FFTs in Graphics and Vision

Surface Reconstruction

Shape Spectrum

There are many ways to represent a shape:

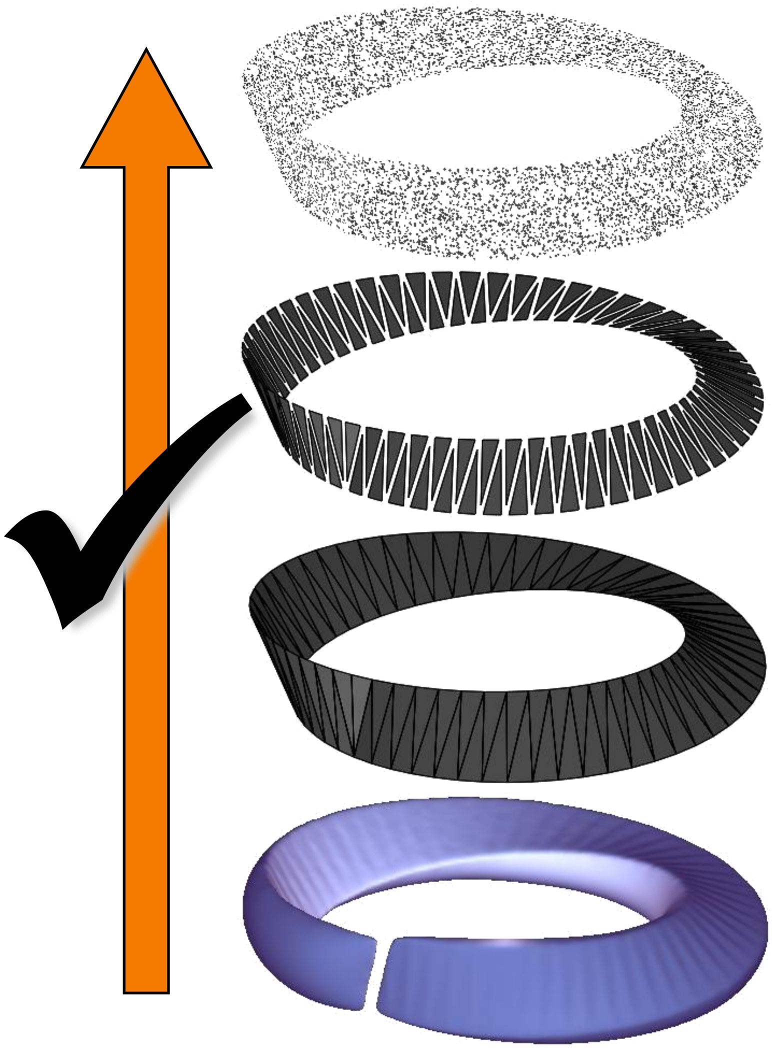
- Point Set
- Polygon Soup
- Polygonal Mesh
- Solid Model

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



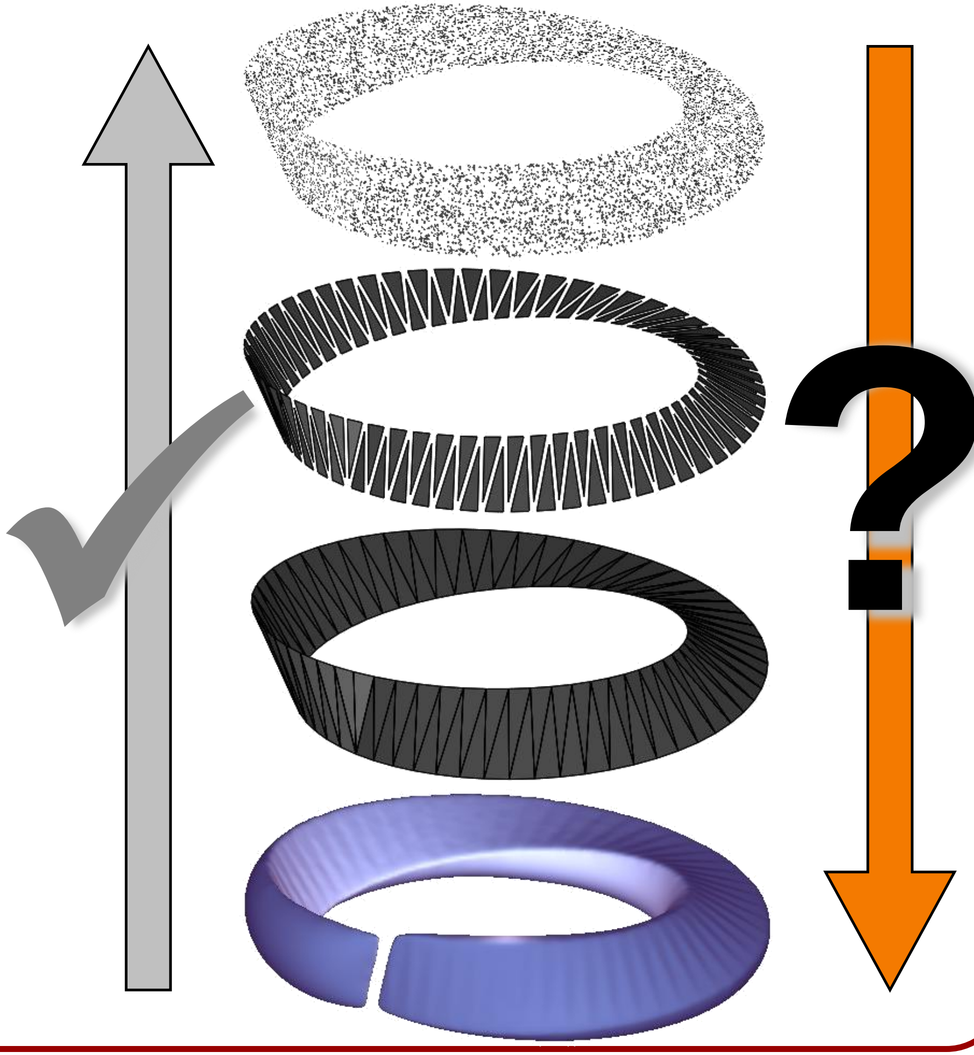
Equivalence of Representations

There are many ways to represent a shape:

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

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

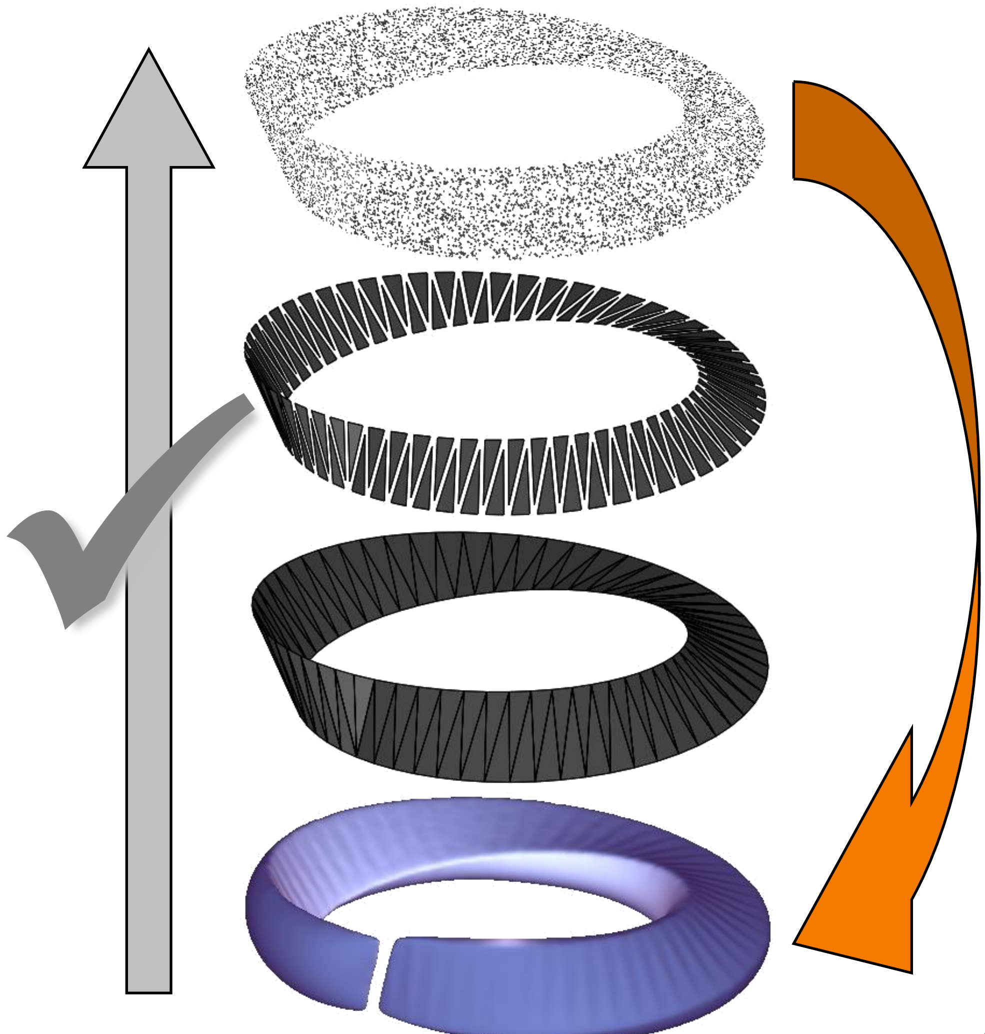
The challenge is to transition in the other direction



Equivalence of Representations

There are many ways to represent a shape:

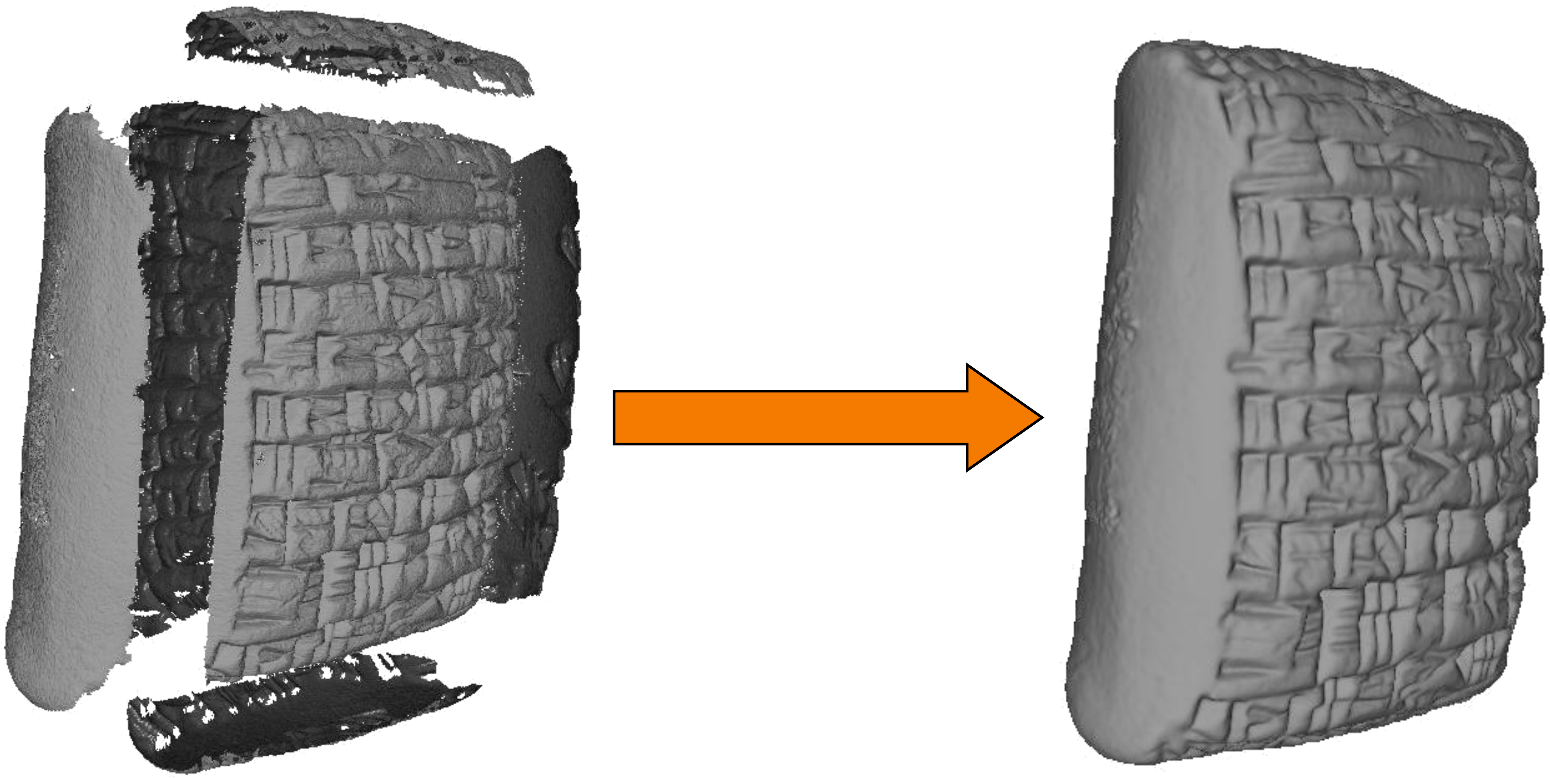
- Point Set
- Polygon Soup
- Polygonal Mesh
- Solid Model



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

Applications

- Surface Blending

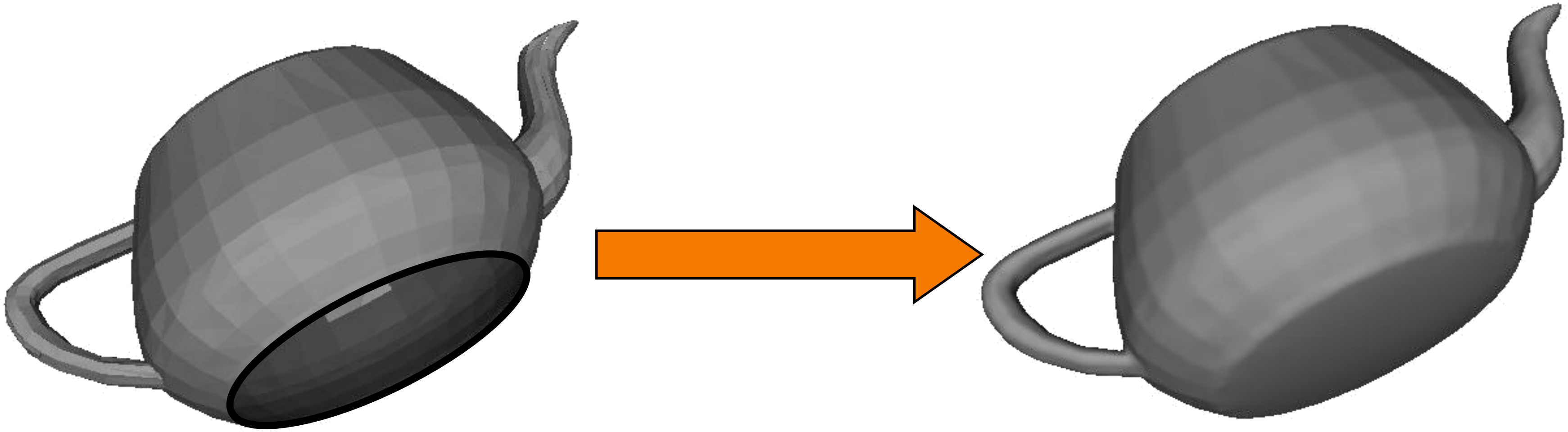


Disjoint Model

"Zippered" Model

Applications

- Surface Blending
- Hole-Filling

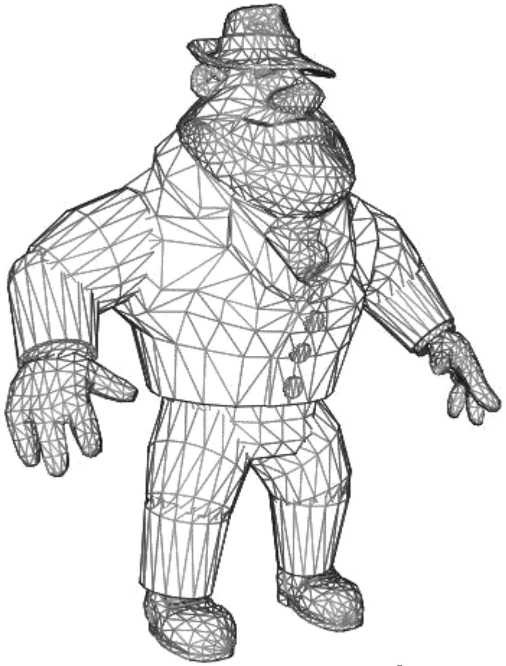


Model with Hole

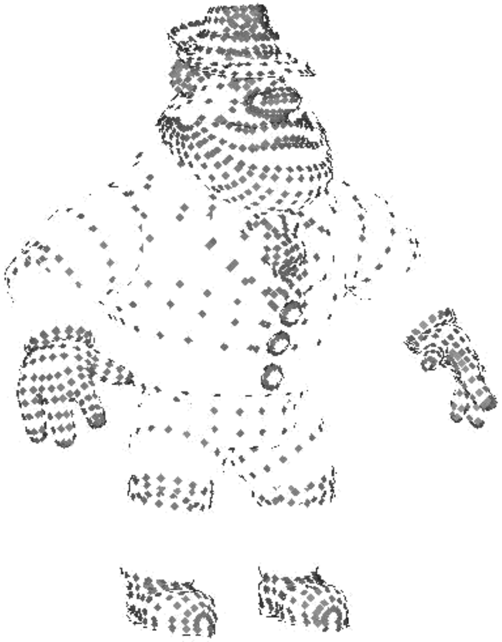
Water-Tight Model

Applications

- Surface Blending
- Hole-Filling
- Compression



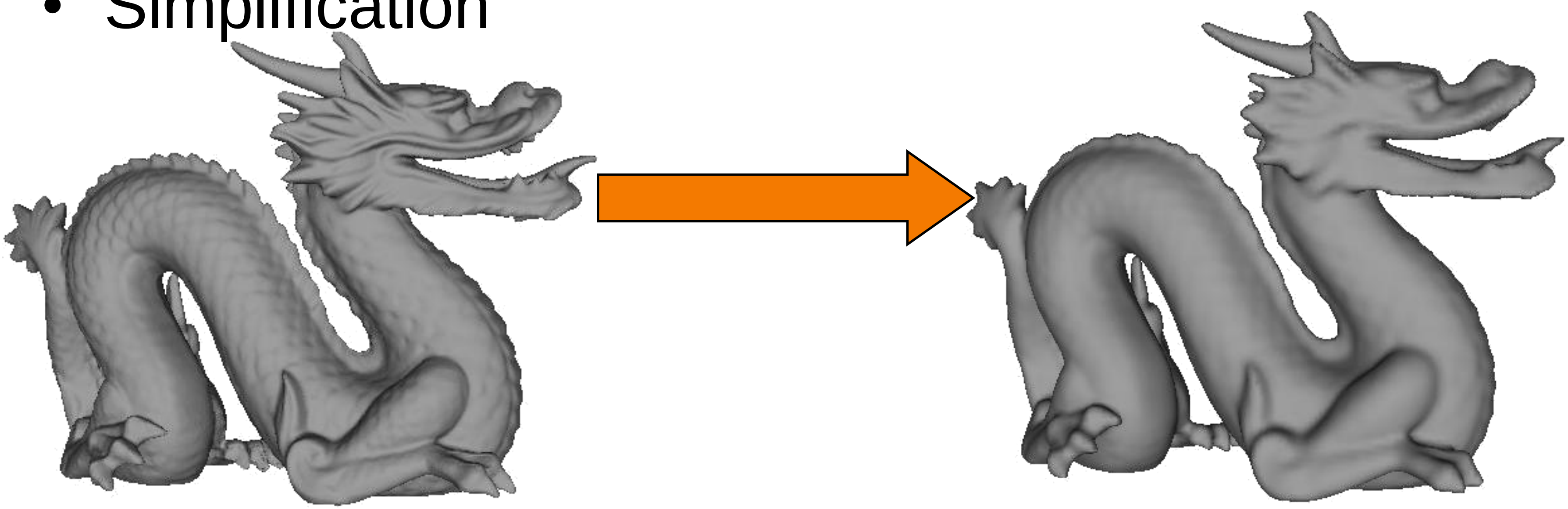
Geometry + Topology
Representation



Geometry
Representation

Applications

- Surface Blending
- Hole-Filling
- Compression
- Simplification



Original Model
871,000 Triangles

Simplified Model
95,000 Triangles

Related Work

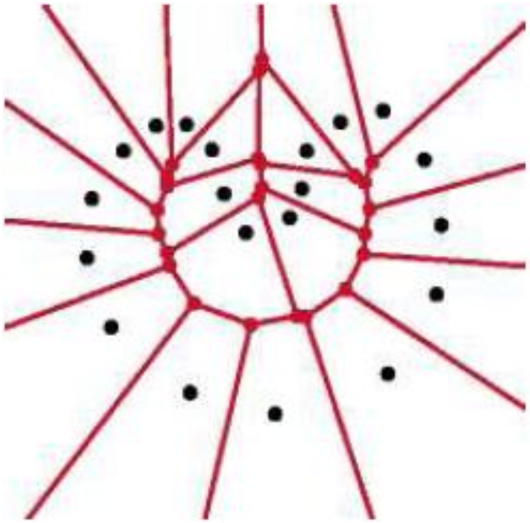
Three general approaches:

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.

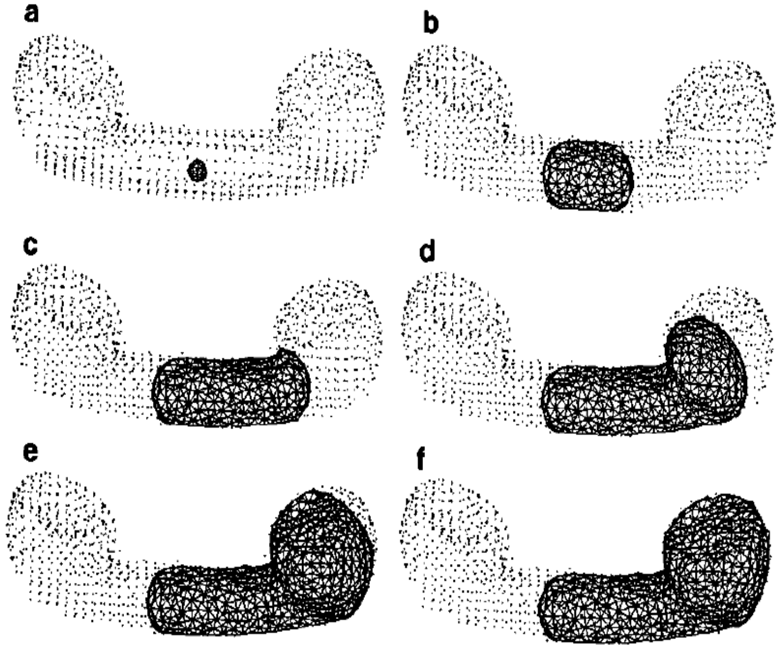
Properties:

- ✓ 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:

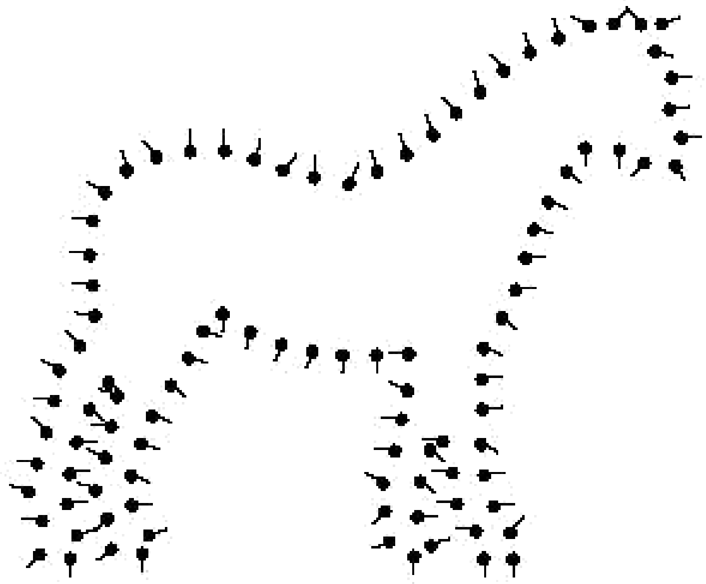
2. Surface Fitting
   - Deform a base model (represented by a spring system) to fit the input sample points.

<u>Properties</u>:

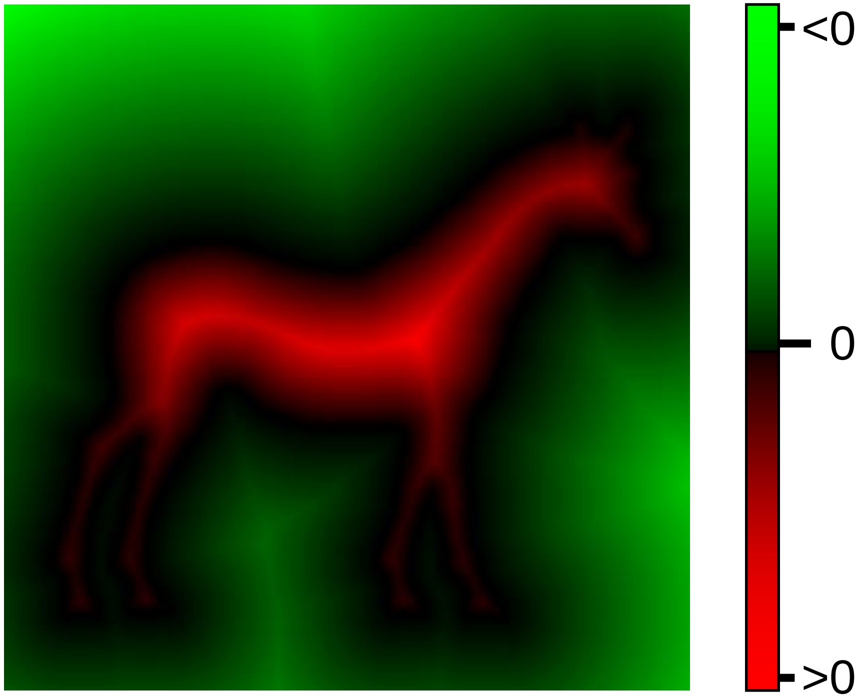
- ✓ Complexity of the output is on the order of the complexity of the input.
- ✗ The reconstructed surface has to have the same topology as the base model.

Related Work

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



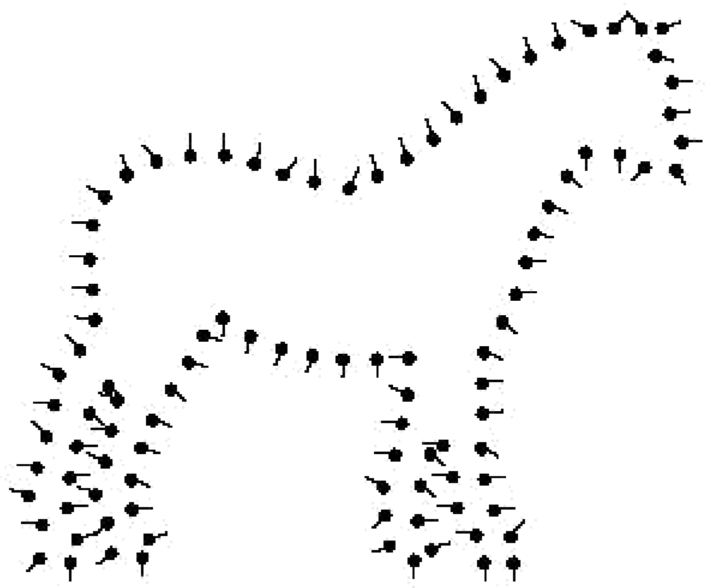
Sample Points



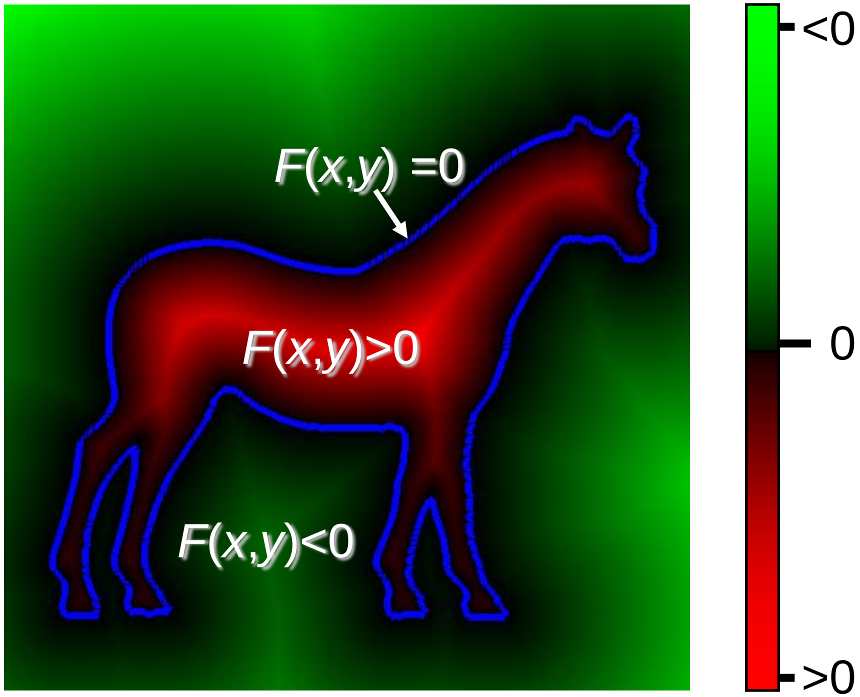
$F(x,y)$

Related Work

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



Sample Points



F(*x*,*y*)=0

Related Work

3. Implicit Function Fitting
   - Use the point samples to define an 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 of the reconstruction depends on the sampling resolution, not the number of input points.

Related Work

3. Implicit Function Fitting
   - Use the point samples to define an 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 of the reconstruction depends on the sampling resolution, not the number of input points.

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

Outline

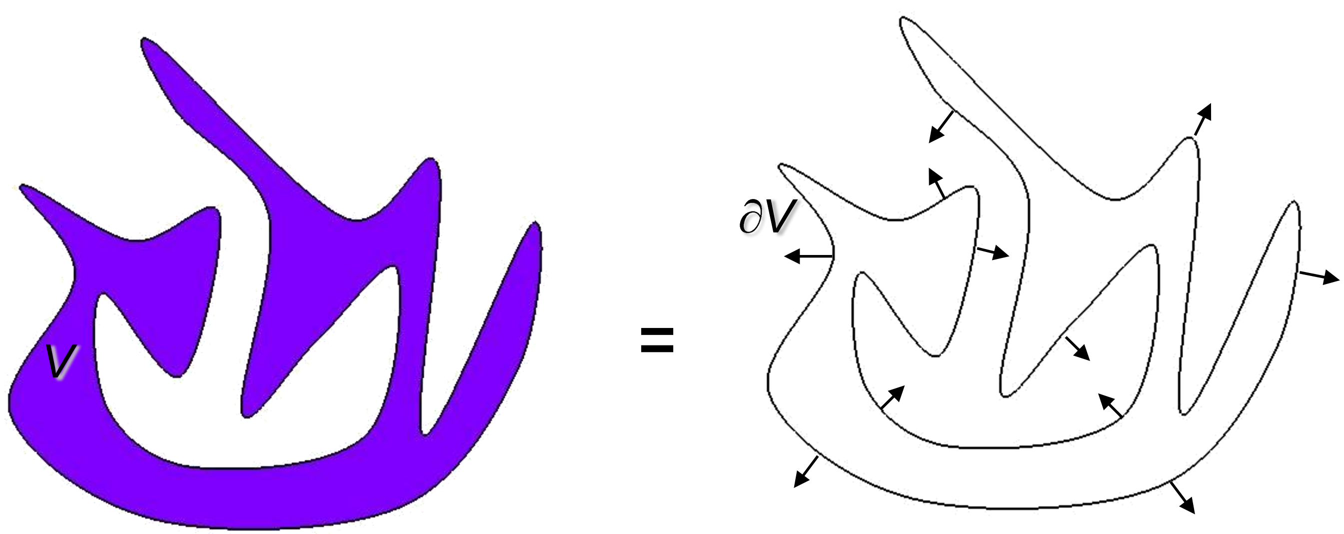
- Introduction
- Related Work
- Approach
    - The Divergence Theorem
    - Reduction to Volume Integration
    - Implementation
- Results
- Conclusion

Divergence (Gauss's) Theorem

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

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

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



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

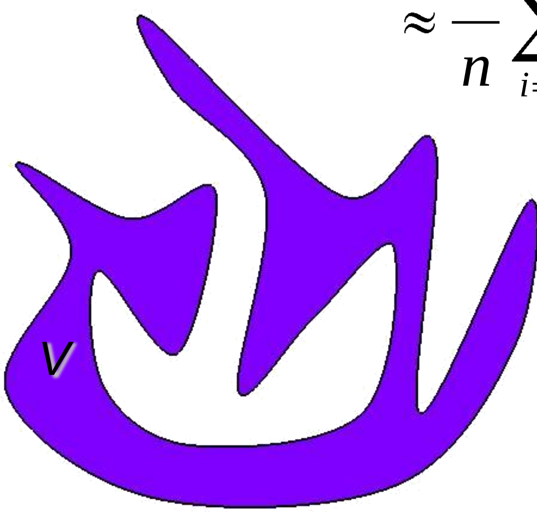
Divergence (Gauss's) Theorem

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

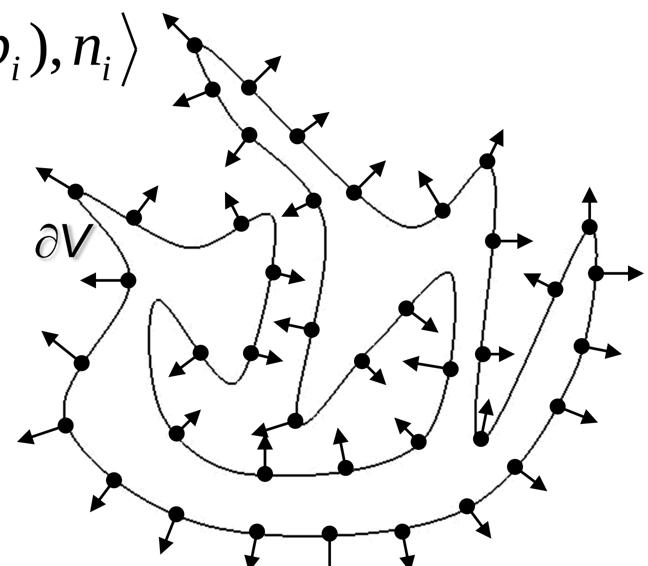
The volume integral of $\nabla\cdot\vec{F}$ over V and the surface integral of $\vec{F}$ over ∂V are equal:

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

$$\approx \frac{A}{n}\sum_{i=1}^{n}\left\langle F(p_i), n_i \right\rangle$$



$\approx$



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

Approach

Reduce surface reconstruction to a volume integration problem:

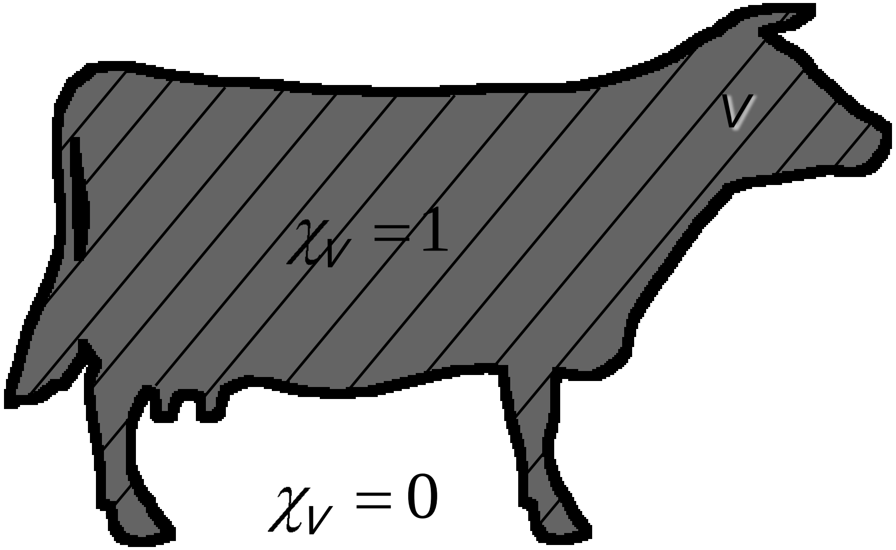
1. Characteristic Function
2. Fourier Coefficients
3. Volume Integrals

Reduction to Volume Integration (Step 1)

Characteristic Function:

The *characteristic function* χ_V of a solid V is the function:

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



Reduction to Volume Integration (Step 2)

Fourier Coefficients:

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

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

Reduction to Volume Integration (Step 3)

<u>Volume Integration</u>:

The Fourier coefficients of the characteristic function χ_V can be obtained by integrating:

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

Reduction to Volume Integration (Step 3)

Volume Integration:

The Fourier coefficients of the characteristic function χ_V can be obtained by integrating:

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

$$= \int_V \frac{1}{\sqrt{2\pi}} e^{-2\pi i (x+my+nz)} dxdydz$$

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

Applying the Divergence Theorem

<u>Surface Integration</u>

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

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

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

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

Reconstruction Algorithm

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

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

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

- 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:

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

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

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

Reconstruction Algorithm

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

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

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

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

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

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

Choosing the Functions $F_{l,m,n}$

There are many solutions to the equation:

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

Choosing the Functions $F_{l,m,n}$

There are many solutions to the equation:

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

<u>Examples</u>:

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

Choosing the Functions $F_{l,m,n}$

There are many solutions to the equation:

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

<u>Examples</u>:

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

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

Choosing the Functions $F_{l,m,n}$

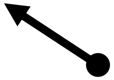
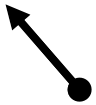
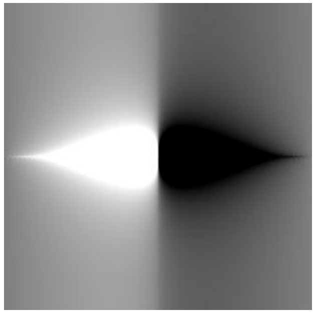
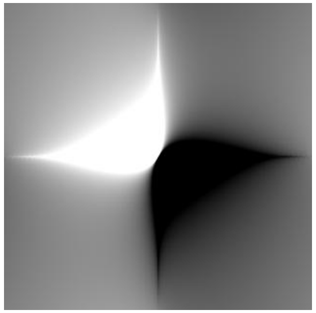
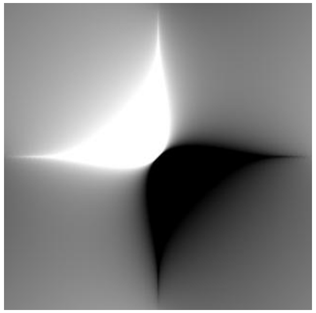
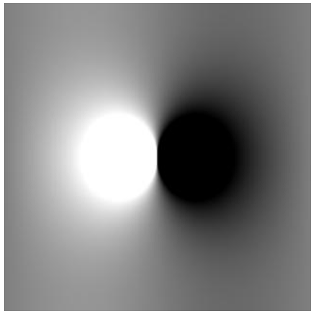
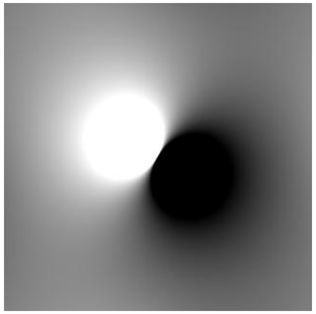
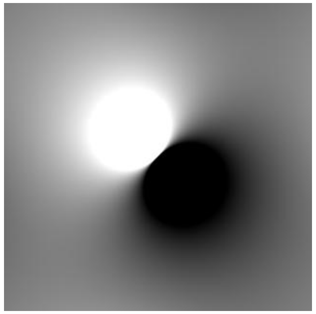
There are many solutions to the equation:

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

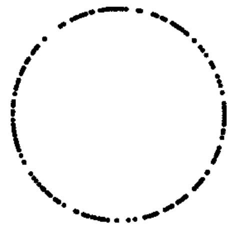
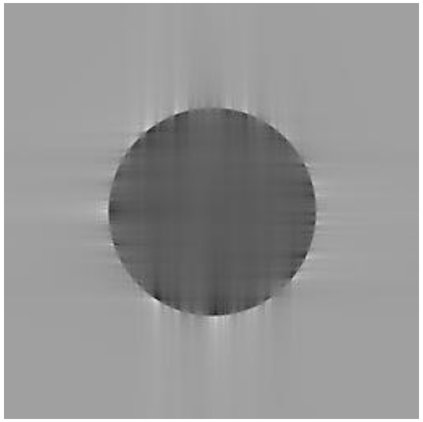
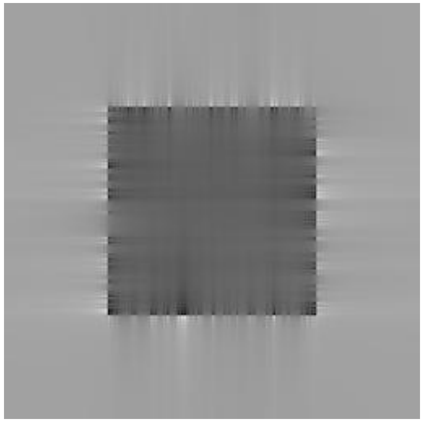
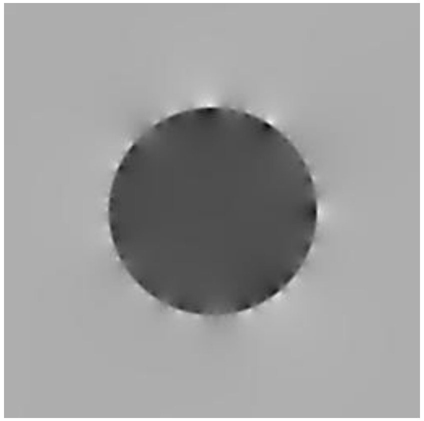
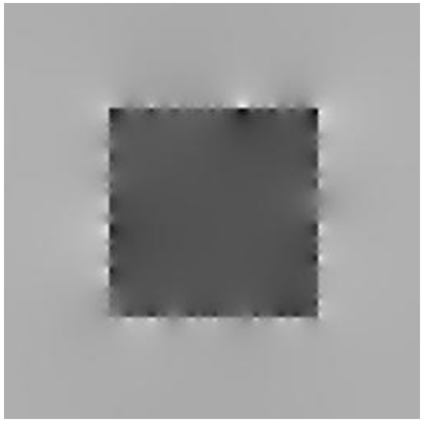
In our implementation, we choose the functions $F_{l,m,n}$ to be the unique vector fields that commute with rotation:

$$F_{l,m,n}(x,y,z) = \frac{1}{(2\pi)^{3/2}} \begin{pmatrix} \dfrac{il}{l^2+m^2+n^2} e^{-2\pi i\left(x+my+nz\right)} \\ \dfrac{im}{l^2+m^2+n^{2^2}} e^{-2\pi i\left(x+my+nz\right)} \\ \dfrac{in}{l^2+m^2+n^2} e^{-2\pi i\left(x+my+nz\right)} \end{pmatrix}$$

Choosing the Functions $F_{l,m,n}$

	 0^o	 30^o	 45^o
Does not Commute: $$F_{l,m,n}(x,y,z)=\frac{1}{(2\pi)^{3/2}}\begin{pmatrix}\frac{i}{l+m+n}e^{-2\pi i(x+my+nz)}\\ \frac{i}{l+m+n}e^{-2\pi i(x+my+nz)}\\ \frac{i}{l+m+n}e^{-2\pi i(x+my+nz)}\end{pmatrix}$$			
Commutes: $$F_{l,m,n}(x,y,z)=\frac{1}{(2\pi)^{3/2}}\begin{pmatrix}\frac{il}{l^2+m^2+n^2}e^{-2\pi i(x+my+nz)}\\ \frac{im}{l^2+m^2+n^2}e^{-2\pi i(x+my+nz)}\\ \frac{in}{l^2+m^2+n^2}e^{-2\pi i(x+my+nz)}\end{pmatrix}$$			

Choosing the Functions $F_{l,m,n}$

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

Choosing the Functions $F_{l,m,n}$

Convolution:

In general, to convolve with a filter *f*:

We take the sum of different scales of f at different translations :

$$(f * g)(t) = \int g(s) f(t-s)\, ds$$

Choosing the Functions $F_{l,m,n}$

Convolution:

In general, to convolve with a filter *f*:

We take the sum of different scales of f at different translations:

$$(f * g)(t) = \int g(s) f(t - s)\, ds$$

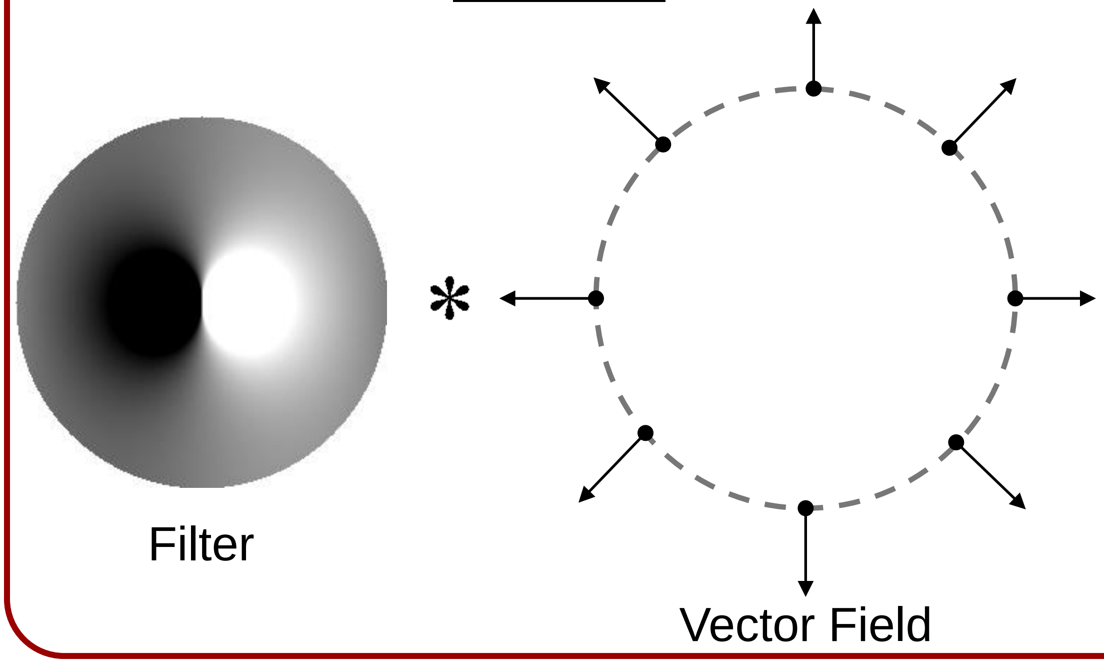
When the functions $F_{l,m,n}$ commute with rotation, we extend the notion of convolution:

We take the sum of different rotations of F, at different scales, and different translations.

Choosing the Functions $F_{l,m,n}$

<u>Extended Convolution</u>:

When the functions $F_{l,m,n}$ commute with rotation, we extend the notion of convolution:
We take the sum of different <u>rotations</u> of F, at different <u>scales</u>, and different <u>translations</u>.

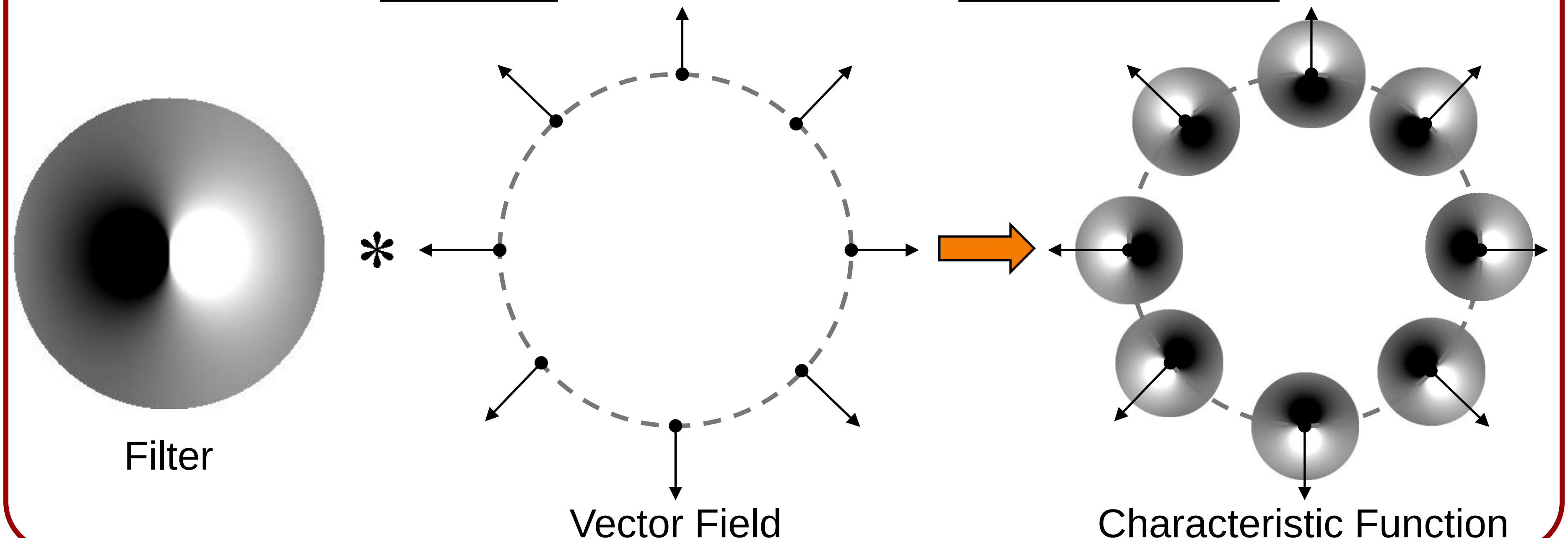


Choosing the Functions $F_{l,m,n}$

<u>Extended Convolution</u>:

When the functions $F_{l,m,n}$ commute with rotation, we extend the notion of convolution:

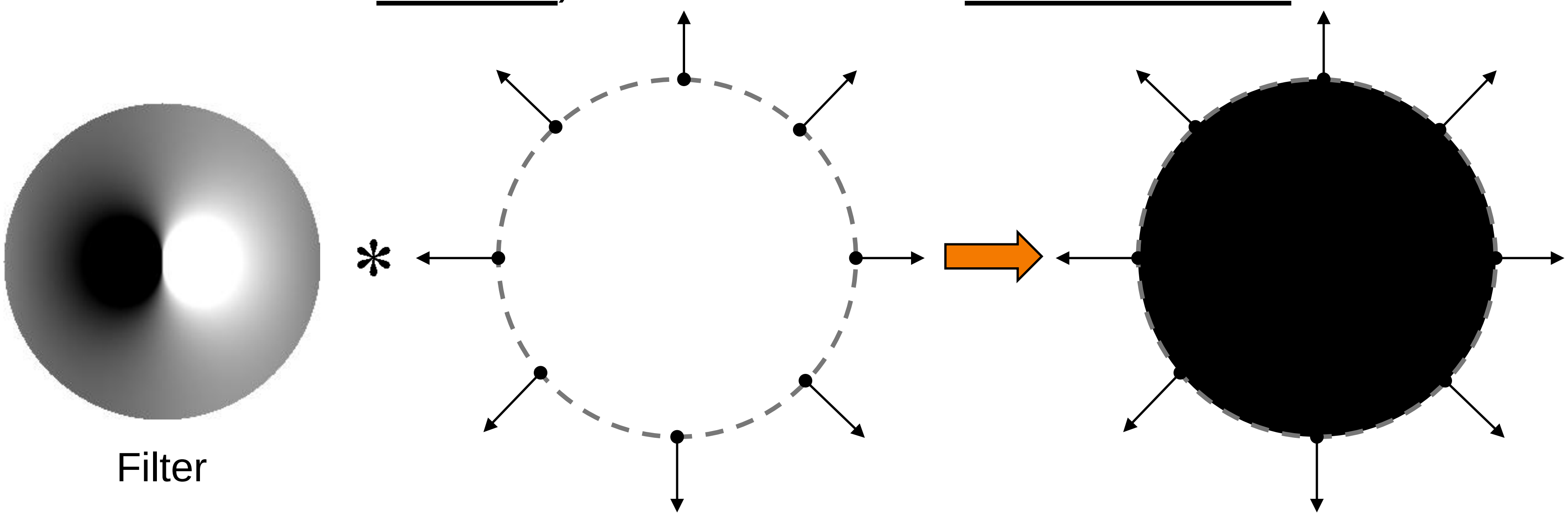
We take the sum of different <u>rotations</u> of F, at different <u>scales</u>, and different <u>translations</u>.



Choosing the Functions $F_{l,m,n}$

Extended Convolution:

When the functions $F_{l,m,n}$ commute with rotation, we extend the notion of convolution:
We take the sum of different rotations of F, at different scales, and different translations.



Choosing the Functions $F_{l,m,n}$

Extended Convolution:

When the functions $F_{l,m,n}$ commute with rotation, we extend the notion of convolution:

By turning the surface integral into a convolution, we can compute the characteristic function efficiently:

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

$$O(n^5)$$

$$\{p_i, n_i\} \rightarrow V(x,y,z) \approx \sum_{i=1}^{k} \delta_{p_i}(x,y,z) n_i \rightarrow \chi_V(x,y,z) = V * F$$

$$O(n^3 \log n)$$

Vector Field

Characteristic Function

Outline

- Introduction
- Related Work
- Approach
  - The Divergence Theorem
  - Reduction to Volume Integration
  - Implementation
    - Defining the vector fields
    - Weighting non-uniform samples
- Results
- Conclusion

Non-Uniform Samples

Challenge:

In a direct implementation of Monte-Carlo integration, it is assumed that the samples are uniformly distributed:

$$\int f(x)dx \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

Non-Uniform Samples

Challenge:

In a direct implementation of Monte-Carlo integration, it is assumed that the samples are uniformly distributed:

$$\int f(x)dx \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

However, often the oriented point samples may not be uniformly distributed over the surface:

- Parts of scans may overlap
- Faces parallel to the view plane may be more densely sampled
- Compressed representations may store fewer points in regions of low curvature

Non-Uniform Samples

<u>Challenge</u>:

If we have a sampling density ρ_i associated to each sample x_i, we can modify the summation:

$$\int f(x)dx \approx \frac{1}{\left(\sum_{i=1}^{n}\rho_i\right)}\sum_{i=1}^{n} f(x_i)\rho_i$$

Non-Uniform Samples

Challenge:

If we have a sampling density ρ_i associated to each sample x_i, we can modify the summation:

$$\int f(x)dx \approx \frac{1}{\left(\sum_{i=1}^{n}\rho_i\right)}\sum_{i=1}^{n} f(x_i)\rho_i$$

However, when we get an oriented point sample, we usually aren't given the sampling density at each sample.

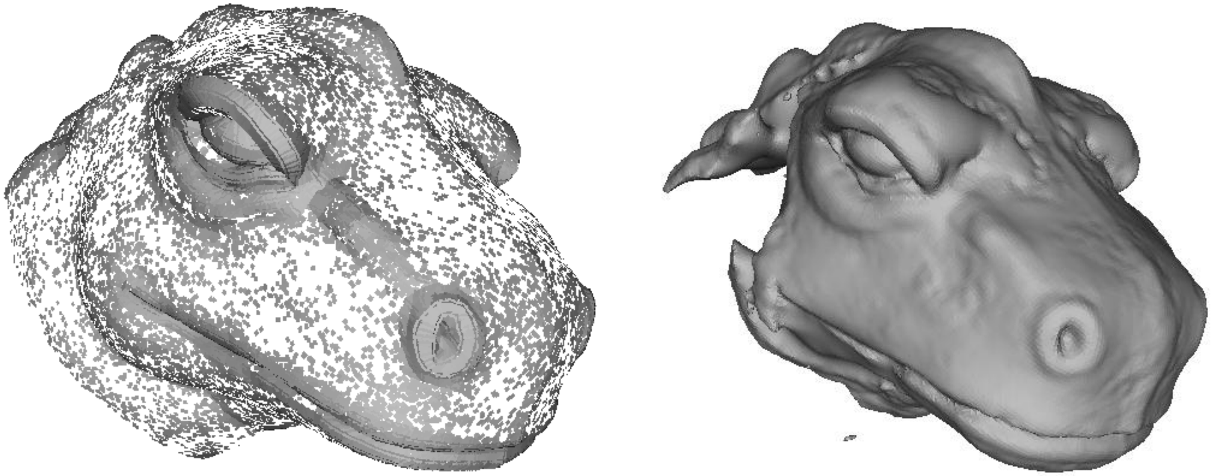
Non-Uniform Samples

Challenge:

If we have a sampling density ρ_i associated to each sample x_i, we can modify the summation:

$$\int f(x)dx \approx \frac{1}{\sum_{i=1}^{n} \rho_i} \sum_{i=1}^{n} f(x_i)\rho_i$$

How … e, we
us … ach
sa …

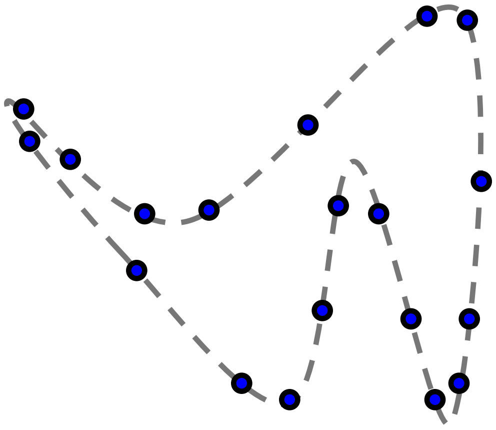


Non-Uniform Samples Unweighted Reconstruction

Non-Uniform Samples

Approach:

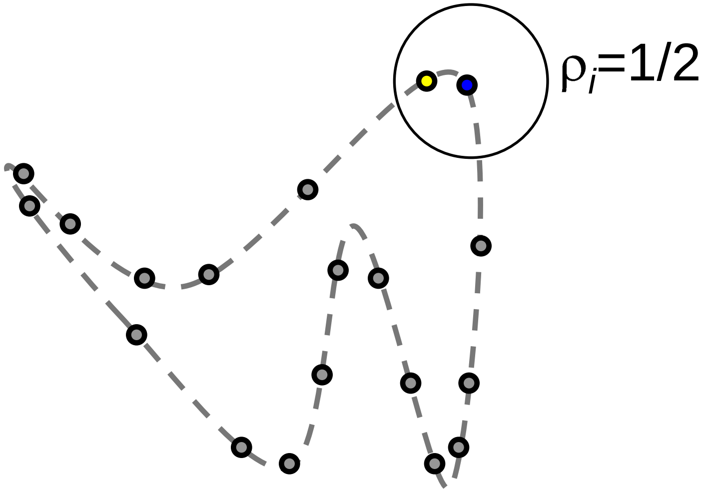
Compute the sampling density at each sample by counting the number of samples around it:



Non-Uniform Samples

Approach:

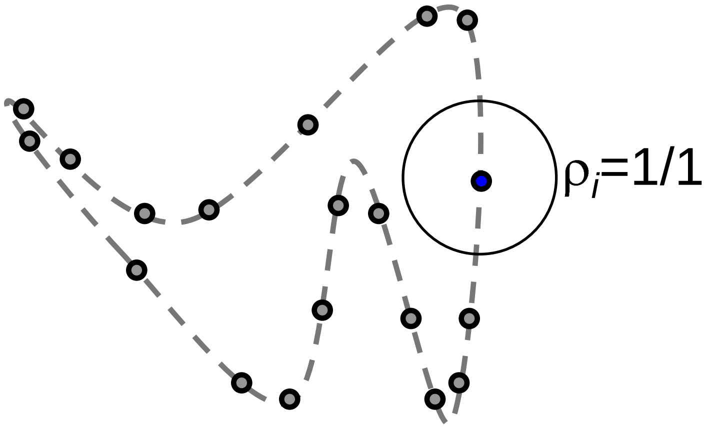
Compute the sampling density at each sample by counting the number of samples around it:



Non-Uniform Samples

<u>Approach</u>:

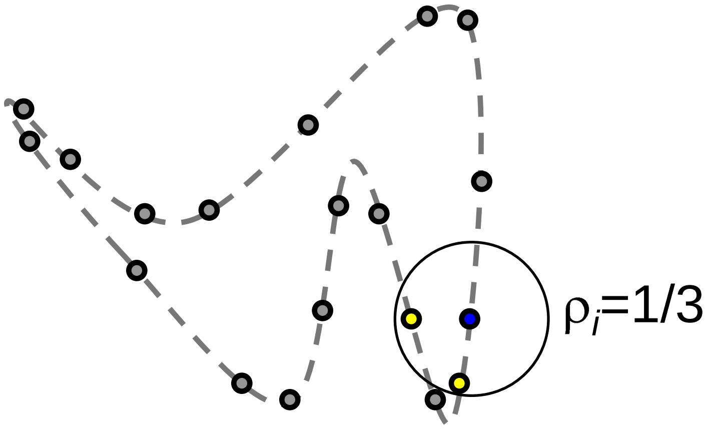
Compute the sampling density at each sample by counting the number of samples around it:



Non-Uniform Samples

<u>Approach</u>:

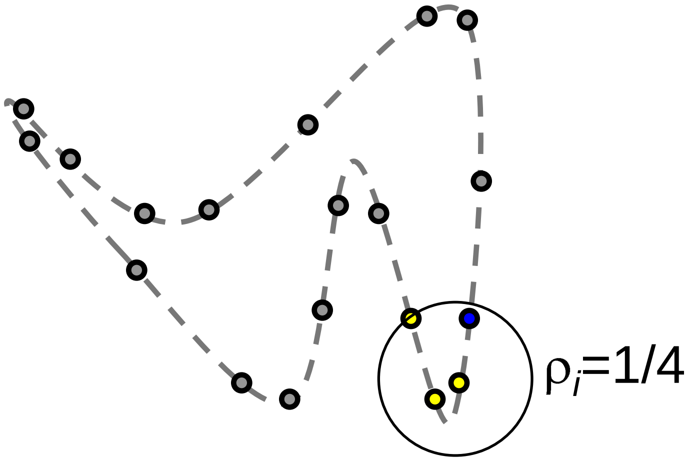
Compute the sampling density at each sample by counting the number of samples around it:



Non-Uniform Samples

Approach:

Compute the sampling density at each sample by counting the number of samples around it:

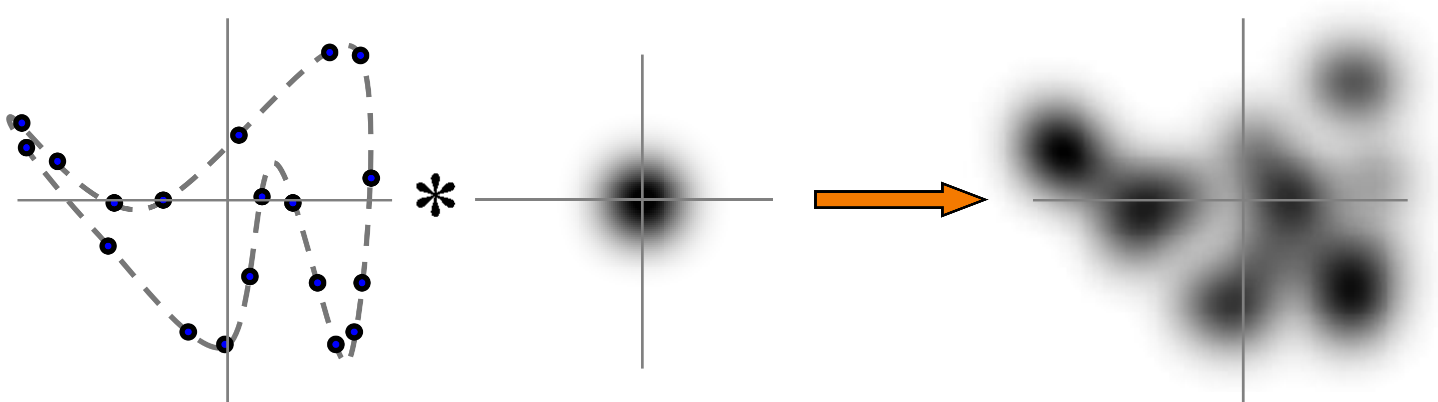


Non-Uniform Samples

<u>Approach</u>:

Compute the sampling density at each sample by counting the number of samples around it.

To do this efficiently, we “splat” the points into a voxel grid and convolve with a low-pass filter.



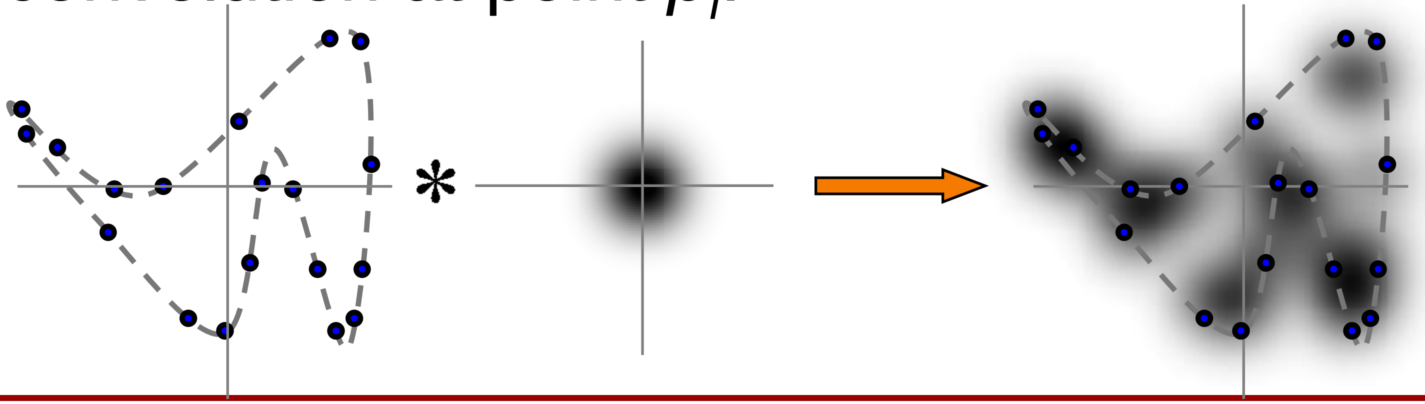
Non-Uniform Samples

<u>Approach</u>:

Compute the sampling density at each sample by counting the number of samples around it.

To do this efficiently, we “splat” the points into a voxel grid and convolve with a low-pass filter.

We set ρ_i equal to the reciprocal of the value convolution at point p_i.

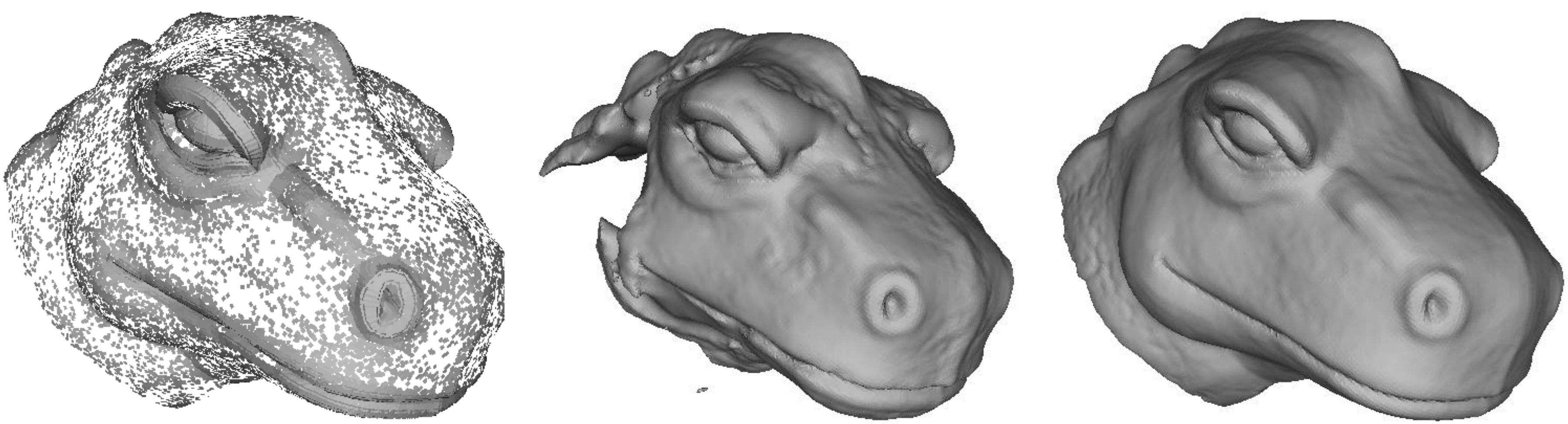


Non-Uniform Samples

Approach:

Compute the sampling density at each sample by counting the number of samples around it.

To do this efficiently, we "splat" the points into a voxel grid and convolve with a low-pass filter.

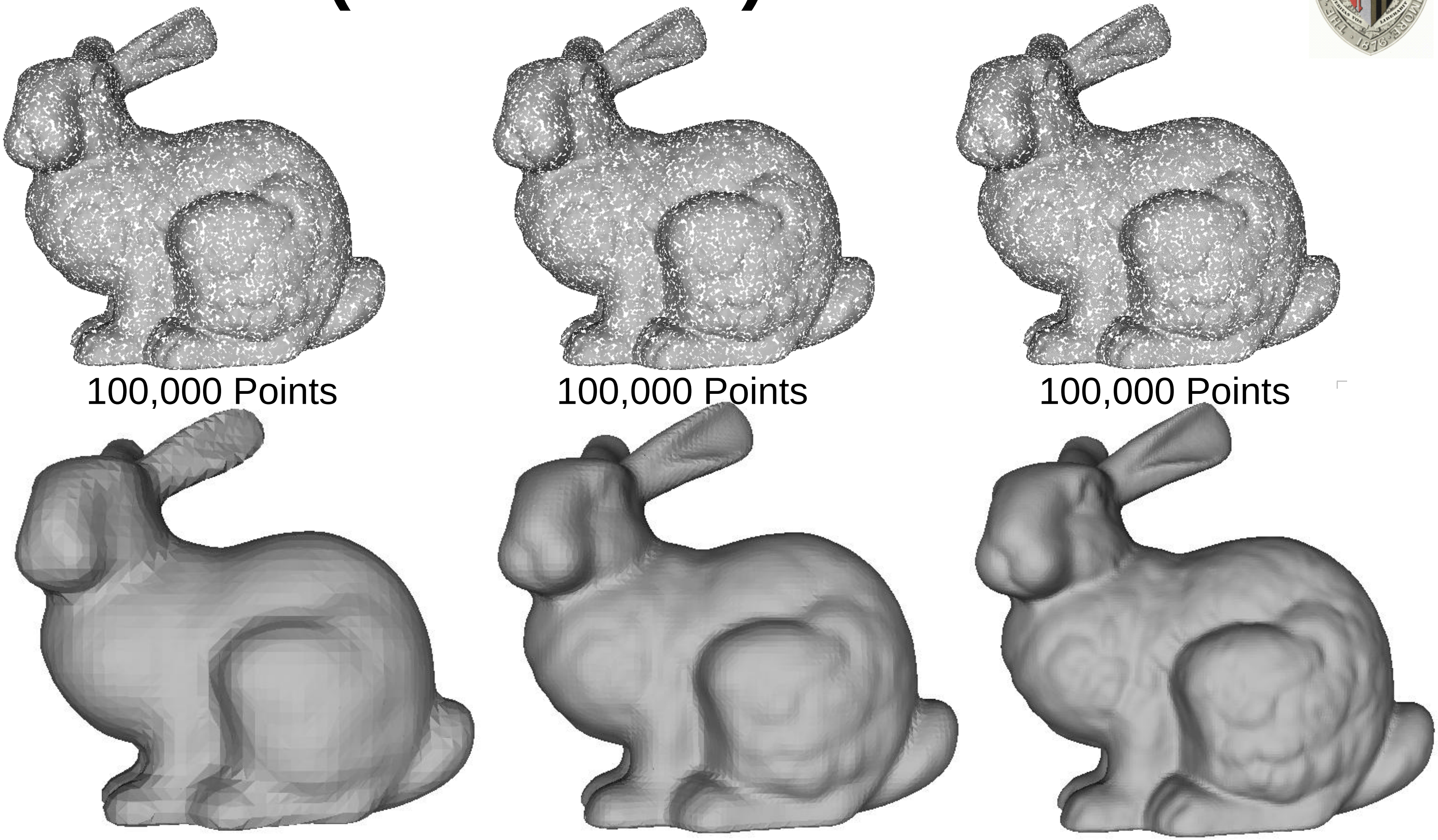


Non-Uniform Samples Unweighted Reconstruction Weighted Reconstruction

Outline

- Introduction
- Related Work
- Approach
- Results
- Conclusion

Results (Resolution)

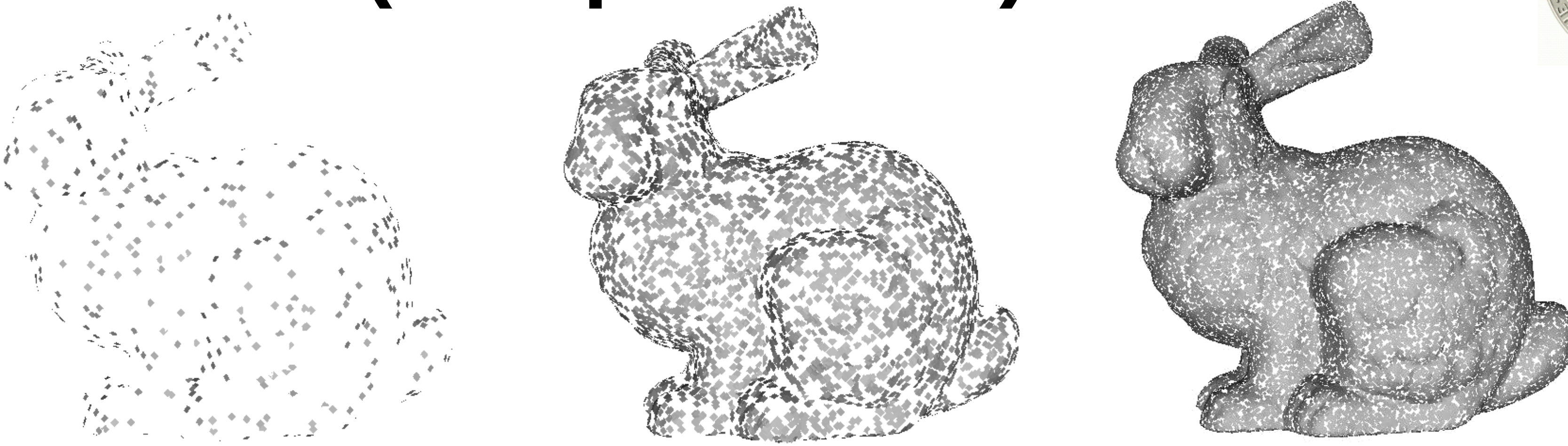


res=64^3
tris=11,900
time=0:01

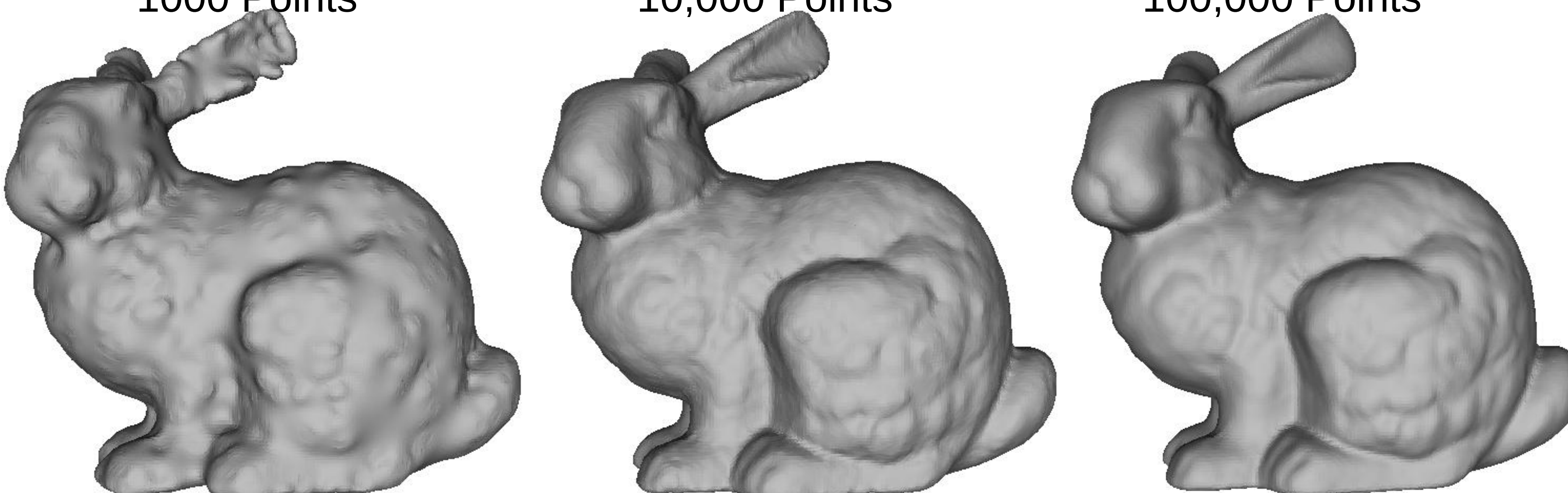
res=128^3
tris=49,556
time=0:03

res=256^3
tris=200,692
time=0:17

Results (Sample Count)



1000 Points | 10,000 Points | 100,000 Points

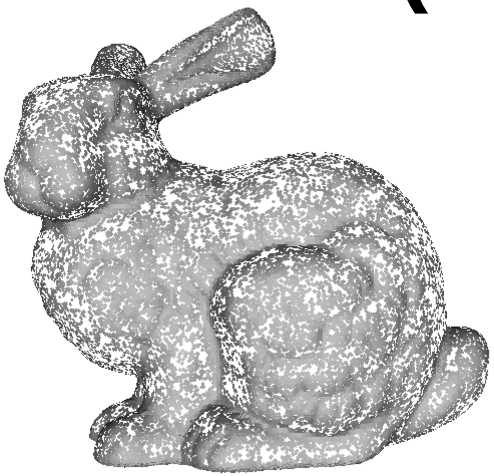


res=256^3
tris=205,440
time=0:17

res=256^3
tris=201,464
time=0:17

res=256^3
tris=200,692
time=0:17

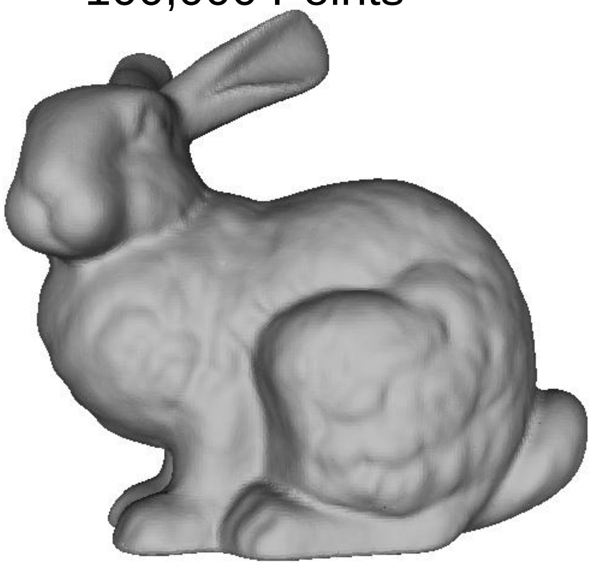
Results (Non-Uniform Sampling)



100,000 Points

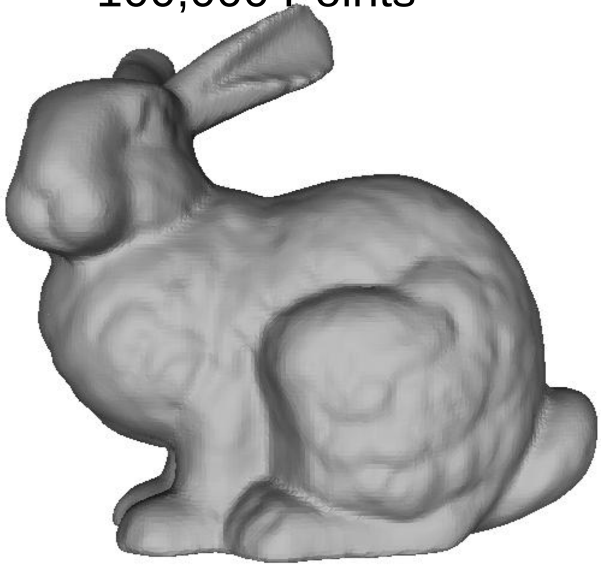
100,000 Points

100,000 Points



res=256^3
tris=209,020
time=0:20

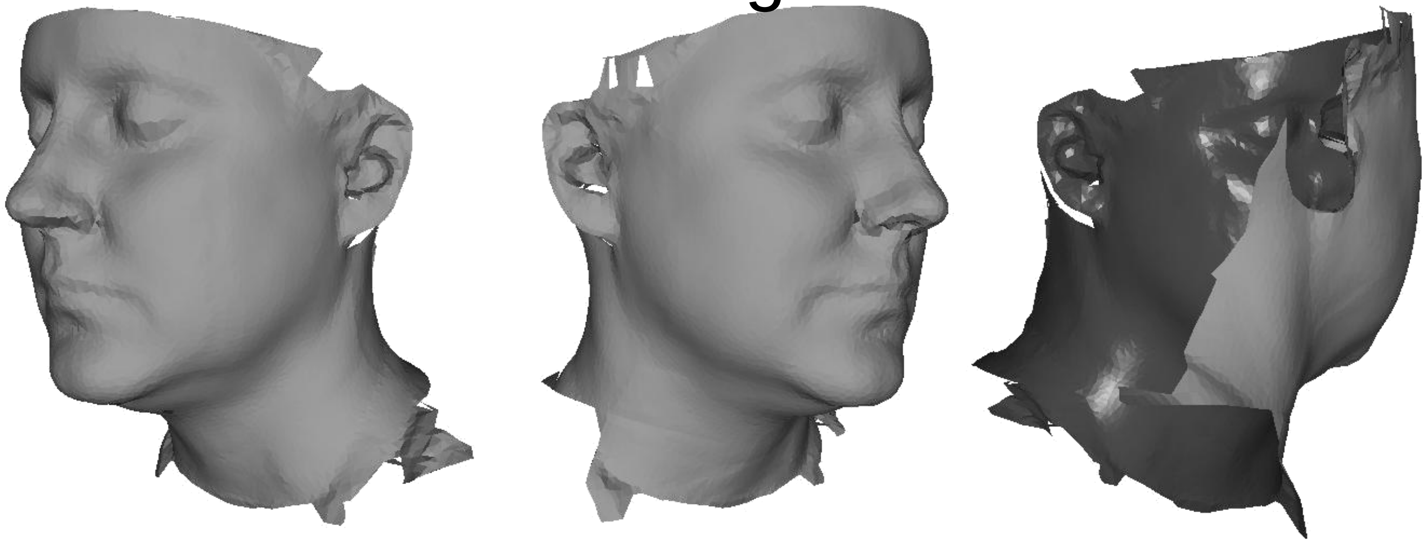
res=256^3
tris=200,692
time=0:17



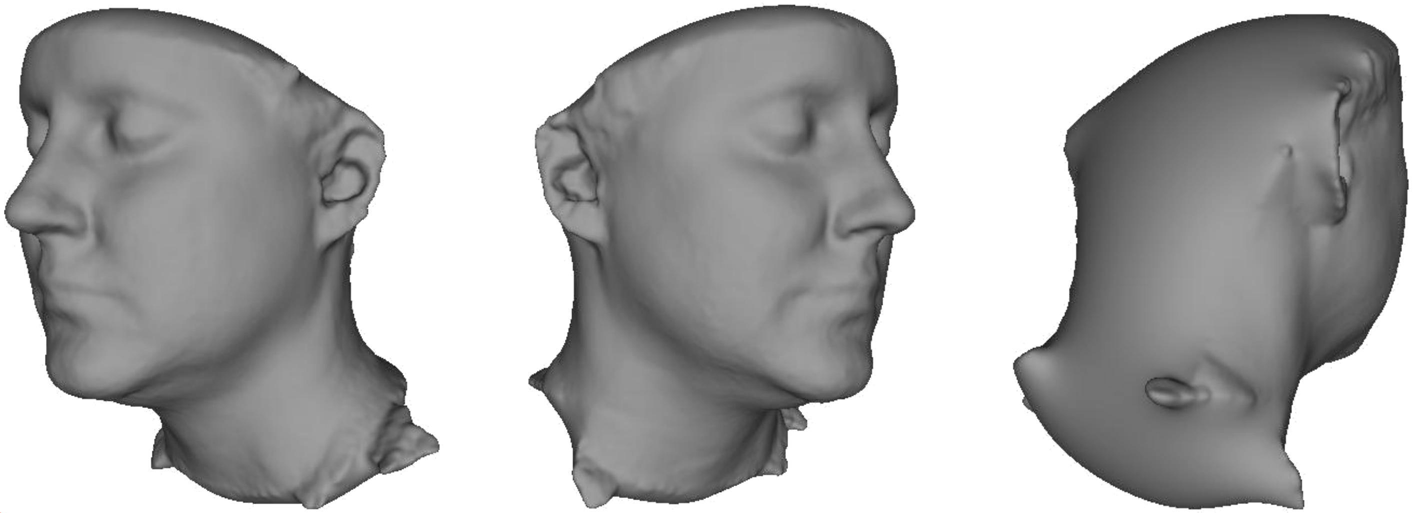
res=256^3
tris=130,440
time=0:19

Results (Holes)

Original

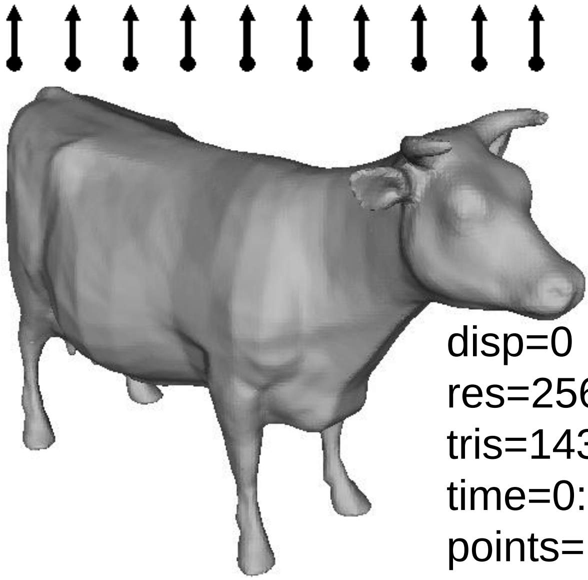


Reconstruction

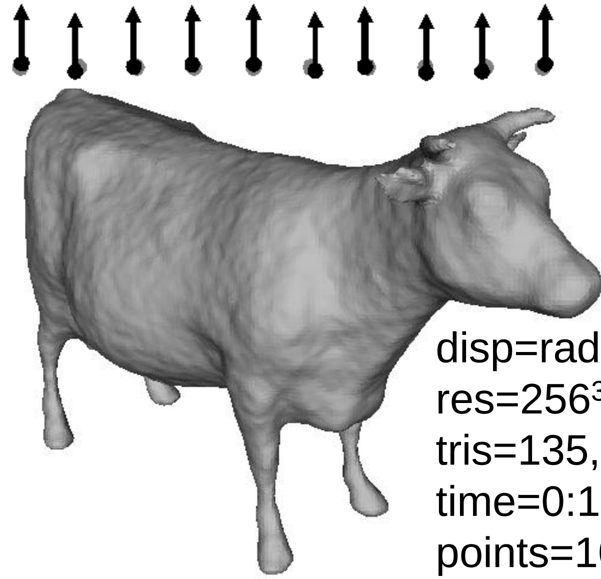


res=256^3
tris=267,736
time=0:07
points=25,000

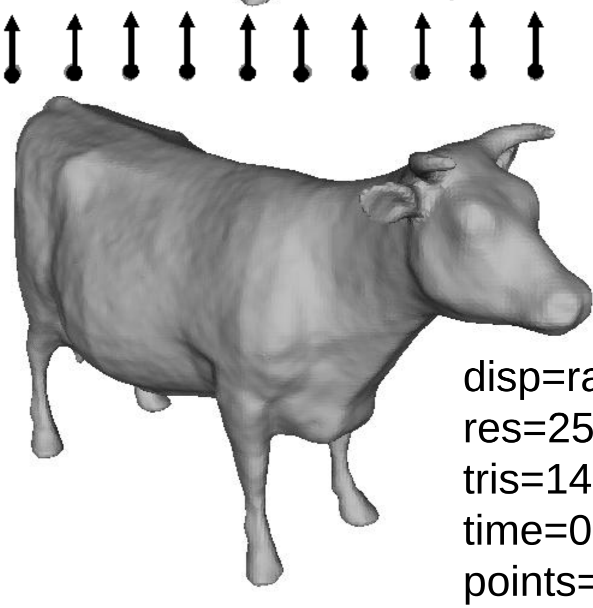
Results (Positional Noise)



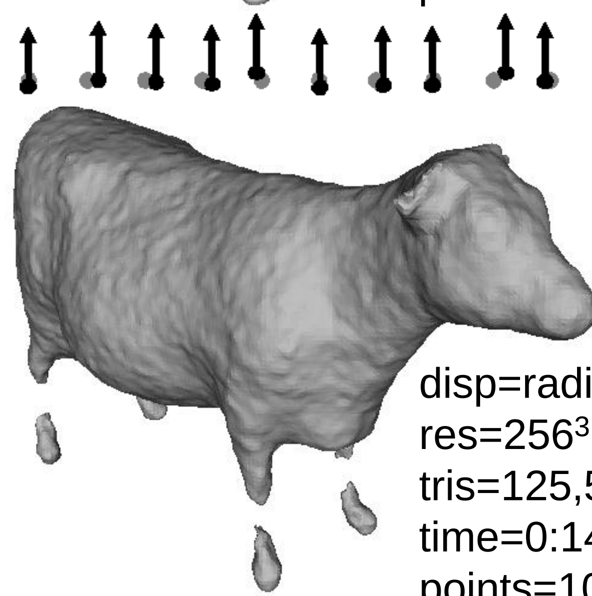
disp=0
res=256^3
tris=143,172
time=0:16
points=100,000



disp=radius/64
res=256^3
tris=135,988
time=0:15
points=100,000

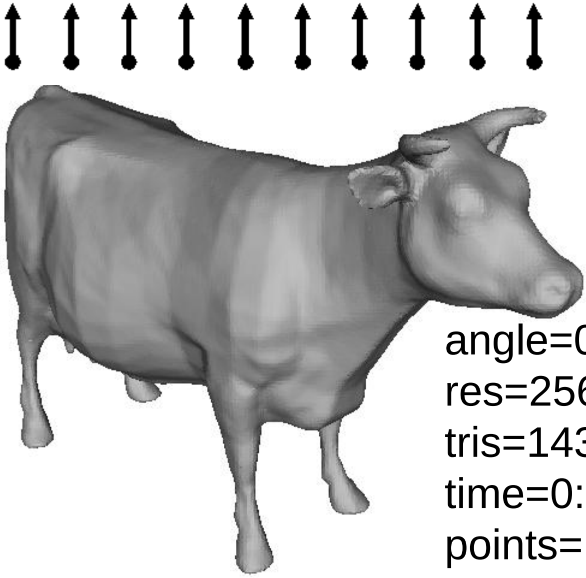


disp=radius/128
res=256^3
tris=140,688
time=0:15
points=100,000

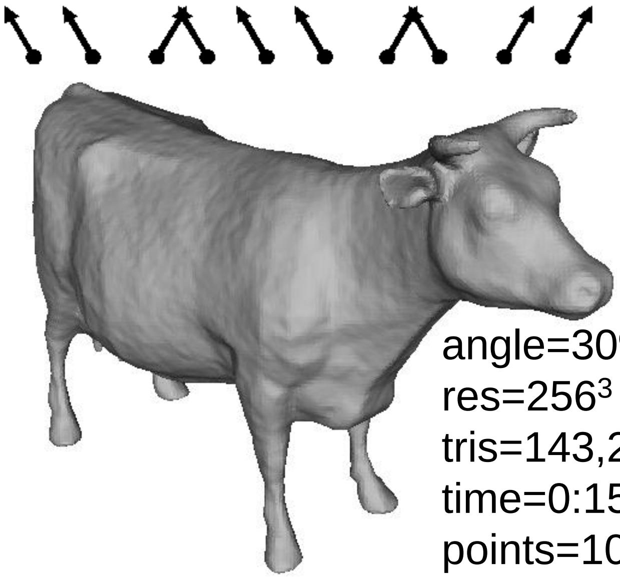


disp=radius/32
res=256^3
tris=125,500
time=0:14
points=100,000

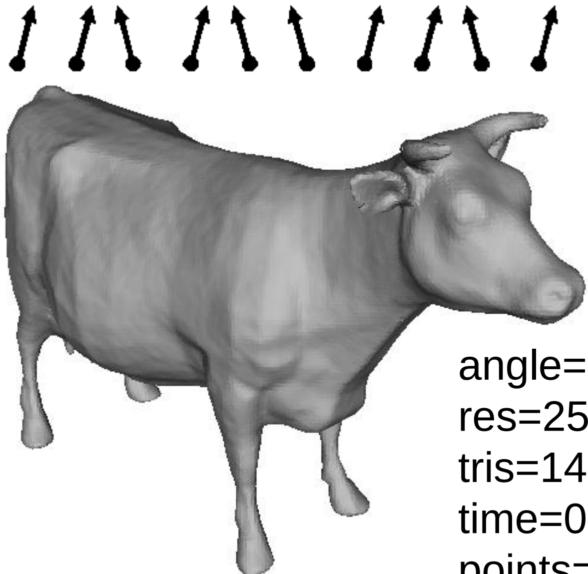
Results (Normal Noise)



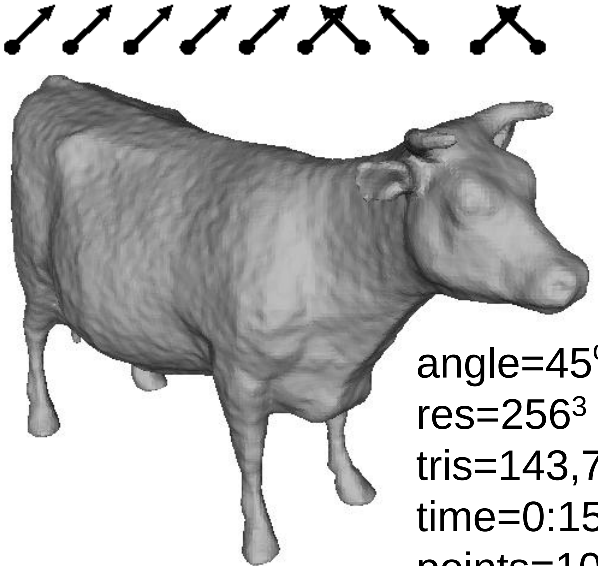
angle=0^o
res=256^3
tris=143,172
time=0:16
points=100,000



angle=30^o
res=256^3
tris=143,268
time=0:15
points=100,000



angle=15^o
res=256^3
tris=143,136
time=0:15
points=100,000



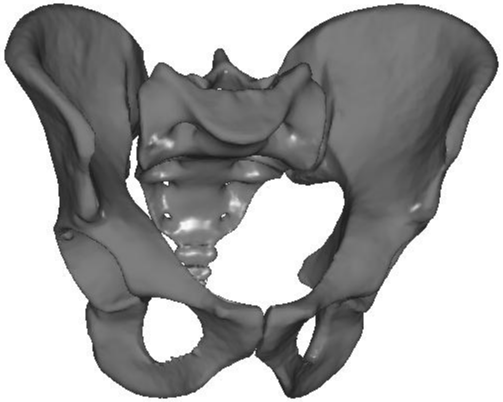
angle=45^o
res=256^3
tris=143,748
time=0:15
points=100,000

Results
(Non-Uniform Sampling / Related Work)

100,000 Points

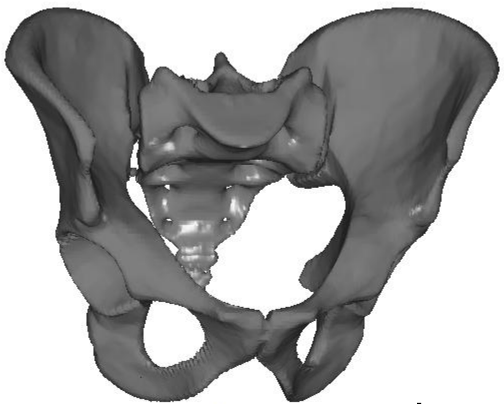
100,000 Points

100,000 Points



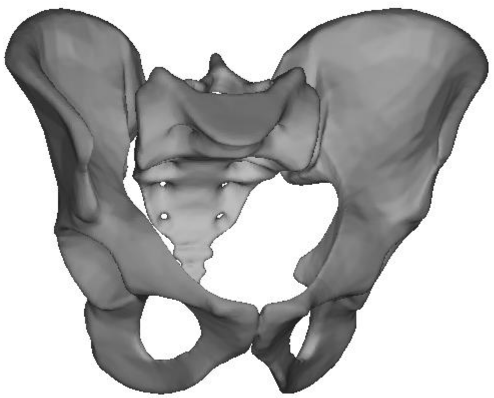
RBF Reconstruction

res=256^3
tris=302,000
time=5:23



MPU Reconstruction

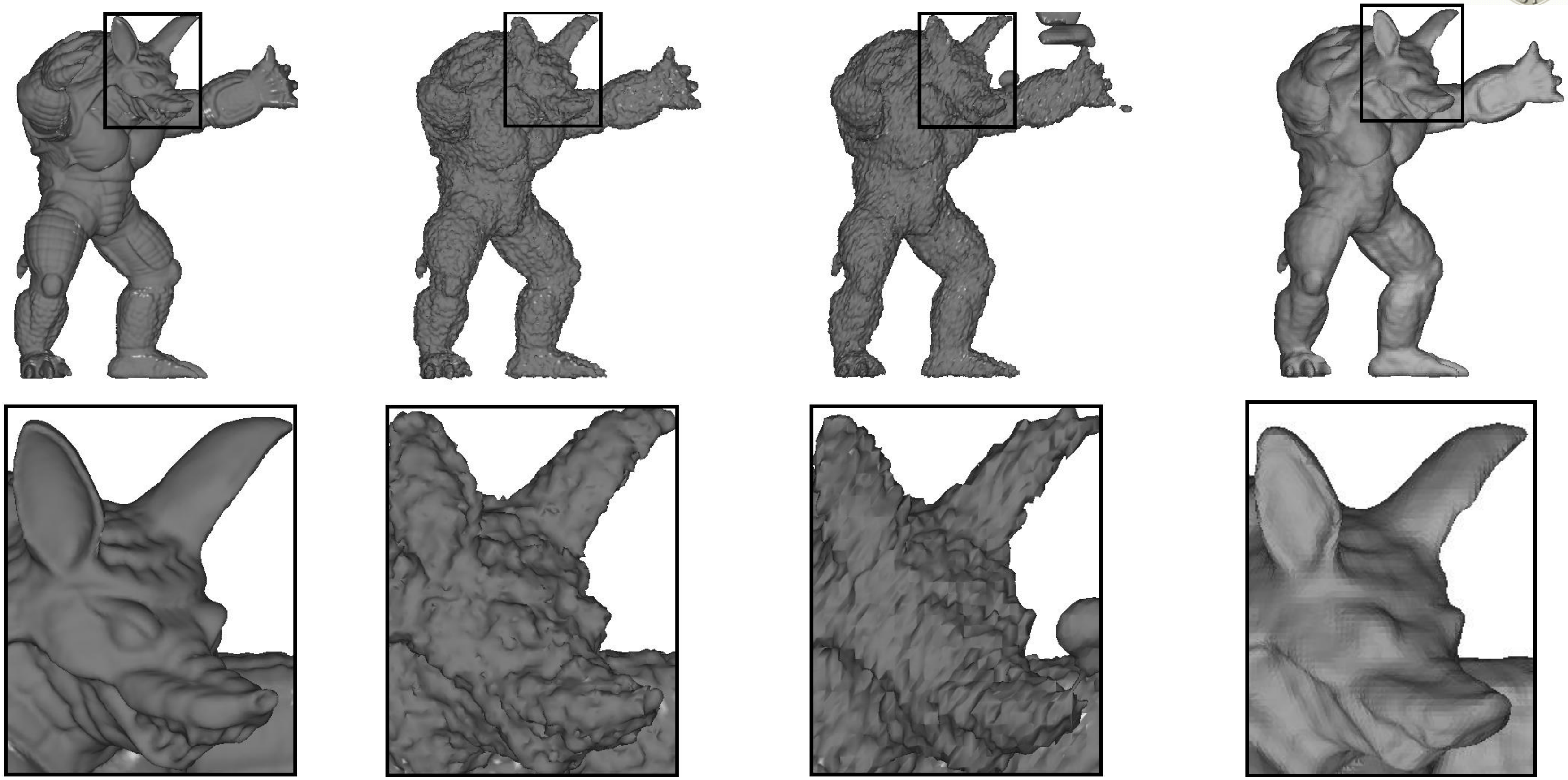
res=256^3
tris=288,000
time=0:39



Our Reconstruction

res=256^3
tris=314,000
time=0:17

Results (Noise / Related Work)



Original	RBF Reconstruction	MPU Reconstruction	Our Reconstruction
	res=256^3 tris=200,000 time=24:10 points=100,000	res=256^3 tris=205,000 time=2:14 points=100,000	res=256^3 tris=177,000 time=0:16 points=100,000

Outline

- Introduction
- Related Work
- Approach
- Results
- Conclusion

Conclusion

Properties:

✓ Fast and simple to compute

✓ Independent of topology

✓ Robust to non-uniform sampling

✓ Robust to noise

✗ O(R^3) memory footprint for O(R^2) reconstruction

Conclusion

Theoretical Contribution:

- 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

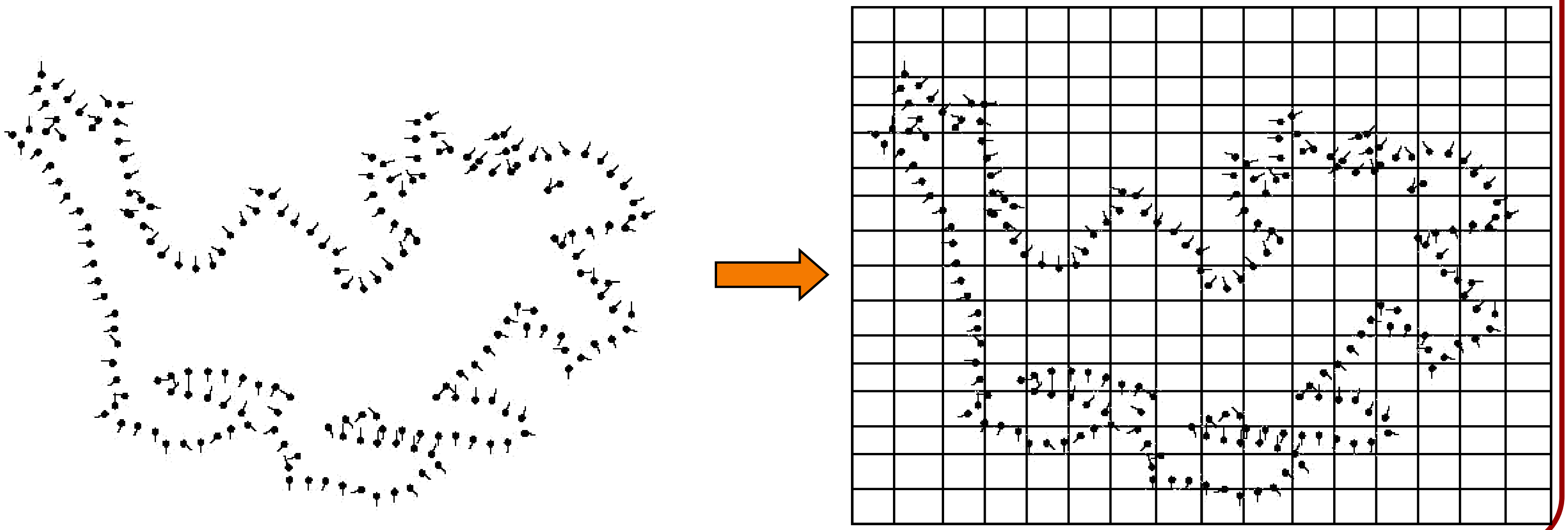
Conclusion

We presented an algorithm for reconstruction that proceeds in three simple steps:

Conclusion

We presented an algorithm for reconstruction that proceeds in three simple steps:

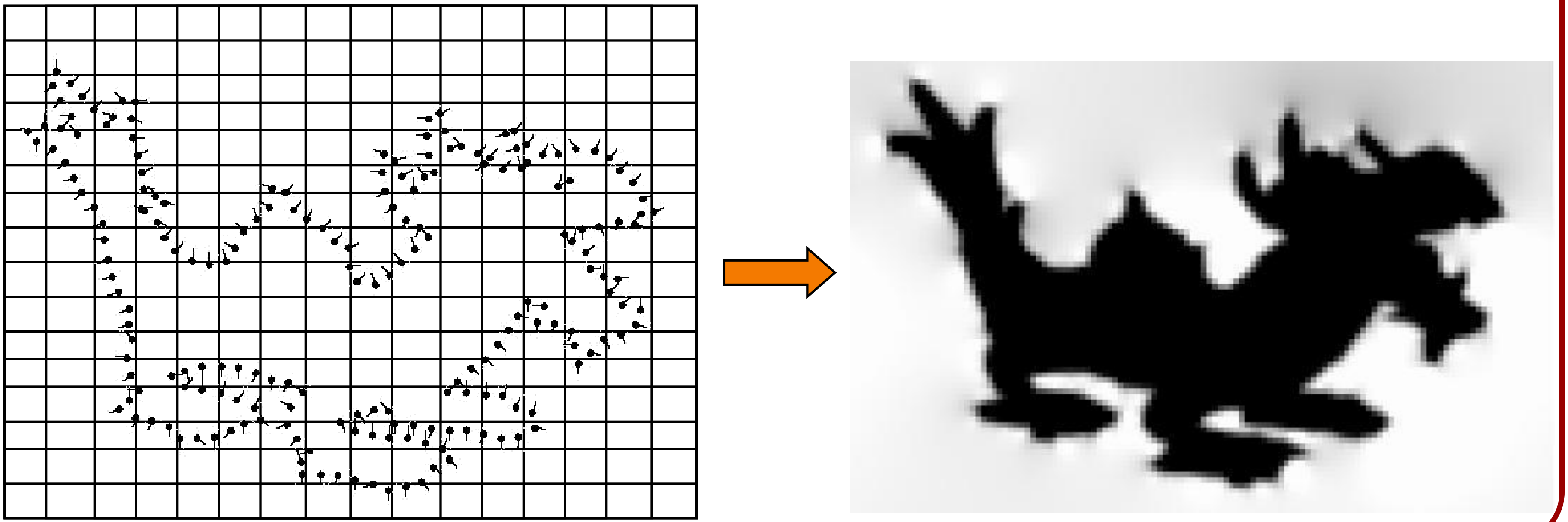
1. Splat the oriented points into a voxel grid



Conclusion

We presented an algorithm for reconstruction that proceeds in three simple steps:

1. Splat the oriented points into a voxel grid
2. Convolve with a fixed filter



Conclusion

We presented an algorithm for reconstruction that proceeds in three simple steps:

1. Splat the oriented points into a voxel grid
2. Convolve with a fixed filter
3. Extract the iso-surface