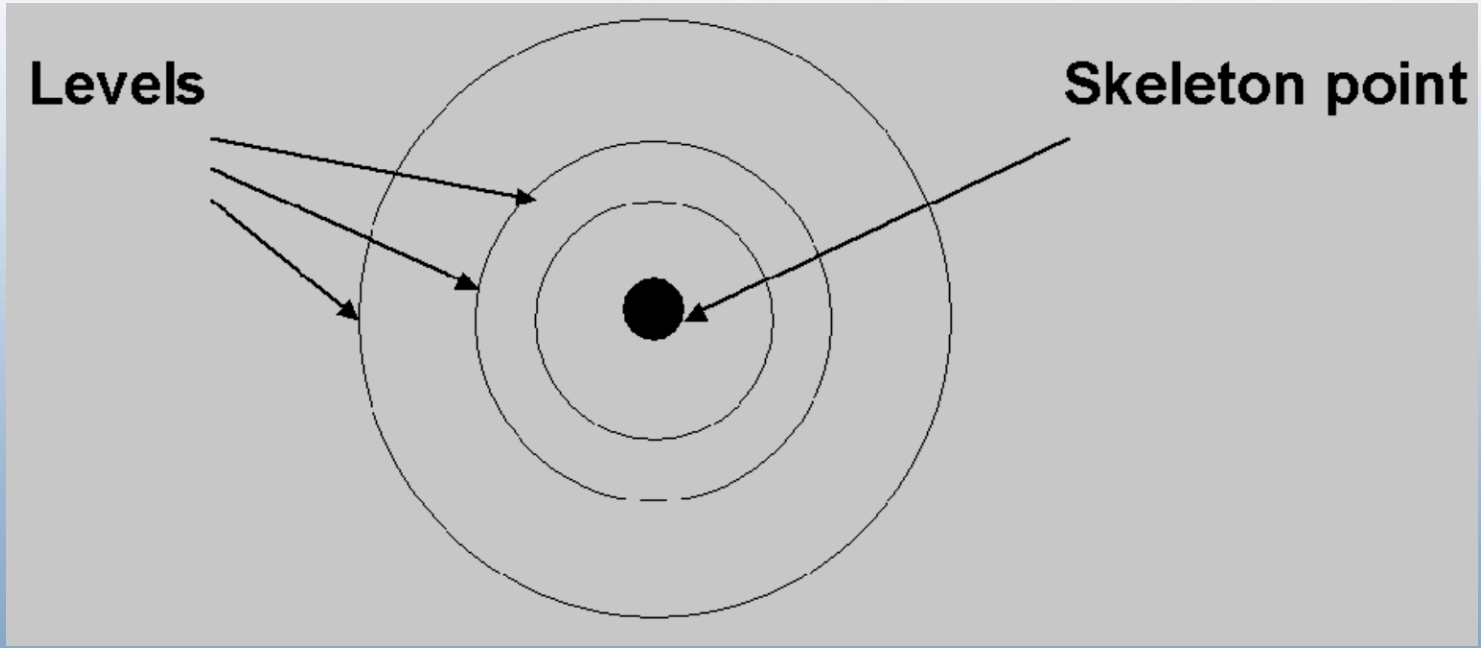
Operations

hfScale,
hfShift,
hfRotate,
hfTwist,
hfStretch,
hfTaper
hfBlendUni,
hfBlendInt

Skeletal Implicit Surfaces

Blinn [1982]:

modeling isosurfaces as a side effect of visualizing electron density fields



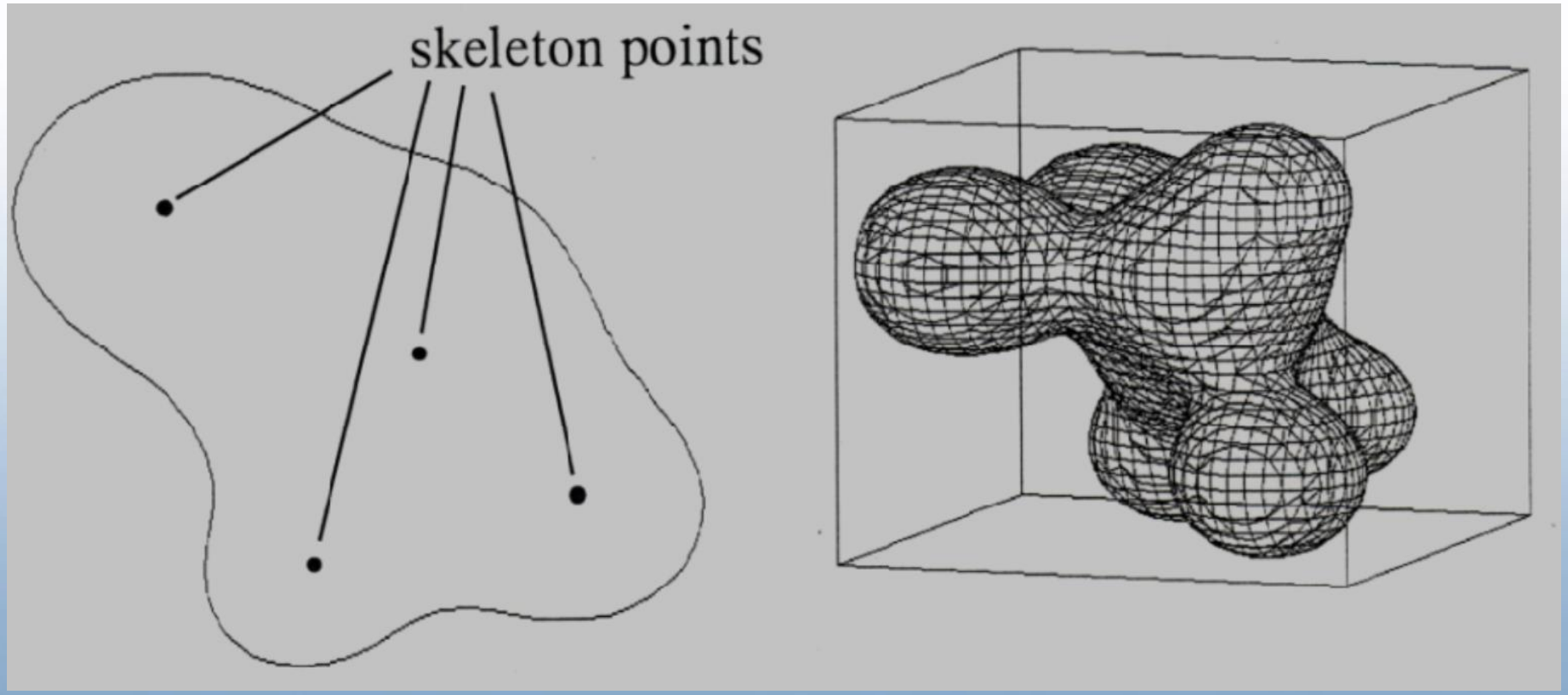
Skeletal model elements:

- *Skeleton* (points, lines and others);
- *Scalar field* with an individual skeleton element as a source;
- *Global field* as an *algebraic sum* of individual fields;
- *Level* (or *threshold*) of the field value defining the isosurface of interest.

Skeletal Implicit Surfaces

2D isoline

3D isosurface



Skeletal Surface Definition

$$F(P) - T = 0$$

with $$F(P) = \sum_{i=1}^{N} c_i F_i(r_i)$$

N is the number of skeletal elements,

F_i is the individual scalar field, (*blending function*) of the i-th element,

r_i is the distance from P to the i-th element,

T is the *threshold* (or *level value*).

Blobby Model

Blinn [1982]:

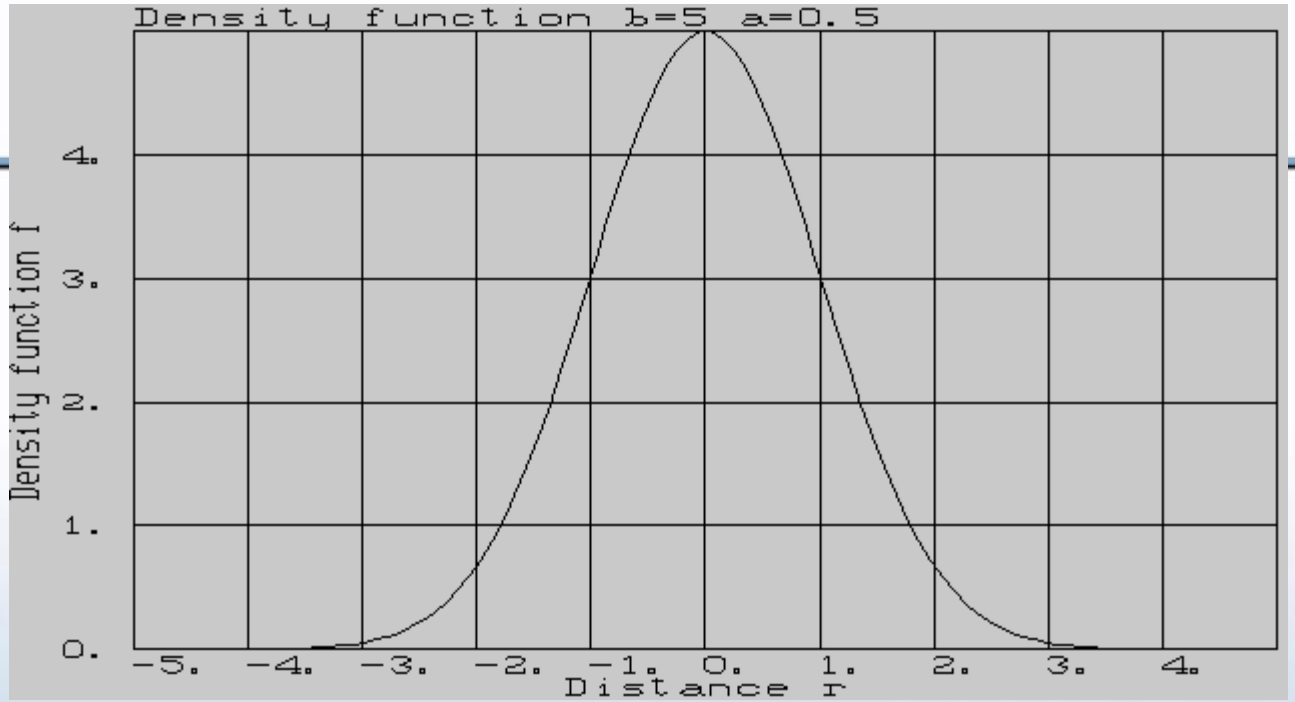
$$F_i(r_i) = b_i\, e^{-a_i r_i^2}$$

$$r_i = \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2}$$

(x,y,z) are coordinates of the given point P,

(x_i,y_i,z_i) are coordinates of the i-th skeleton point

Blobby model



Properties:

- Exponential field does not fall to zero;
- All elements contribute to the field in any point (global influence);
- Local influence introduced in Metaballs and Soft Objects

Skeletal primitives

HyperFun supports all three major models of skeletal surfaces

- **hfBlobby** - Blobby model has global influence of each skeletal point
- **hfMetaball** – Metaballs have local influence and flexible control
- **hfSoft** - Soft objects have local influence and limited control

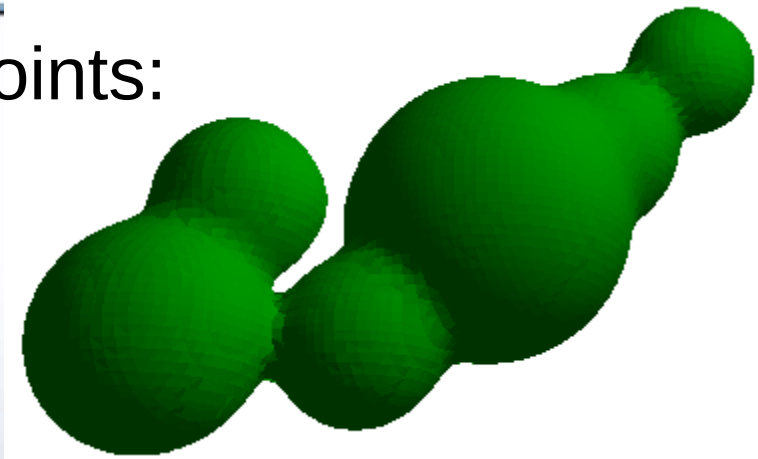
Library Primitive: Blobby Model

hfBlobby(x,x0,y0,z0,p,b,T)

x – given point coordinates for function evaluation;
x0, y0, z0 - arrays of blob skeletal points;
p - array of exponent coefficients for each blob; higher $\mathbf{p_i}$ means more narrow peak and smaller i-th blob;
b - array of blob weights; smaller $\mathbf{b_i}$ means smaller i-th blob;
T - threshold value for entire model; smaller **T** means expanded surface; bigger **T** means entire contracted surface;

Library primitive: Blobby model

Blobby model with six skeletal points:



```
blob(x[3], a[1])
{
array x0[6], y0[6], z0[6],
      p[6], b[6];
x0 = [-7., -6.0, -4., -2.0, 0., 1];
y0 = [-7, -4, -7, -4, -2, 0];
z0 = [-7, -6, -4, -2, 0, 2];
p = [0.7, 1, 1, 0.3, 1, 1];
b = [2, 1, 1, 1, 1, 0.5];
blob = hfBlobby(x,x0,y0,z0,p,b,0.04);
}
```

Library primitive: Metaballs

hfMetaball(x,x0,y0,z0,b,d,T)

x – given point coordinates for the function evaluation;

x0,y0,z0 - arrays of blob centers;

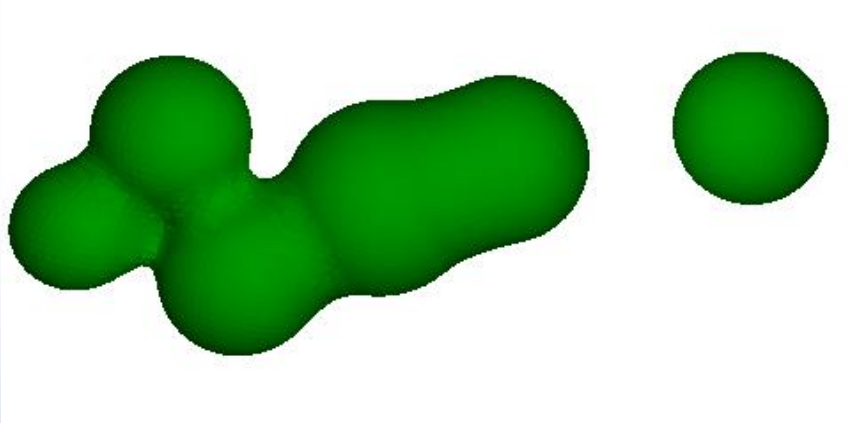
b - array of metaball weights; smaller $\mathbf{b_i}$ means smaller i-th blob;

d - array of d radii of influence for each metaball; higher d_i means bigger i-th metaball;

T - threshold value for the entire model; smaller **T** means expanded surface; bigger **T** means entire contracted surface;

Library primitive: Metaballs

Metaballs with
six skeletal points:



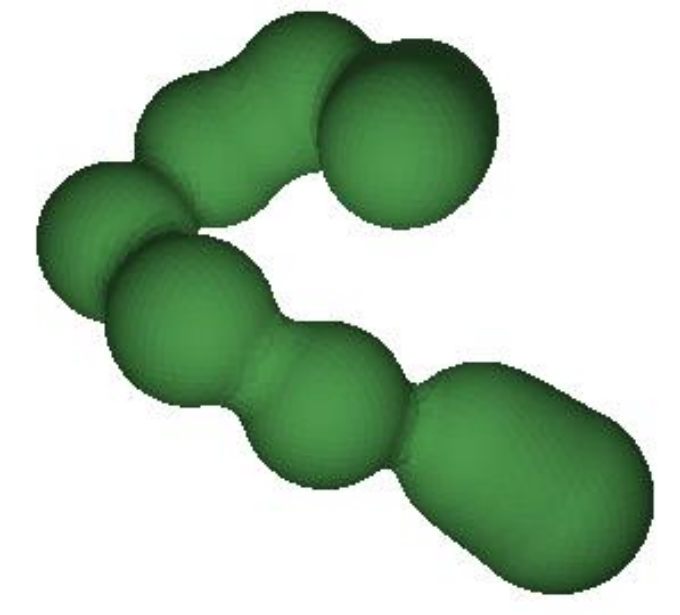
```
meta(x[3], a[1]){
array x0[6], y0[6], z0[6], d[6], b[6];
x0 = [-7., -6.0, -4., -2.0, 0., 6];
y0 = [-7, -4, -7, -4, -2, 1];
z0 = [-7, -6, -4, -2, 0, 4];
b = [1, 1, 1, 1, 1, 1];
d = [2.5, 3, 3.3, 4, 3.5, 3.5];
meta = hfMetaball(x,x0,y0,z0,b,d,0.18);
}
```

Library primitive: Soft Object

hfSoft(x,x0,y0,z0,d,T)

- **x** – given point coordinates for the function evaluation;
- **x0,y0,z0** - arrays of blob centers;
- **d** - array of d radii of influence; higher d_i means bigger i-th component;
- **T** - threshold value for the entire model; smaller **T** means expanded surface; bigger **T** means entire contracted surface

Library primitive: Soft object



Soft object with
nine skeletal points:

```
soft(x[3], a[1]) {
array x0[9], y0[9], z0[9], d[9];
x0 = [2.,1.4, -1.4, -3, -3, 0, 2.5, 5., 6.5];
y0 = [8, 8, 8, 6.5, 5, 4.5, 3, 2, 1];
z0 = [0, -1.4,-1.4, 0, 3, 4, 2.5, 0, -1];
d=[2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.7, 3];
soft = hfSoft(x,x0,y0,z0,d,0.2);
}
```