Universidade Federal do Rio Grande do Sul
Instituto de Informática
Programa de Pós-Graduação em Computação

Surface Reconstruction

June, 2005

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Model Representation

- Models can be stored/represented using different techniques
  - Polygon meshes
  - BRep
  - CSG
  - Implicit function

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Model Representation
Polygon Meshes

- Object is represented by a list of polygons

Copyright © Manuel M. Oliveira, Fausto R. Blanco

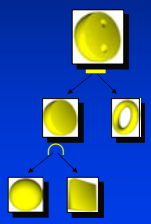
Model Representation
BRep

- Boundary representation – BRep
- Object is represented by its boundaries
- More general than triangle meshes
- Faces
  - can have arbitrary number of vertices
  - may be non convex
  - may be non planar
  - may contain holes

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Model Representation
CSG

- Object represented by a tree of operations on solid models



Copyright © Manuel M. Oliveira, Fausto R. Blanco

Model Representation
Implicit function

- Object is represented by its mathematical function
  - f(x,y,z)=0
- Simple operations
  - Morphing f to g: (1-t)f(x,y,z)+tg(x,y,z)
  - CSG
  - Blending
  - Twisting


$x^2+y^2+z^2-r=0$


$(x^2+y^2+z^2+R^2-r^2)^2-4R^2(x^2+y^2)=0$


$(2x^2+y^2+z^2-1)^3-(0.1x^2+y^2)z^3=0$

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
3D scanners

- 3D Scanners are becoming popular and cheaper
- Useful for building virtual models of complex objects
- Problems
  - Discrete representation
    - Point-cloud
  - Noisy data

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
The problem

- Given a set of sample points, reproduce the original model
- Models can be complex
  - Sharp features
  - Holes
  - Millions of points

[Carr01]

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
Known Techniques

- Computational Geometry
  - Alpha Shapes
  - Crust
  - Cocone
- Algebraic Methods
- Implicit Function
  - RBF Based
  - Multilevel Partition of Unity

Copyright © Manuel M. Oliveira, Fausto R. Blanco

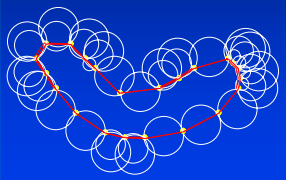
Surface Reconstruction
Alpha Shapes

- Connect two points if there's a circle containing the points
- Radio of the circles is 1/alpha
- Each value of alpha define a possibly different surface

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
Alpha Shapes

- Alpha Shapes in action



- Surface is not smooth
- Alpha value must be manually adjusted !

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
RBF Based - The Linear System

- We want a function that gives the distance from any point to the reconstructed surface
  - Use the distance from the points to all other points in the point cloud
  - When applied to a sampled point, the function result must be zero (the point is on the surface)
  - All points which result is zero are surface points!

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

RBF Based- The Linear System

- Let ϕ be the distance function (eg.: Euclidean)
- ϕ is called "the basic function"

$$\begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} & \phi_{14} \\ \phi_{21} & \phi_{22} & \phi_{23} & \phi_{24} \\ \phi_{31} & \phi_{32} & \phi_{33} & \phi_{34} \\ \phi_{41} & \phi_{42} & \phi_{43} & \phi_{44} \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

RBF Based- The Linear System

- Generalizing

$$\begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1n} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2n} \\ \phi_{31} & \phi_{32} & \cdots & \phi_{3n} \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{n1} & \phi_{n2} & \cdots & \phi_{nn} \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \cdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \cdots \\ 0 \end{bmatrix}$$

This gives us the trivial solution $\lambda_i=0$, i=1..n

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

RBF Based- The Linear System

- We must add new ϕ values that gives non-zero results
  - Add off surface points

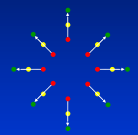
$$\begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1n} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2n} \\ \phi_{31} & \phi_{32} & \cdots & \phi_{3n} \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{n1} & \phi_{n2} & \cdots & \phi_{nn} \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \cdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ \cdots \\ h_n \end{bmatrix}$$

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

RBF Based- The Linear System

- Generate off-surface points using normals



$$\begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1n} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2n} \\ \phi_{31} & \phi_{32} & \cdots & \phi_{3n} \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{n1} & \phi_{n2} & \cdots & \phi_{nn} \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \cdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ \cdots \\ h_n \end{bmatrix}$$

- f(x,y,z)= 0
- f(x,y,z)= 1
- f(x,y,z)= -1

$f(x,y,z) = h$

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

RBF Based- The Linear System

- To ensure positive-definiteness of the solution
  - Add a degree one polynomial
    - $p_x x+p_y y+p_z z+p_e=0$
      - p_x, p_y, p_z, p_e are the polynomial coefficients
      - x,y,z are point coordinates
- Accounts for linear and constant portions of *f*

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

RBF Based- The Linear System

- Modify matrix to accomplish the polynomial

$$\begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1n} \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2n} \\ \phi_{11} & \phi_{12} & \cdots & \phi_{1n} \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{n1} & \phi_{n2} & \cdots & \phi_{nn} \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \cdots \\ \lambda_n \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ \cdots \\ h_n \end{bmatrix}$$

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

RBF Based- The Linear System

- Modify matrix to accomplish the polynomial

$$\begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1n} & x_1 & y_1 & z_1 & 1 \\ \phi_{21} & \phi_{22} & \cdots & \phi_{2n} & x_2 & y_2 & z_2 & 1 \\ \phi_{31} & \phi_{32} & \cdots & \phi_{3n} & x_3 & y_3 & z_3 & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \phi_{n1} & \phi_{n2} & \cdots & \phi_{nn} & x_n & y_n & z_n & 1 \\ x_1 & x_2 & \cdots & x_n & 0 & 0 & 0 & 0 \\ y_1 & y_2 & \cdots & y_n & 0 & 0 & 0 & 0 \\ z_1 & z_2 & \cdots & z_n & 0 & 0 & 0 & 0 \\ 1 & 1 & \cdots & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \cdots \\ \lambda_n \\ p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ \cdots \\ h_n \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

RBF Based- The Linear System

- Writing it simpler

$$\begin{bmatrix} A & P \\ P^T & 0 \end{bmatrix} \cdot \begin{bmatrix} \lambda \\ p \end{bmatrix} = \begin{bmatrix} h \\ 0 \end{bmatrix}$$

- Solve the system
  - Cholesky factorization
  - GMRES iterative method
  - LU factorization

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

RBF Based- Algorithm

- Assume that each point has an associated normal
- Generate extra surface points along normals
- Solve linear system

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

Normal Estimation

The estimation algorithm:

- For every point
  - Find k nearest points
  - Calculate the tangent plane
  - Get plane normal and associate it to the point

- Estimated normals can be in different senses

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

Principal Component Analysis

- Calculate the tangent plane
  - Use the point we want to estimate the normal vector and its k-nearest neighbors
  - Create a 3x3 covariance matrix

$$\text{Covariance Matrix} = \begin{bmatrix} \text{cov}(x,x) & \text{cov}(x,y) & \text{cov}(x,z) \\ \text{cov}(x,y) & \text{cov}(y,y) & \text{cov}(y,z) \\ \text{cov}(z,x) & \text{cov}(z,y) & \text{cov}(z,z) \end{bmatrix}$$

where $\text{cov}(a,b) = \dfrac{\sum_{i=1}^{n}(a_i - \bar{a})(b_i - \bar{b})}{n-1}$

- Get plane normal
  - Extract the eigenvector of the covariance matrix

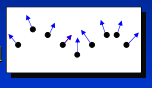
Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction

Normal Propagation

The propagation algorithm:

- Get an arbitrary point p and assume that its normal is on the correct sense
- While there's a normal sense not adjusted



  - For each one of the nearest neighbors n of p
    - If the angle between p normal and n normal > 90

      - Invert normal sense
    - Mark the normal sense as adjusted

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
RBF Based- Algorithm

- Estimate Normals
  - Find k nearest neighbors
  - Find a tangent plane on these points (PCA)
    - Get plane normal (eigenvector)
  - Propagate normals senses
- Generate extra surface points along normals
- Solve linear system

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
RBF Based – The Linear System

- Problem:
  - The linear system can be very large
  - This may not be solvable in an useful time
- Solution:
  - Partition of Unity
    - Space partition using overlapped regions
    - Weight function

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
RBF Based – Partition of Unity

- Create overlapping regions
  - Each region has a limited number of points
  - Each region has a weight function
- Solve a system for each region

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
RBF Based – Partition of Unity

- The global solution is:
  - $F(p) = \sum f_i(p) w_i(p)$, for each i region that contains the point p
    - f_i: local system solution for region i
    - w_i: weight function of the region i
    - p: x,y,z point coordinates
- To ensure $\sum w_i = 1$
  - $w_i(p) = \frac{W_i(p)}{\sum_j W_j(p)}$

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
RBF Based – Partition of Unity

- $w_i(x,y,z) =$

$$\frac{2(x-s_i.x)(t_i.x-x)}{(t_i.x-s_i.x)^2} \frac{2(y-s_i.y)(t_i.y-y)}{(t_i.y-s_i.y)^2} \frac{2(z-s_i.z)(t_i.z-z)}{(t_i.z-s_i.z)^2}$$

- s_i: lower limit of region i
- t_i: higher limit of region i

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
RBF Based- Algorithm

- Estimate Normals
  - Find k-nearest neighbors
  - Find a tangent plane on these points (PCA)
    - Get plane normal (eigenvector)
  - Propagate normals senses
- Generate extra surface points along normals
- Subdivide space in overlapped regions
- Solve linear system for each region
- Global solution is obtained considering the weight functions

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
Visualization

- Once we got the implicit function of the model, how can we visualize it ?
  - Create a polygon mesh
  - Draw the polygons
- Marching Cubes

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Iso-Surface Extraction
Marching Cubes

- Divide space in a set of cubes
- Evaluate the function on the vertices of each cube
- Create a polygon mesh based on the results of the evaluation

Copyright © Manuel M. Oliveira, Fausto R. Blanco

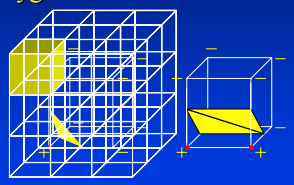
Iso-Surface Extraction
Marching Cubes

- Divide space in a set of cubes

Copyright © Manuel M. Oliveira, Fausto R. Blanco

Iso-Surface Extraction
Marching Cubes

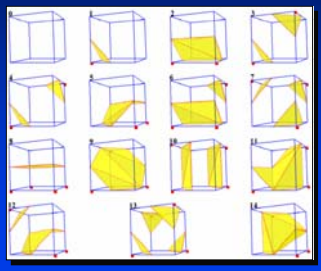
- Evaluate function on each vertex
- Create a polygon mesh



Copyright © Manuel M. Oliveira, Fausto R. Blanco

Iso-Surface Extraction
Marching Cubes

- All cases



Copyright © Manuel M. Oliveira, Fausto R. Blanco

Surface Reconstruction
RBF Based – Algorithm: Summary

- Estimate Normals
  - Find k-nearest neighbors
  - Find a tangent plane on these points (PCA)
    - Get plane normal (eigenvector)
  - Propagate normals senses
- Generate extra surface points along normals
- Subdivide space in overlapped regions
- Solve linear system for each region
- Global solution is obtained using weight functions
- Polygonize with marching cubes

Copyright © Manuel M. Oliveira, Fausto R. Blanco