Reconstruction of Solid Models from Oriented Point Sets

Misha Kazhdan

Reconstruction of Solid Models from Oriented Points Sets. Kazhdan (2005)
Poisson Surface Reconstruction. Kazhdan, et al. (2006)
Multilevel Streaming for Out-of-Core Surface Reconstruction. Bolitho et al. (2007)

Shape Spectrum

There are many ways to represent a shape:

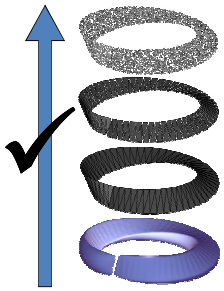
- Point Set
- Polygon Soup
- Polygonal Mesh
- Solid Model

Equivalence of Representations

There are many ways to represent a shape:

- Point Set
- Polygon Soup
- Polygonal Mesh
- Solid Model

In one direction, the transition between representations is straight-forward.

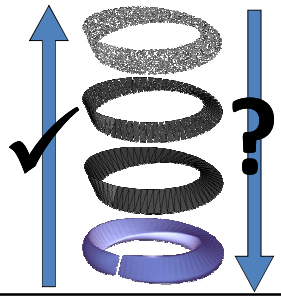


Equivalence of Representations

There are many ways to represent a shape:

- Point Set
- Polygon Soup
- Polygonal Mesh
- Solid Model

The challenge is to transition in the other direction.

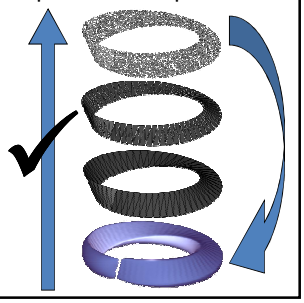


Equivalence of Representations

There are many ways to represent a shape:

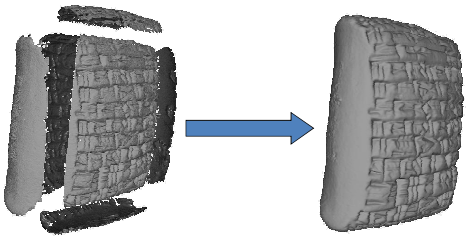
- Point Set
- Polygon Soup
- Polygonal Mesh
- Solid Model

The goal of this work is to define a method for computing solid models from oriented point sets.



Applications

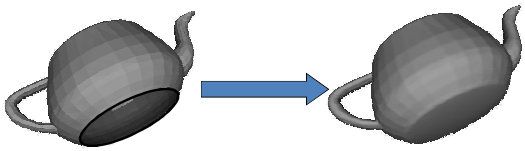
- Surface Blending



Disjoint Model "Zippered" Model

Applications

- Surface Blending
- Hole-Filling

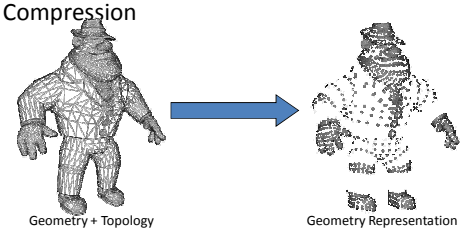


Model with Hole

Water-Tight Model

Applications

- Surface Blending
- Hole-Filling
- Compression

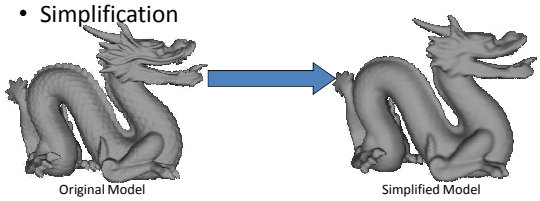


Geometry + Topology Representation

Geometry Representation

Applications

- Surface Blending
- Hole-Filling
- Compression
- Simplification



Original Model
871,000 Triangles

Simplified Model
95,000 Triangles

Related Work

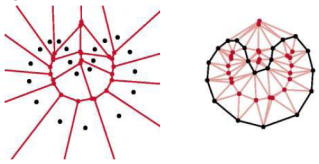
Three general approaches:

1. Computational Geometry
2. Surface Fitting
3. Implicit Function Fitting

Related Work

Three general approaches:

1. Computational Geometry
   - Use the Voronoi diagram / Delaunnay triangulation to extract the surface.



Amenta, Bern and Eppstein. *Graphical Models and Image Processing.* (1998).

Related Work

Three general approaches:

1. Computational Geometry
   - Use the Voronoi diagram / Delaunnay triangulation to extract the surface.

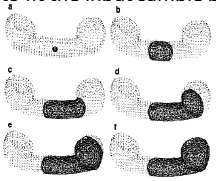
Properties:

- ✓ Complexity of the output is on the order of the complexity of the input.
- × Computing the Voronoi diagram / Delaunnay triangulation can be time consuming.
- × Does not perform well in the presence of noise and non-uniform sampling.

Related Work

Three general approaches:

2. Surface Fitting
   - Deform a base model (represented by a spring system) to fit the input sample points.



Chen and Medioni. *Computer Vision and Image Understanding*. (1995).

Related Work

Three general approaches:

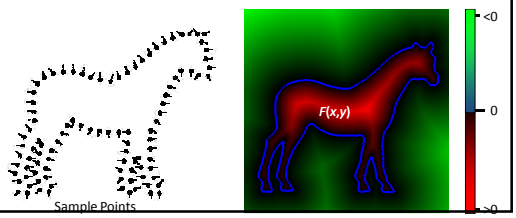
2. Surface Fitting
   - Deform a base model (represented by a spring system) to fit the input sample points.

Properties:

- ✓ Complexity of the output is on the order of the complexity of the input.
- × The reconstructed surface has to have the same topology as the base model.

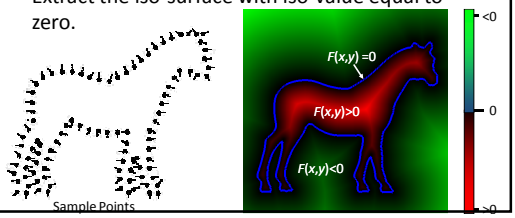
Related Work

3. Implicit Function Fitting
   - Use the point samples to define a function whose values at the sample positions are zero.



Related Work

3. Implicit Function Fitting
   - Use the point samples to define a function whose values at the sample positions are zero.
   - Extract the iso-surface with iso-value equal to zero.



Related Work

3. Implicit Function Fitting
   - Use the point samples to define a function whose values at the sample positions are zero.
   - Extract the iso-surface with iso-value equal to zero.

Properties:

- ✓ No topological restrictions.
- ✓ Noise can be smoothed out.
- × The complexity can be proportional to the size of the volume, not the surface.

Related Work

3. Implicit Function Fitting
   - Use the point samples to define a function whose values at the sample positions are zero.
   - Extract the iso-surface with iso-value equal to zero.

Properties:

- ✓ No topological restrictions.
- ✓ Noise can be smoothed out.
- × The complexity can be proportional to the size of the

How should we define the implicit function so that the reconstruction fits the samples?

Outline

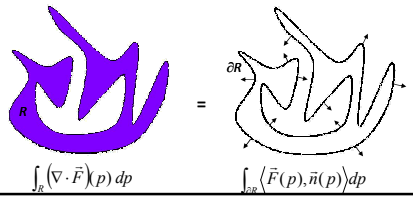
- Introduction
- Related Work
- Approach
  - The Divergence Theorem
  - Reduction to Volume Integration
  - Implementation
- Results
- Conclusion, Extensions, and Future Work

Divergence (Gauss's) Theorem

Given a vector field $\vec{F}$ and a region R:

The volume integral of $\nabla \cdot \vec{F}$ over R and the surface integral of $\vec{F}$ over ∂R are equal:

$$\int_R \left(\nabla \cdot \vec{F}\right)(p)\, dp = \int_{\partial R} \left\langle \vec{F}(p), \vec{n}(p) \right\rangle dp$$



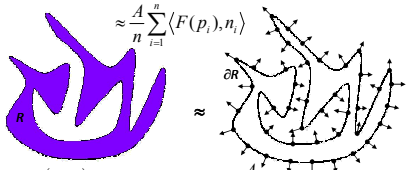
$\int_R \left(\nabla \cdot \vec{F}\right)(p)\, dp$ $\int_{\partial R} \left\langle \vec{F}(p), \vec{n}(p) \right\rangle dp$

Divergence (Gauss's) Theorem

Given a vector field $\vec{F}$ and a region R:

The volume integral of $\nabla \cdot \vec{F}$ over R and the surface integral of $\vec{F}$ over ∂R are equal:

$$\int_R \left(\nabla \cdot \vec{F}\right)(p)\, dp = \int_{\partial R} \left\langle \vec{F}(p), \vec{n}(p) \right\rangle dp \approx \frac{A}{n} \sum_{i=1}^{n} \left\langle F(p_i), n_i \right\rangle$$



$\int_R \left(\nabla \cdot \vec{F}\right)(p)\, dp$ $\frac{A}{n} \sum \left\langle F(p_i), n_i \right\rangle$

Approach

Reduce surface reconstruction to a volume integration problem:

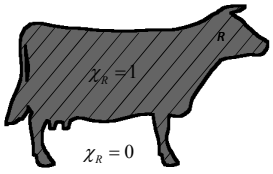
1. Characteristic Function
2. Fourier Coefficients
3. Volume Integrals

Reduction to Volume Integration (Step 1)

Characteristic Function:

The *characteristic function* χ_R of a solid R is the function:

$$\chi_R(x,y,z) = \begin{cases} 1 & \text{if } (x,y,z) \in R \\ 0 & \text{otherwise} \end{cases}$$



Reduction to Volume Integration (Step 2)

Fourier Coefficients:

The *Fourier coefficients* of the characteristic function give an expression of χ_R as a sum of complex exponentials:

$$\chi_R(x,y,z) = \frac{1}{\sqrt{2\pi}} \sum_{l,m,n} \hat{\chi}_R(l,m,n) e^{2\pi i (lx+my+nz)}$$

Reduction to Volume Integration (Step 3)

Volume Integration:
The Fourier coefficients of the characteristic function χ_R can be obtained by integrating:

$$\hat{\chi}_R(l,m,n) = \int_{[0,1]^3} \chi_R(x,y,z) \frac{1}{\sqrt{2\pi}} e^{-2\pi i(lx+my+nz)} dxdydz$$

Reduction to Volume Integration (Step 3)

Volume Integration:
The Fourier coefficients of the characteristic function χ_R can be obtained by integrating:

$$\hat{\chi}_R(l,m,n) = \int_{[0,1]^3} \chi_R(x,y,z) \frac{1}{\sqrt{2\pi}} e^{-2\pi i(lx+my+nz)} dxdydz$$
$$= \int_R \frac{1}{\sqrt{2\pi}} e^{-2\pi i(lx+my+nz)} dxdydz$$

since the characteristic function is one inside of R and zero everywhere else.

Applying the Divergence Theorem

Surface Integration
If $\vec{F}_{lmn}(x,y,z)$ is any function whose divergence is equal to the (l,m,n)-th complex exponential:

$$\left(\nabla \cdot \vec{F}_{lmn}\right)(x,y,z) = \frac{1}{\sqrt{2\pi}} e^{-2\pi i(lx+my+nz)}$$

applying the Divergence Theorem, the volume integral can be expressed as a surface integral:

$$\int_R \frac{1}{\sqrt{2\pi}} e^{-2\pi i(lx+my+nz)} dxdydz = \int_{\partial R} \left\langle \vec{F}_{lmn}(p), n(p) \right\rangle dp$$

Reconstruction Algorithm

Given an oriented point sample $\{(p_i,n_i)\}$:

- Compute a Monte-Carlo approximation of the Fourier coefficients of the characteristic function:

$$\hat{\chi}_R(l,m,n) \approx \sum_{i=1}^{k} \left\langle \vec{F}_{lmn}(p_i), n_i \right\rangle$$

- Apply the inverse Fourier Transform to obtain the characteristic function.
- Extract the reconstruction an iso-surface of the characteristic function

Reconstruction Algorithm

Given an oriented point sample $\{(p_i,n_i)\}$:

- Compute a Monte-Carlo approximation of the Fourier coefficients of the characteristic function:

1. To do this, we need to find a vector valued function $\vec{F}_{l,m,n}(x,y,z)$ such that:

$$\left(\nabla \cdot \vec{F}_{l,m,n}\right)(x,y,z) = \frac{1}{\sqrt{2\pi}} e^{-2\pi i(lx+my+nz)}$$

- Extract the reconstruction an iso-surface of the characteristic function

Reconstruction Algorithm

Given an oriented point sample $\{(p_i,n_i)\}$:

- Compute a Monte-Carlo approximation of the Fourier coefficients of the characteristic function:

1. To do this, we need to find a vector valued function $\vec{F}_{l,m,n}(x,y,z)$ such that:

$$\left(\nabla \cdot \vec{F}_{l,m,n}\right)(x,y,z) = \frac{1}{\sqrt{2\pi}} e^{-2\pi i(lx+my+nz)}$$

2. Directly computing the Fourier coefficients requires summing over all point samples for each of the $O(R^3)$ coefficients.

Choosing the Functions $\vec{F}_{l,m,n}$

There are many solutions to the equation:

$$\left(\nabla \cdot \vec{F}_{l,m,n}\right)(x,y,z) = \frac{1}{\sqrt{2\pi}} e^{-2\pi i(lx+my+nz)}$$

Choosing the Functions $\vec{F}_{l,m,n}$

There are many solutions to the equation:

$$\left(\nabla \cdot \vec{F}_{l,m,n}\right)(x,y,z) = \frac{1}{\sqrt{2\pi}} e^{-2\pi i(lx+my+nz)}$$

Examples:

$$\vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}} \begin{pmatrix} \frac{1}{l} \\ 0 \\ 0 \end{pmatrix} e^{-2\pi i(lx+my+nz)} \quad \vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}} \begin{pmatrix} 0 \\ \frac{1}{m} \\ 0 \end{pmatrix} e^{-2\pi i(lx+my+nz)} \quad \vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}} \begin{pmatrix} 0 \\ 0 \\ \frac{1}{n} \end{pmatrix} e^{-2\pi i(lx+my+nz)}$$

Choosing the Functions $\vec{F}_{l,m,n}$

There are many solutions to the equation:

$$\left(\nabla \cdot \vec{F}_{l,m,n}\right)(x,y,z) = \frac{1}{\sqrt{2\pi}} e^{-2\pi i(lx+my+nz)}$$

Examples:

$$\vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}} \begin{pmatrix} \frac{1}{l} \\ 0 \\ 0 \end{pmatrix} e^{-2\pi i(lx+my+nz)} \quad \vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}} \begin{pmatrix} 0 \\ \frac{1}{m} \\ 0 \end{pmatrix} e^{-2\pi i(lx+my+nz)} \quad \vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}} \begin{pmatrix} 0 \\ 0 \\ \frac{1}{n} \end{pmatrix} e^{-2\pi i(lx+my+nz)}$$

$$\vec{F}_{l,m,n}(x,y,z) = \frac{i}{3(2\pi)^{3/2}} \begin{pmatrix} \frac{1}{l} \\ \frac{1}{m} \\ \frac{1}{n} \end{pmatrix} e^{-2\pi i(lx+my+nz)} \qquad \vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}(l+m+n)} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} e^{-2\pi i(lx+my+nz)}$$

Choosing the Functions $\vec{F}_{l,m,n}$

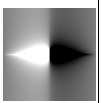
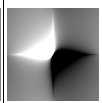
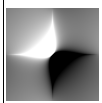
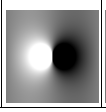
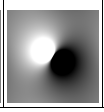
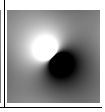
There are many solutions to the equation:

$$\left(\nabla \cdot \vec{F}_{l,m,n}\right)(x,y,z) = \frac{1}{\sqrt{2\pi}} e^{-2\pi i(lx+my+nz)}$$

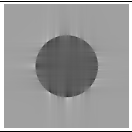
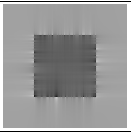
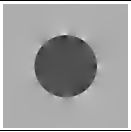
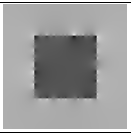
In our implementation, we choose the functions $\vec{F}_{l,m,n}$ to be the unique vector fields that commute with translation and rotation:

$$\vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}(l^2+m^2+n^2)} \begin{pmatrix} l \\ m \\ n \end{pmatrix} e^{-2\pi i(lx+my+nz)}$$

Choosing the Functions $\vec{F}_{l,m,n}$

	← 0°	↘ 30°	↘ 45°
Does not Commute: $\vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}(l+m+n)} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} e^{-2\pi i(lx+my+nz)}$			
Commutes: $\vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}(l^2+m^2+n^2)} \begin{pmatrix} l \\ m \\ n \end{pmatrix} e^{-2\pi i(lx+my+nz)}$			

Choosing the Functions $\vec{F}_{l,m,n}$

		
Does not Commute: $\vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}(l+m+n)} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} e^{-2\pi i(lx+my+nz)}$		
Commutes: $\vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}(l^2+m^2+n^2)} \begin{pmatrix} l \\ m \\ n \end{pmatrix} e^{-2\pi i(lx+my+nz)}$		

Computing χ_R (Efficiently)

Given an oriented point set $\{(p_i,n_i)\}$, the (l,m,n)-th Fourier coefficient of the characteristic function is defined to be:

$$\hat{\chi}_R(l,m,n) = \sum_{j=1}^{k} \left\langle \vec{F}_{lmn}(p_j), n_j \right\rangle$$

Using this to compute each Fourier coefficient would result in an algorithm that would be much too slow.

Computing χ_R (Efficiently)

$$\hat{\chi}_R(l,m,n) = \sum_{j=1}^{k} \left\langle \vec{F}_{lmn}(p_j), n_j \right\rangle$$

If we denote by $\vec{V}(p)$ the vector field:

$$\vec{V}(p) = \sum_{j=1}^{k} \delta_{p_j}(p) \cdot n_j$$

Computing χ_R (Efficiently)

$$\hat{\chi}_R(l,m,n) = \sum_{j=1}^{k} \left\langle \vec{F}_{lmn}(p_j), n_j \right\rangle$$

If we denote by $\vec{V}(p)$ the vector field:

$$\vec{V}(p) = \sum_{j=1}^{k} \delta_{p_j}(p) \cdot n_j$$

Then we can re-write the equation for the (l,m,n)-th Fourier coefficient as:

$$\hat{\chi}_R(l,m,n) = \int \left\langle \vec{F}_{lmn}(p), \vec{V}(p) \right\rangle dp$$

Computing χ_R (Efficiently)

$$\hat{\chi}_R(l,m,n) = \int \left\langle \vec{F}_{lmn}(p), \vec{V}(p) \right\rangle dp$$

Using the fact that the function $\vec{F}_{lmn}$ is a vector-multiple of (l,m,n)-th complex exponential:

$$\vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}\left(l^2+m^2+n^2\right)} \begin{pmatrix} l \\ m \\ n \end{pmatrix} e^{-2\pi i(lx+my+nz)}$$

Computing χ_R (Efficiently)

$$\hat{\chi}_R(l,m,n) = \int \left\langle \vec{F}_{lmn}(p), \vec{V}(p) \right\rangle dp$$

Using the fact that the function $\vec{F}_{lmn}$ is a vector-multiple of (l,m,n)-th complex exponential:

$$\vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}\left(l^2+m^2+n^2\right)} \begin{pmatrix} l \\ m \\ n \end{pmatrix} e^{-2\pi i(lx+my+nz)}$$

We get an expression for the (l,m,n)-th Fourier coefficient of the characteristic function:

$$\hat{\chi}_R(l,m,n) = \frac{i}{(2\pi)^{3/2}\left(l^2+m^2+n^2\right)} \left\langle (l,m,n), \int \vec{V}(x,y,z) e^{-2\pi i(lx+my+nz)} dxdydz \right\rangle$$

Computing χ_R (Efficiently)

$$\hat{\chi}_R(l,m,n) = \int \left\langle \vec{F}_{lmn}(p), \vec{V}(p) \right\rangle dp$$

Using the fact that the function $\vec{F}_{lmn}$ is a vector-multiple of (l,m,n)-th complex exponential:

$$\vec{F}_{l,m,n}(x,y,z) = \frac{i}{(2\pi)^{3/2}\left(l^2+m^2+n^2\right)} \begin{pmatrix} l \\ m \\ n \end{pmatrix} e^{-2\pi i(lx+my+nz)}$$

We get an expression for the (l,m,n)-th Fourier coefficient of the characteristic function:

$$\hat{\chi}_R(l,m,n) = \frac{i}{(2\pi)^{3/2}\left(l^2+m^2+n^2\right)} \left\langle (l,m,n), \int \vec{V}(x,y,z) e^{-2\pi i(lx+my+nz)} dxdydz \right\rangle$$

$$= \frac{i}{(2\pi)^{3/2}\left(l^2+m^2+n^2\right)} \left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

Computing χ_R (Efficiently)

Thus, we obtain the characteristic function by:

1. Computing the Fourier coefficients of $\vec{V}$:

$$\vec{V} \to \hat{\vec{V}}$$

Computing χ_R (Efficiently)

Thus, we obtain the characteristic function by:

1. Computing the Fourier coefficients of $\vec{V}$:

$$\vec{V} \to \hat{\vec{V}}$$

2. Computing the Fourier coefficients of the characteristic function:

$$\hat{\chi}_R(l,m,n) = \frac{i}{(2\pi)^{3/2}\left(l^2+m^2+n^2\right)}\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$$

Computing χ_R (Efficiently)

Thus, we obtain the characteristic function by:

1. Computing the Fourier coefficients of $\vec{V}$:

$$\vec{V} \to \hat{\vec{V}}$$

2. Computing the Fourier coefficients of the characteristic function:

$$\hat{\chi}_R(l,m,n) = \frac{i}{(2\pi)^{3/2}\left(l^2+m^2+n^2\right)}\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$$

3. Computing the inverse Fourier transform.

Computing χ_R (Efficiently)

Thus, we obtain the characteristic function by:

1. Computing the Fourier coefficients of $\vec{V}$:

$$\vec{V} \to \hat{\vec{V}}$$

2. Computing the Fourier coefficients of the

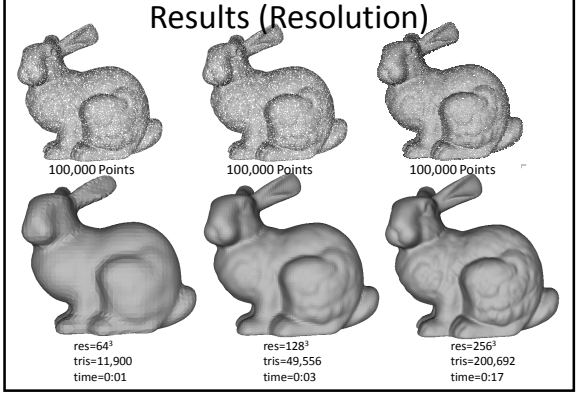
The whole reconstruction process can be performed in $O(N^3 \log n)$

3. Computing the inverse Fourier transform.

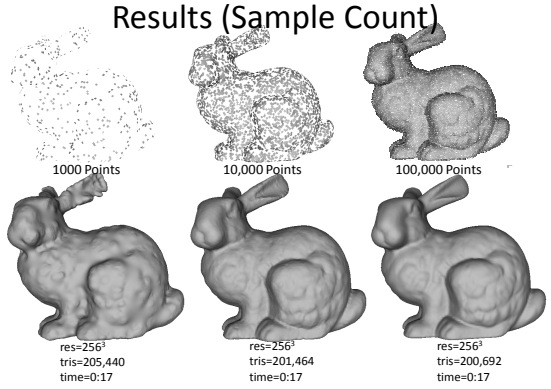
Outline

- Introduction
- Related Work
- Approach
- Results
- Conclusion, Extensions, and Future Work

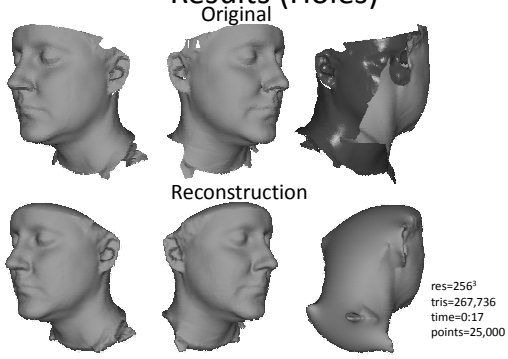
Results (Resolution)



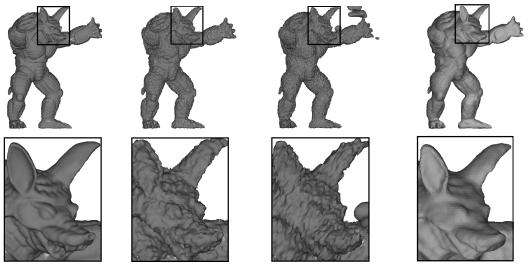
Results (Sample Count)



Results (Holes)



Results (Noise / Related Work)



Outline

- Introduction
- Related Work
- Approach
- Results
- Discussion

Summary

<u>Theoretical Contribution</u>

- Transformed the surface reconstruction problem into a Volume integral
- Used the Divergence Theorem to express the integral as a surface integral
- Used Monte-Carlo integration to approximate the surface integral as a summation over an oriented point sample

Summary

It's easy to implement, but …

<u>Limitations</u>

× For an $O(R^2)$ reconstruction:

- Temporal Complexity: $O(R^3 \log(R))$
- Spatial Complexity: $O(R^3)$

Revisiting the Algorithm

Recall:

We compute the Fourier coefficients of the characteristic function by setting:

$$\hat{\chi}_R(l,m,n) = \frac{i}{(2\pi)^{3/2}\left(l^2+m^2+n^2\right)}\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$$

where $\vec{V}$ is the "splatted" normal field.

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}}\frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$$

What is the operator $i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$ doing?

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}}\frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$$

What is the operator $i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$ doing?

Fourier:

Given a 1D function $f(\theta)$, we can express f in terms of its Fourier decomposition:

$$f(\theta) = \sum_k a_k e^{ik\theta}$$

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}}\frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$$

What is the operator $i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$ doing?

Fourier:

Given a 1D function $f(\theta)$, we can express f in terms of its Fourier decomposition:

$$f(\theta) = \sum_k a_k e^{ik\theta}$$

Then the derivative of f can be expressed as:

$$f'(\theta) = i\sum_k a_k k e^{ik\theta}$$

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}}\frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$$

What is the operator $i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$ doing?

Fourier:

Given a 1D function $f(\theta)$, we can express f in terms of its Fourier decomposition:

$$\widehat{f'}(k) = ik\hat{f}(k)$$

Then the derivative of f can be expressed as:

$$f'(\theta) = i\sum_k a_k k e^{ik\theta}$$

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}}\frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$$

What is the operator $i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n)\right\rangle$ doing?

Fourier:

Given a 2D function $f(\theta,\phi)$, we can express f in terms of its Fourier decomposition:

$$f(\theta,\phi) = \sum_{k,l} a_{k,l} e^{i(k\theta+l\phi)}$$

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}} \frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

What is the operator $i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$ doing?

Fourier:

Given a 2D function $f(\theta,\phi)$, we can express f in terms of its Fourier decomposition:

$$f(\theta,\phi) = \sum_{k,l} a_{k,l} e^{i(k\theta+l\phi)}$$

Then the partials of f can be expressed as:

$$\frac{\partial f}{\partial \theta}(\theta,\phi) = i\sum_{k,l} a_{k,l} k e^{i(k\theta+l\phi)} \qquad \frac{\partial f}{\partial \phi}(\theta,\phi) = i\sum_{k,l} a_{k,l} l e^{i(k\theta+l\phi)}$$

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}} \frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

What is the operator $i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$ doing?

Fourier:

Given a 2D function $f(\theta,\phi)$, we can express f in

$$\widehat{\frac{\partial \vec{F}}{\partial \theta}}(k,l) = ik\vec{F}(k,l) \qquad \widehat{\frac{\partial \vec{F}}{\partial \phi}}(k,l) = il\vec{F}(k,l)$$

Then the partials of f can be expressed as:

$$\frac{\partial f}{\partial \theta}(\theta,\phi) = i\sum_{k,l} a_{k,l} k e^{i(k\theta+l\phi)} \qquad \frac{\partial f}{\partial \phi}(\theta,\phi) = i\sum_{k,l} a_{k,l} l e^{i(k\theta+l\phi)}$$

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}} \frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

What is the operator $i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$ doing?

Fourier:

Given a 2D vector field $\vec{F}(\theta,\phi)$, we can express $\vec{F}$ in terms of its Fourier decomposition:

$$\vec{F}(\theta,\phi) = \left(\sum_{k,l} a_{k,l} e^{i(k\theta+l\phi)}, \sum_{k,l} b_{k,l} e^{i(k\theta+l\phi)}\right)$$

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}} \frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

What is the operator $i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$ doing?

Fourier:

Given a 2D vector field $\vec{F}(\theta,\phi)$, we can express $\vec{F}$ in terms of its Fourier decomposition:

$$\vec{F}(\theta,\phi) = \left(\sum_{k,l} a_{k,l} e^{i(k\theta+l\phi)}, \sum_{k,l} b_{k,l} e^{i(k\theta+l\phi)}\right)$$

Then the divergence of f can be expressed as:

$$\mathrm{div}(\vec{F})(\theta,\phi) = i\sum_{k,l} a_{k,l} k e^{i(k\theta+l\phi)} + i\sum_{k,l} b_{k,l} l e^{i(k\theta+l\phi)}$$

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}} \frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

What is the operator $i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$ doing?

Fourier:

Given a 2D vector field $\vec{F}(\theta,\phi)$, we can express $\vec{F}$ in t

$$\widehat{\mathrm{div}(\vec{F})}(k,l) = i\left\langle (k,l), \hat{\vec{F}}(k,l) \right\rangle$$

The

$$\mathrm{div}(\vec{F})(\theta,\phi) = i\sum_{k,l} a_{k,l} k e^{i(k\theta+l\phi)} + i\sum_{k,l} b_{k,l} l e^{i(k\theta+l\phi)}$$

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}} \frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

What is the operator $\frac{1}{\left(l^2+m^2+n^2\right)}$ doing?

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}} \frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

What is the operator $\frac{1}{\left(l^2+m^2+n^2\right)}$ doing?

Fourier:

Given a 2D function $f(\theta,\phi)$, we can express f in terms of its Fourier decomposition:

$$f(\theta,\phi) = \sum_{k,l} a_{k,l} e^{i(k\theta+l\phi)}$$

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}} \frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

What is the operator $\frac{1}{\left(l^2+m^2+n^2\right)}$ doing?

Fourier:

Given a 2D function $f(\theta,\phi)$, we can express f in terms of its Fourier decomposition:

$$f(\theta,\phi) = \sum_{k,l} a_{k,l} e^{i(k\theta+l\phi)}$$

Then the Laplacian of f can be expressed as:

$$\Delta F(\theta,\phi) = -\sum_{k,l} a_{k,l} k^2 e^{i(k\theta+l\phi)} - \sum_{k,l} a_{k,l} l^2 e^{i(k\theta+l\phi)}$$

Revisiting the Algorithm

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}} \frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

What is the operator $\frac{1}{\left(l^2+m^2+n^2\right)}$ doing?

Fourier:

Given a 2D function $f(\theta,\phi)$, we can express f in terms of its Fourier decomposition:

$$\widehat{\Delta F}(k,l) = -(k^2+l^2)\hat{F}(k,l)$$

Then the Laplacian of f can be expressed as:

$$\Delta F(\theta,\phi) = -\sum_{k,l} a_{k,l} k^2 e^{i(k\theta+l\phi)} - \sum_{k,l} a_{k,l} l^2 e^{i(k\theta+l\phi)}$$

Revisiting the Algorithm

In sum, what we're really doing to reconstruct the characteristic function:

1. Computing the divergence of the normal field $\vec{V}$.

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}} \frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

Revisiting the Algorithm

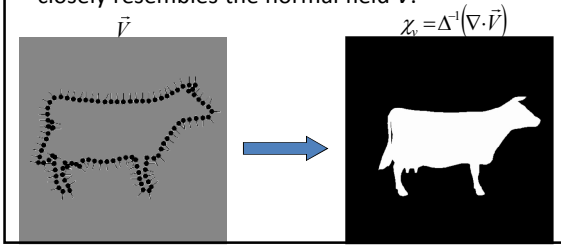
In sum, what we're really doing to reconstruct the characteristic function:

1. Computing the divergence of the normal field $\vec{V}$.
2. And then solving Laplace's equation (inverting the process of taking the Laplacian)

$$\hat{\chi}_R(l,m,n) = \frac{1}{(2\pi)^{3/2}} \frac{1}{\left(l^2+m^2+n^2\right)} i\left\langle (l,m,n), \hat{\vec{V}}(l,m,n) \right\rangle$$

Revisiting the Algorithm

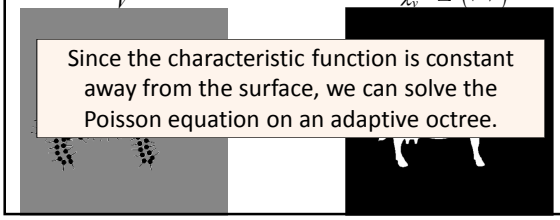
So what we're really doing is solving for the characteristic function χ_R whose gradient most closely resembles the normal field $\vec{V}$.



Revisiting the Algorithm

So what we're really doing is solving for the characteristic function χ_R whose gradient most closely resembles the normal field $\vec{V}$.

$\vec{V}$ $\chi_v = \Delta^{-1}(\nabla \cdot \vec{V})$

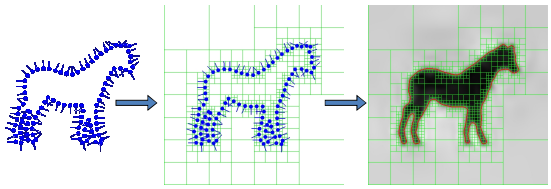


Since the characteristic function is constant away from the surface, we can solve the Poisson equation on an adaptive octree.

Solving on an Octree

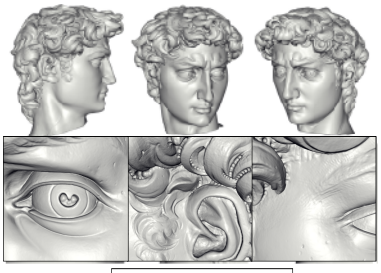
Computational Implications:

The temporal and spatial complexity now becomes proportional to the size of the surface (i.e. quadratic in resolution), not the volume (i.e. cubic in resolution).



Solving on an Octree

In Practice:



Effective Resolution: 2048^3

Octree Recon:
Time: ~3hrs
Memory: 5GB

Projected FFT Recon:
Time: ~8hrs
Memory: 512GB

216×10^6 points (4.8 GB)

Pushing the Envelope

In fact, if we want to reconstruct really really high-res models, we can consider using a streaming out-of-core solver that does not need to maintain the entire octree in local memory.

Pushing the Envelope

216×10^6 points (4.8 GB)

Res.	Octree Memory	Peak Memory	Running Time
256	48 49	521 309	0.53 0.50
512	168 188	278 442	0.68 0.65
1024	702 818	213 1,285	1.20 1.05
2048	3,070 3,695	212 4,442	3.33 2.65
4096	13,367 N/A	427 N/A	12.6 N/A
8192	39,452 N/A	780 N/A	32.3 N/A

Out-of-Core Reconstruction
In-Core Reconstruction

Pushing the Envelope

In-Core Reconstruction
Peak Mem: 4.4 GB

Out-of-Core Reconstruction
Peak Mem: 0.8 GB