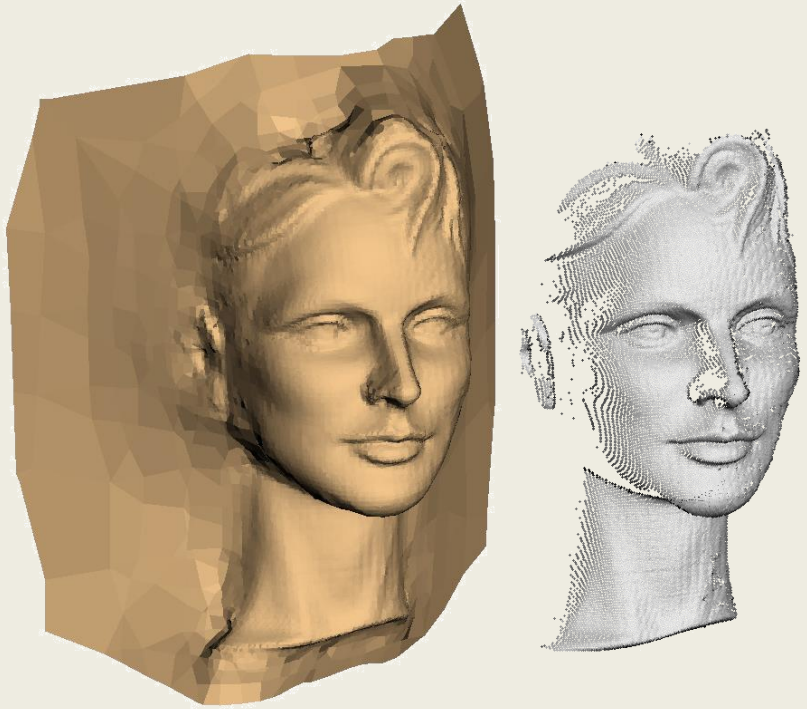

SSD: Smooth Signed Distance Surface Reconstruction

F. Calakli and G. Taubin, 2011

Presented by L. Fernandez

General Approaches

- Interpolating polygon meshes
  - Amenta et al. 1998
  - Bernardini et al. 1999
- Implicit function fitting
  - Hoppe et al. 1992
  - Curlesss et al. 1996
  - Ohtake et al. 2004
  - Turk et al. 2004
  - Shen et al. 2004
  - ***Kazhdan et al. 2006***

Method Overview

- Input: Oriented point cloud
- Method: Implicit surface representation
  - Implicit function = Smooth approximation of signed distance function
- Output: Watertight, adaptive manifold surface

Implicit Reconstruction

- Given: Oriented point set

  $D = \{\ldots, (p_i, n_i), \ldots\}$ sampled from surface S

- Compute implicit surface

  $S = \{x \mid f(x) = 0\}$

  with ***interpolating conditions***

  $f(p_i) = 0$ and $\nabla f(p_i) = n_i$ for all samples

Approximating Scheme

- Interpolating conditions

  $\mathbf{f(p_i) = 0}$ and $\mathbf{\nabla f(p_i) = n_i}$ for all samples

  - Vulnerable to noise
  - Needs parametric families of functions with many degrees of freedom

- Solution: Least Squares Energy (local)

$$E_D(f) = \lambda_0 \sum_{i=1}^{N} \boldsymbol{f(p_i)}^2 + \lambda_1 \sum_{i=1}^{N} \|\boldsymbol{\nabla f(p_i) - n_i}\|^2$$

Minimizing Data Energy

$$E_D(f) = \lambda_0 E_{D0}(f) + \lambda_1 E_{D1}(f)$$

$$E_{D0}(f) = \lambda_0 \sum_{i=1}^{N} \boldsymbol{f}(\boldsymbol{p_i})^2, \qquad E_{D1}(f) = \lambda_1 \sum_{i=1}^{N} \|\nabla \boldsymbol{f}(\boldsymbol{p_i}) - \boldsymbol{n_i}\|^2$$

- Defines energy ***near*** data points {…, (p_i, n_i), …}
- Minimize to approximate signed distanced function

Defining non-local behavior

$$E(f) = E_D(f) + \boldsymbol{\lambda_2 E_R(f)}$$

$$E_R(f) = \frac{1}{\|V\|} \int_V \|\boldsymbol{Hf}(\boldsymbol{x})\|^{\boldsymbol{2}} dx$$

$$Hf(x) = \left[\frac{\partial \nabla f(x)}{\partial x_1} \quad \frac{\partial \nabla f(x)}{\partial x_2} \quad \frac{\partial \nabla f(x)}{\partial x_3} \right]$$

- Defines energy ***far from*** data points
- Tends to make $\boldsymbol{\nabla f(p_i)}$ constant

Linear Families of Functions

Why linear functions?

- Most have a unique solution
- Can be computed by solving a system of linear equations

Linear Families of Functions

$$f(x) = \sum_{a \in \Lambda} f_\alpha \phi_\alpha(x) = \Phi(x)^t F$$

Λ = set of K elements

F = K-dimensional coefficient vector

K = chosen number of basis functions

Popular ***smooth basis functions***

- RBFs (Carr et al. 2001)
- Compactly supported BFs (Ohtake et al. 2004)
- Trigonometric polynomials (Kazhdan et al. 2005)
- Basis splines (Kazhdan et al. 2006)
- Wavelets (Manson et al. 2008)

Choosing a function set

- Small K
  - Example: polynomial basis functions
  - Straightforward solution: AF = b
  - Likely to get low quality results
- Large K
  - Example: RBFs, trigonometric polynomials
  - Non-compact support → Dense A matrix
  - Compact support → Sparse A matrix
    - May still be complicated to compute regularization term
    - **Solution: Hybrid FE/DF discretization with discontinuous gradient**

Independent Discretization

- Hybrid FE/FD discretization
  - Trilinear interpolant for f(x)
  - Finite differences for ∇f(x)
  - Finite differences for Hf(x)
- Non-homogenous quadratic energy
  - If f(x), ∇f(x) and Hf(x) can be written as homogeneous linear functions of parameters F

$$E(F) = F^t AF - 2b^t F + c$$

- Global minimum

$$AF = b$$

Relation to Poisson Reconstruction

- Both implement implicit reconstruction with interpolating conditions
- Poisson Reconstruction: Implicit function is recovered by integration over the vector field

$$\nabla f(x) = v$$

Relation to Poisson Reconstruction

Why SSD?

- For $\nabla f(x) = v$ to be true need to impose constraint that integral over v is zero for all closed curves
- SSD offering: Solve for f(x) in a single step by minimizing with respect to f(x)

$$\int_V \|\nabla \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{v}(\boldsymbol{x})\|^2 dx$$

Which is equivalent to solving the Poisson equation

$$\Delta \boldsymbol{f}(\boldsymbol{x}) = \nabla \boldsymbol{v}(\boldsymbol{x})$$

Relation to Poisson Reconstruction

SSD Vector field formulation: Minimization of vector field energy

$$\frac{\lambda_1}{N}\sum_{i=1}^{N}\|v(p_i)-n_i\|^2+\frac{\lambda_2}{|V|}\int_V\|Dv(x)\|^2\,dx$$

Dv(x) = Jacobian of vector field v(x)

Given $\nabla f(x) = v$, follows that $Hf(x) = Dv(x)$

Implementation

- Iterative solver
  - Cascading multi-grid approach
    - Solve problem on coarser level
    - Use previous solution to initialize solution at next level
- Iso-surface extraction
  - Dual marching cubes (Schaefer 2005)
    - Crack-free

Complexity

- Iterative solver (Conjugate gradients)
  - Cost/iteration: O(N)
    - N by N system
    - Number of non-zero entries per row is constant
  - Number of iterations: O(N^0.5)
  - Overall: O(N^1.5)
- Hierarchical solver (Multigrid)
  - Cost/(hierarchy level): O(1)
  - Overall: O(N)

Complexity

- Empirically: Super linear growth
- From “Screened Poisson Surface Reconstruction” by Kazhdan and Hoppe 2013

		Time in seconds				Memory in MB				Vertices $\times 10^6$			
Model	Depth	Poisson	Wavelet	SSD	Screened	Poisson	Wavelet	SSD	Screened	Poisson	Wavelet	SSD	Screened
Neptune	8	10 [13]	3	275	14	113	4	238	133	0.1	0.1	0.1	0.1
	9	25 [17]	4	547	20	149	11	455	269	0.2	0.2	0.2	0.2
	10	89 [36]	6	3302	44	422	35	1247	604	0.9	0.7	0.9	0.9
	11	320 [105]	9	15441	126	1387	118	3495	1622	3.1	1.5	2.9	3.2
David	8	41 [45]	9	492	48	427	11	863	454	0.2	0.2	0.2	0.2
	9	108 [66]	12	2355	73	510	38	1724	932	0.8	0.7	0.8	0.9
	10	412 [157]	20	19158	182	1498	151	4895	2194	3.4	2.8	3.3	3.5
	11	1710 [522]	43	†119119	609	5318	545	>8192	6188	12.8	7.3	11.6	13.3

Table I. : Runtime performance of the different reconstruction techniques on the David and Neptune datasets at depths 8, 9, 10, and 11. The numbers in brackets are timing results obtained using our method without screening ($\alpha = 0$) i.e., after the algorithmic improvements of Section 5.
†Since the memory usage of SSD exceeds the available RAM, we report the CPU user time for this experiment rather than the wall-clock time.

Results: Inaccurate Normal Data

Figure 5: *Input point cloud (leftmost) of David's head, and reconstructions using (from second-left to right) MPU, Poisson, D4 Wavelets, and our SSD reconstruction.*

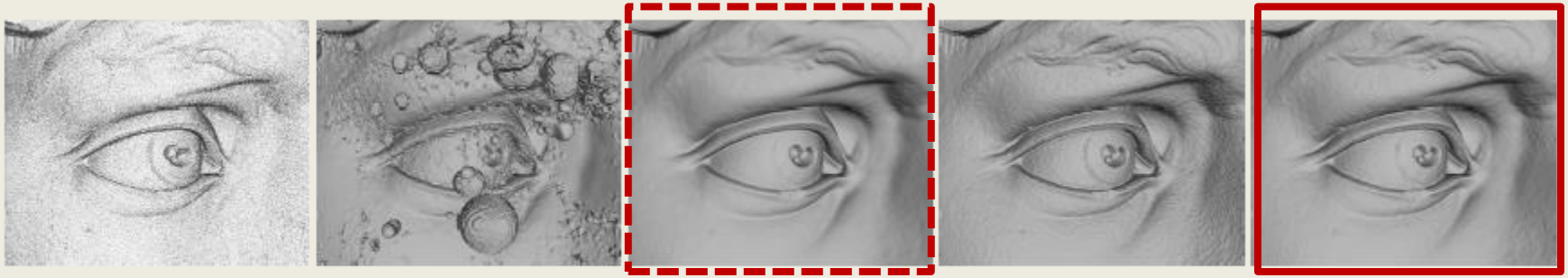

Figure 6: *Input point cloud (leftmost) of David's eye, and reconstructions using (from second-left to right) MPU, Poisson, D4 Wavelets, and our SSD reconstruction.*

Results: Non-uniform sampling

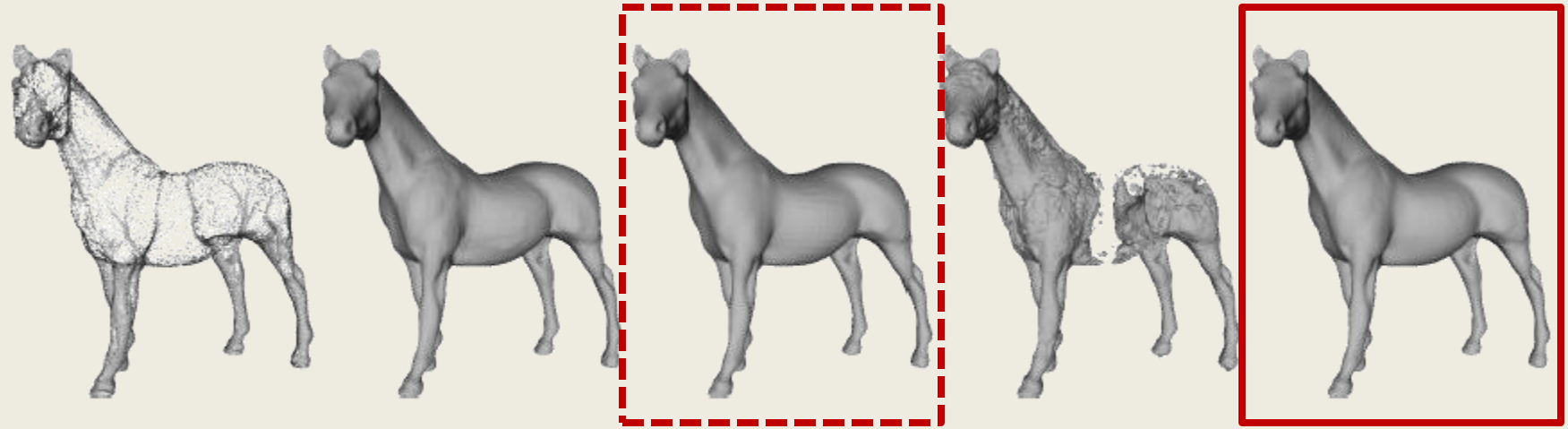

Figure 7: *Input point cloud (leftmost) of the Horse model, and reconstructions using (from second-left to right) MPU, Poisson, D4 Wavelets, and our SSD reconstruction.*

Figure 8: *Input point cloud (leftmost) of the Chiquita model, and reconstructions using (from second-left to right) MPU, Poisson, D4 Wavelets, and our SSD reconstruction.*

Results

Method	Time (Sec)	Memory (MB)	Polygons
MPU	27	148	378925
Poisson	43	283	319989
D4	17	63	365974
SSD	72	264	346652

Table 1: *The running time, the peak memory usage, and the number of triangles in the reconstructed surface of the David's head generated with different methods.*

Model	MPU	Poisson	D4	SSD
Armadillo	1.0000	0.4617	0.2383	0.3514
Dragon	0.8779	1.0000	0.5301	0.6810
Horse	0.0752	0.0827	1.0000	0.0551
Igea	1.0000	0.7761	0.5701	0.4018

Table 2: *Hausdorff distance between real surfaces and reconstructed surfaces. Each row is normalized by the maximum error to provide relative comparison (lower is better).*

References

- Slides from Fatih Calakli's talk at Pacific Graphics 2011
- Calakli and Taubin, SSD: Smooth Signed Distance Surface Reconstruction. (2011)