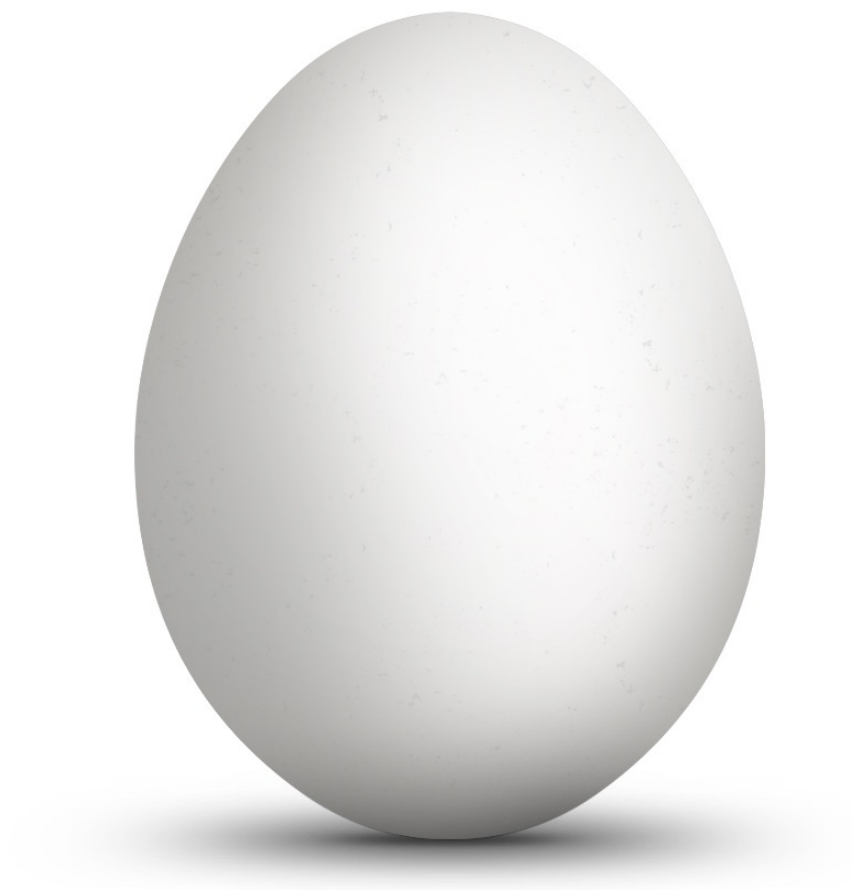
Shape Representations: Point Clouds

Siddhartha Chaudhuri http://www.cse.iitb.ac.in/~cs749

Shapes can be very different

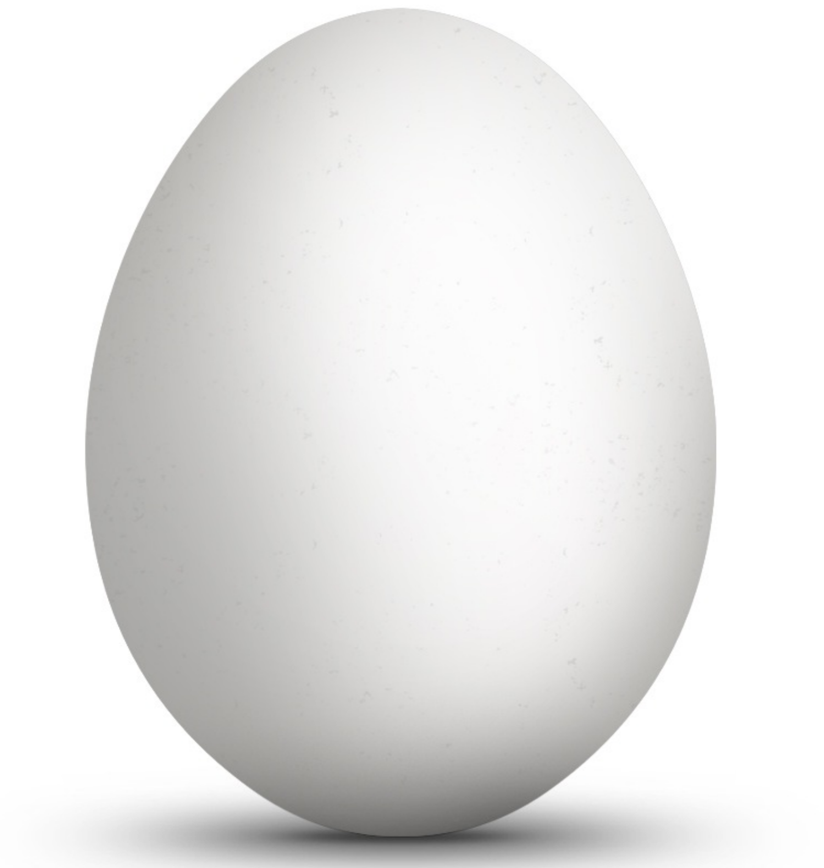


This is a shape

Shapes can be very different

This is also a shape

Shapes can be very different



Can you use the same representation for both?

Why particular representations?

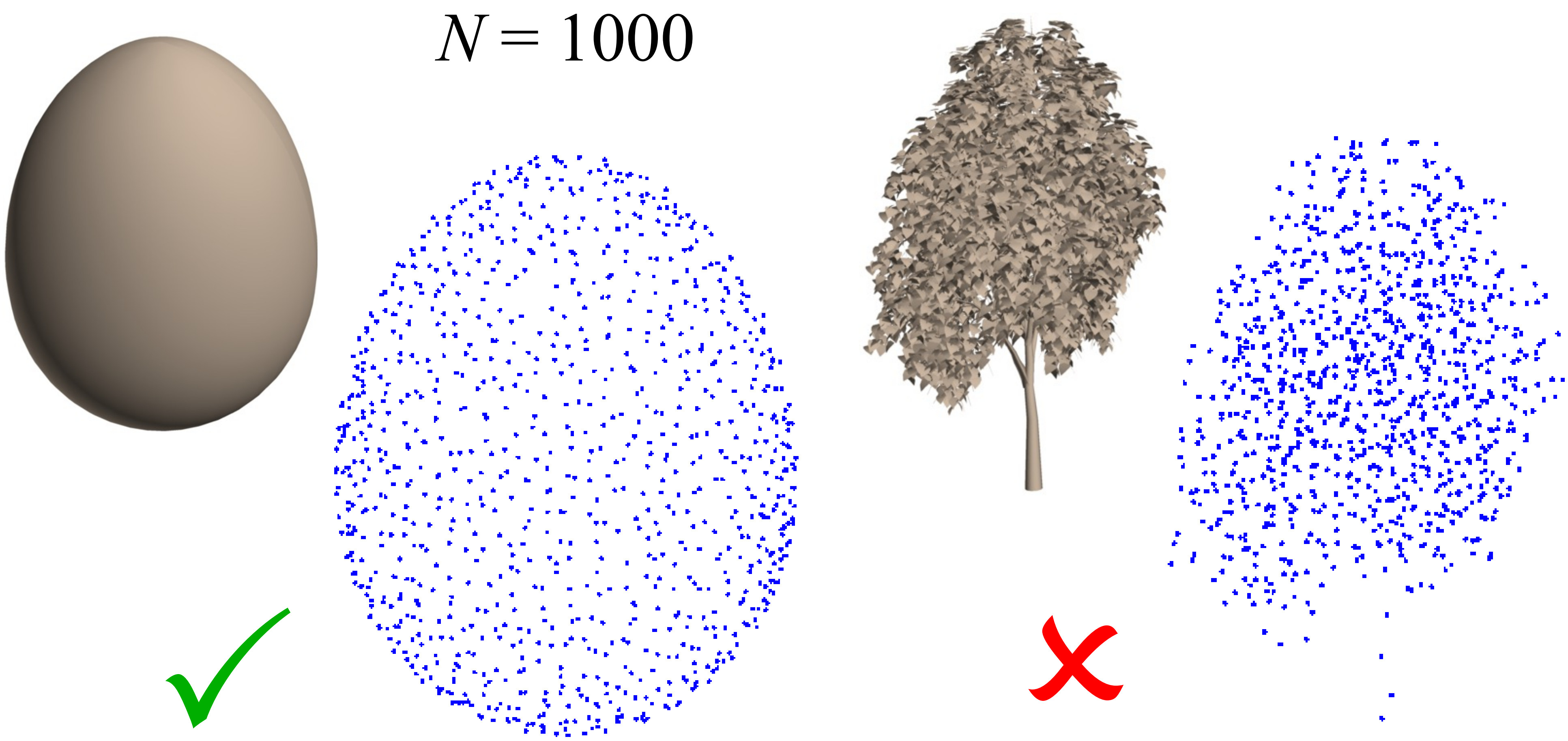
- Accuracy
- Storage
- Algorithmic efficiency
- Richness

Point Clouds

- The simplest representation of 3D shapes
- Sample N points from the surface of the shape
  - How large should N be?

How large should N be?

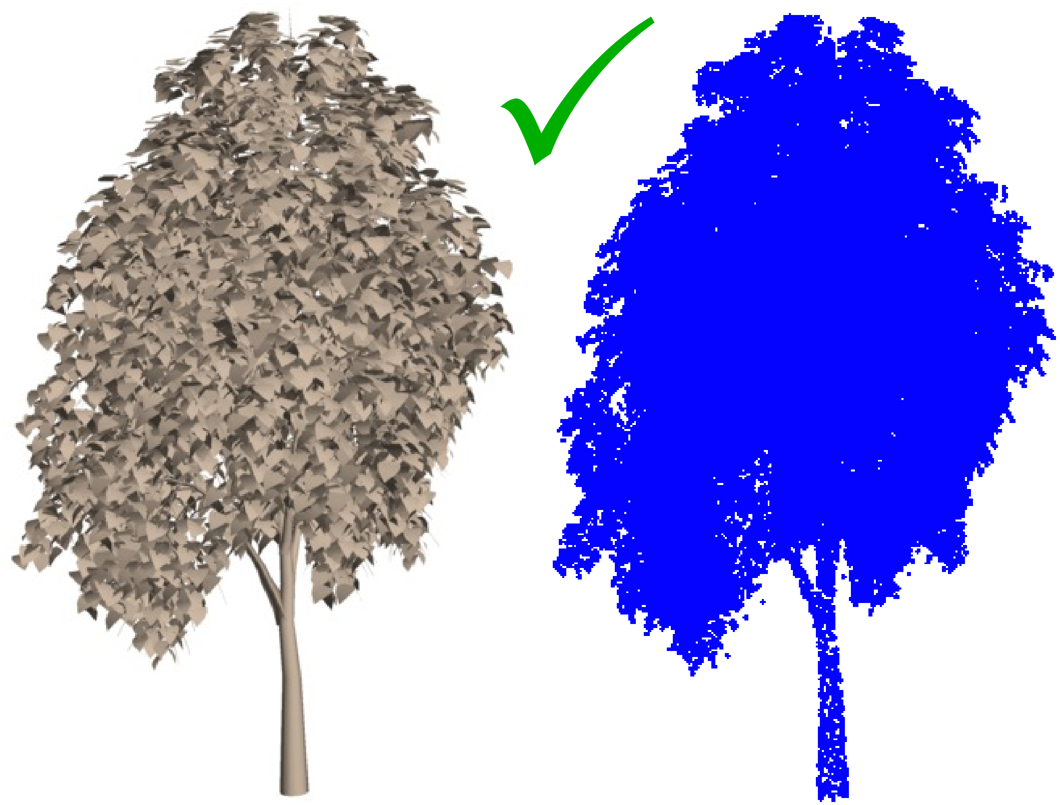
- Some big fixed number?



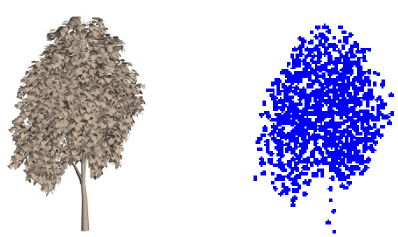
How large should N be?

- Dependent on the scale of the shape?

$N = 100000$

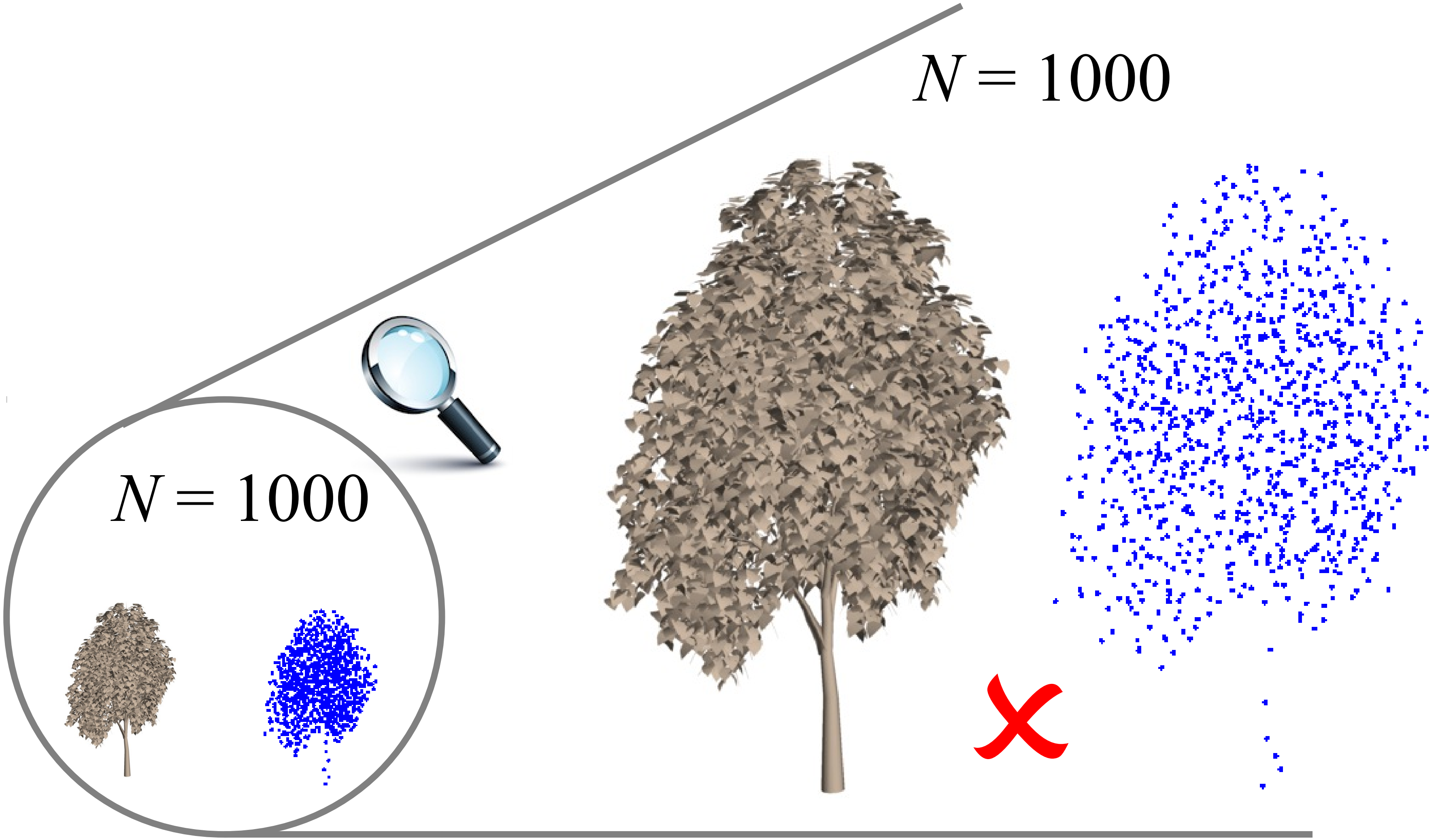


$N = 1000$



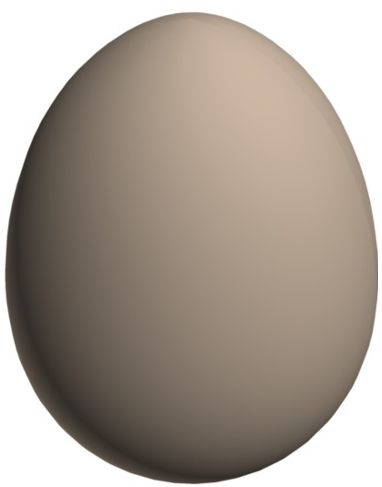
How large should N be?

- Dependent on the scale of the shape?

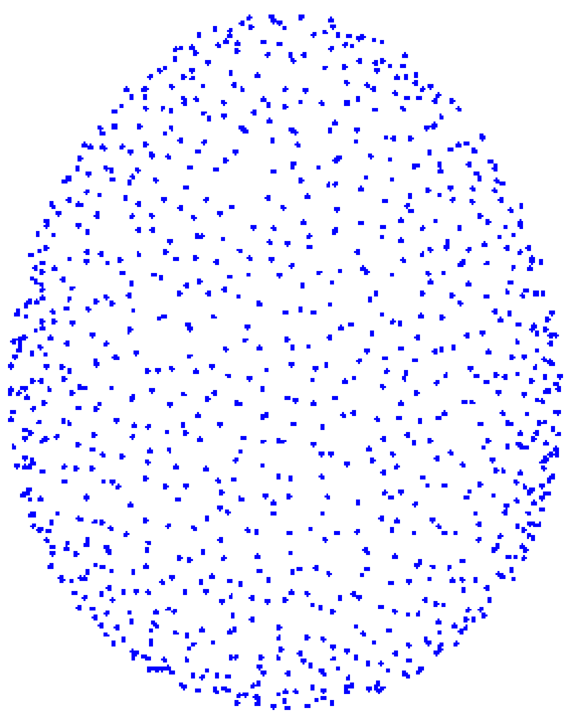


How large should N be?

- Dependent on the complexity of the shape?

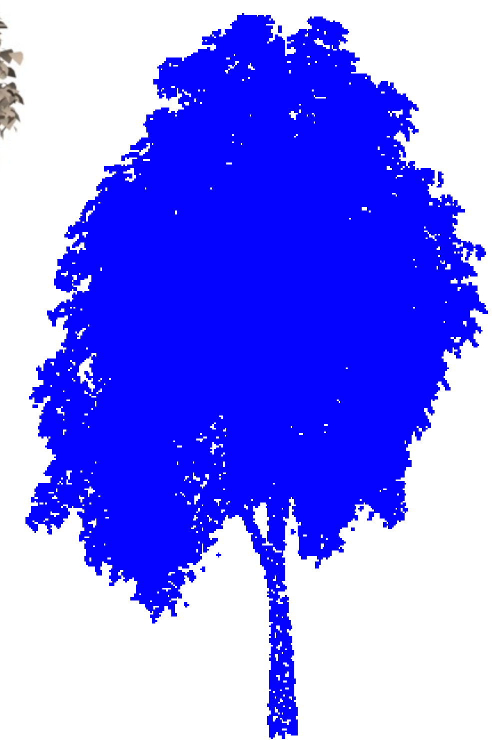


$N = 1000$



✓

$N = 100000$

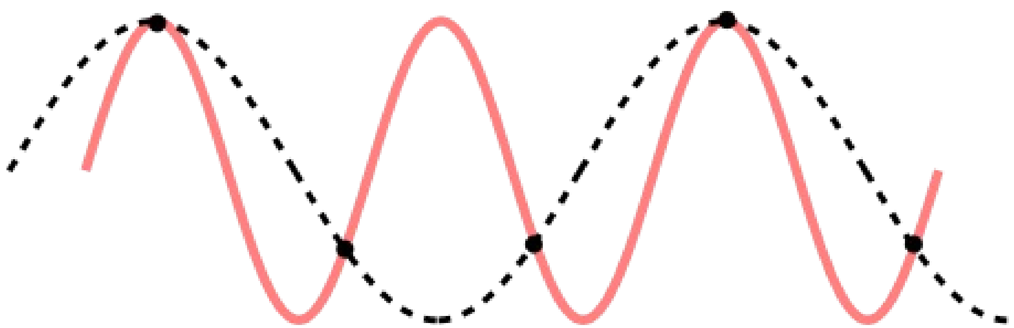


✓

How can we characterize complexity?

- Nyquist-Shannon Sampling Theorem

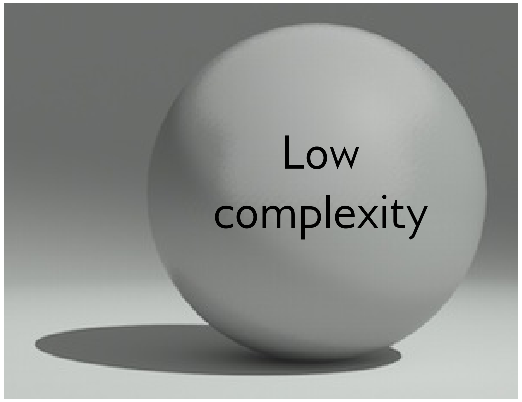
  If a function $x(t)$ contains no frequencies higher than B hertz, it is completely determined by samples spaced $1/2B$ apart

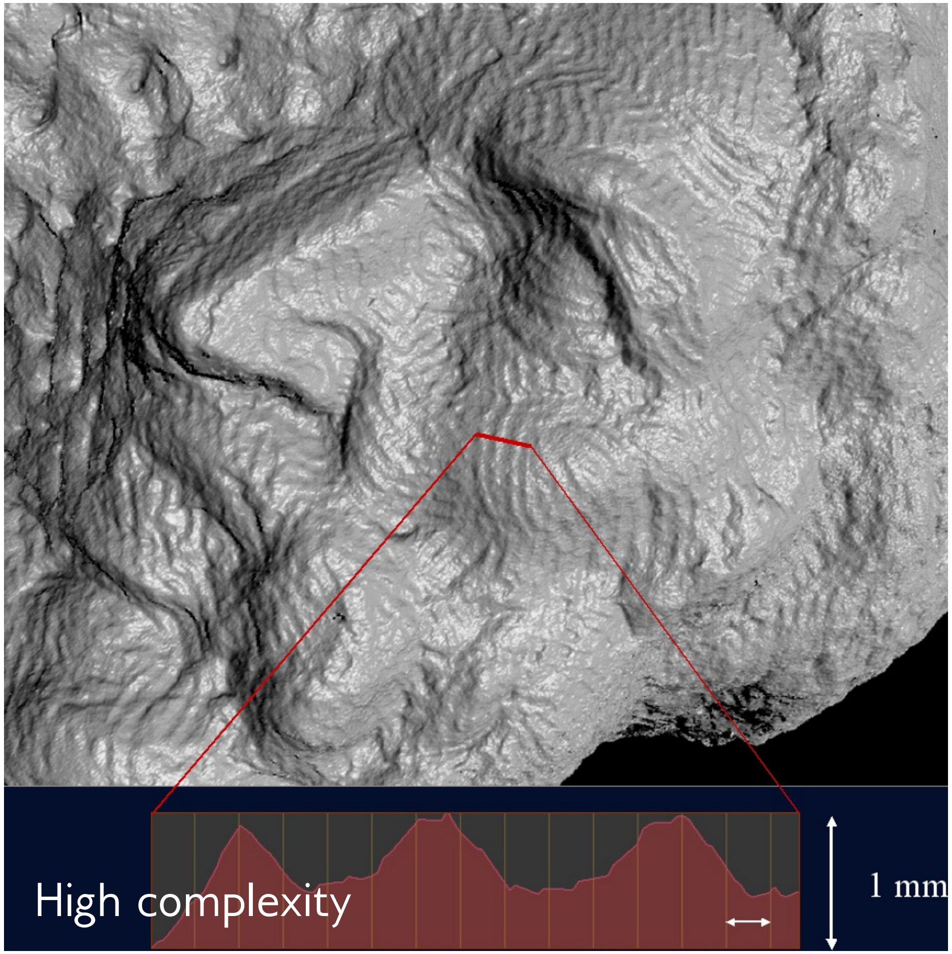


- What is the “frequency” of a shape?
  - Intuitively: lots of fine detail → high frequency

How can we characterize complexity?

- What is the "frequency" of a shape?
  - Intuitively: lots of fine detail → high frequency
  - We will revisit this when we study spectral decompositions



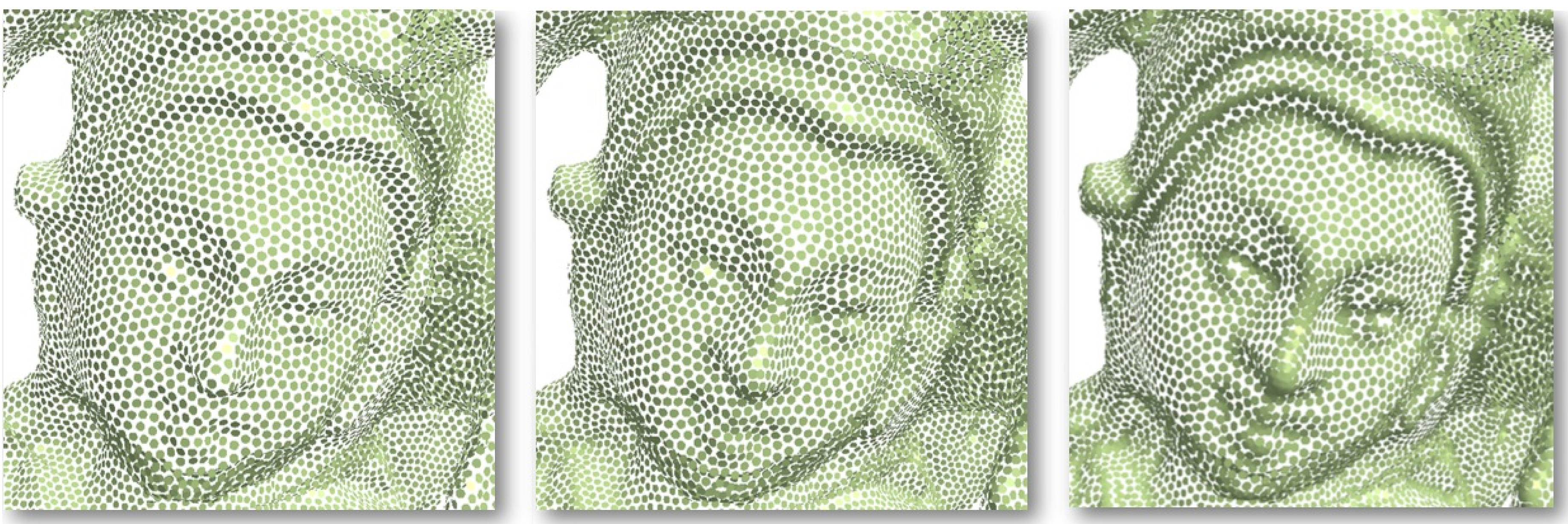


Digital Michelangelo Project

Thought for the Day #1

Do we need the same sampling rate everywhere on the shape?

No, we can do adaptive sampling!



Öztireli, Alexa, and Gross, "Spectral Sampling of Manifolds", SIGGRAPH Asia 2010

Thought for the Day #2

Can you predict the complexity of a shape, without knowing the shape itself?

It depends on what you mean by “knowing”, and how much confidence you want in your prediction

A simple storage format

X, Y, Z coordinates of points, one triplet per line

```
-1.67671     9.06038    -2.40807
-4.81769     6.48015     0.012154
-2.80832    10.0916      3.70869
-1.27393    13.9169      0.889338
 0.532515   15.1396     -0.103159
 1.31195     8.38292    -0.486781
-1.92718    13.8969     -1.19995
-0.489696    7.1351     -1.51946
-3.02698    10.7924      1.28467
```

A simple storage format

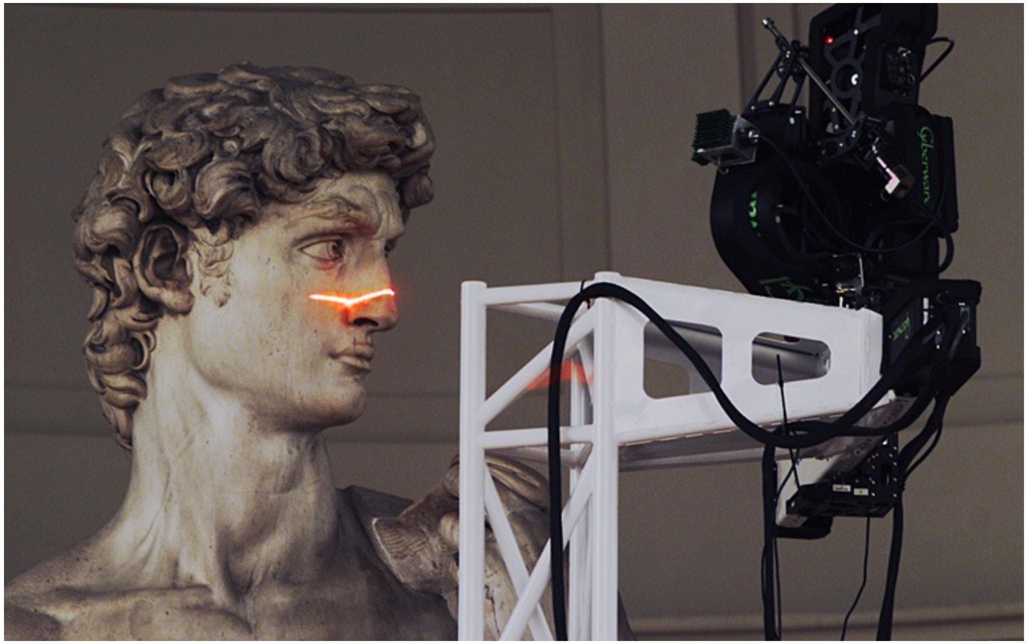
X, Y, Z coordinates of points, one triplet [+ more] per line

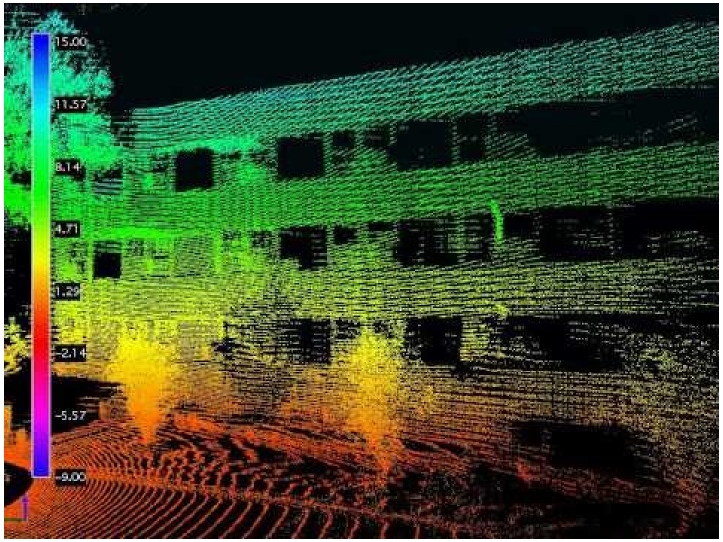
```
-1.67671     9.06038    -2.40807     0.2323   1.2943   -1.23
-4.81769     6.48015     0.012154    0.3234   0.8473    0.57
-2.80832    10.0916      3.70869    -0.6799   0.4434   -0.34
-1.27393    13.9169      0.889338
 0.532515   15.1396     -0.103159
 1.31195     8.38292    -0.486781
-1.92718    13.8969     -1.19995
-0.489696    7.1351     -1.51946
-3.02698    10.7924      1.28467
```

Additional per-point fields
(color, normal, features...)

Acquiring point clouds

- From the real world
  - 3D scanning
    - Telltale characteristic: data is “striped”
    - Need multiple views to compensate for occlusion



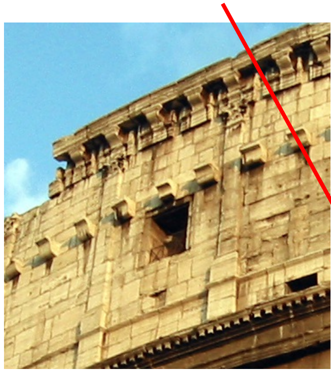


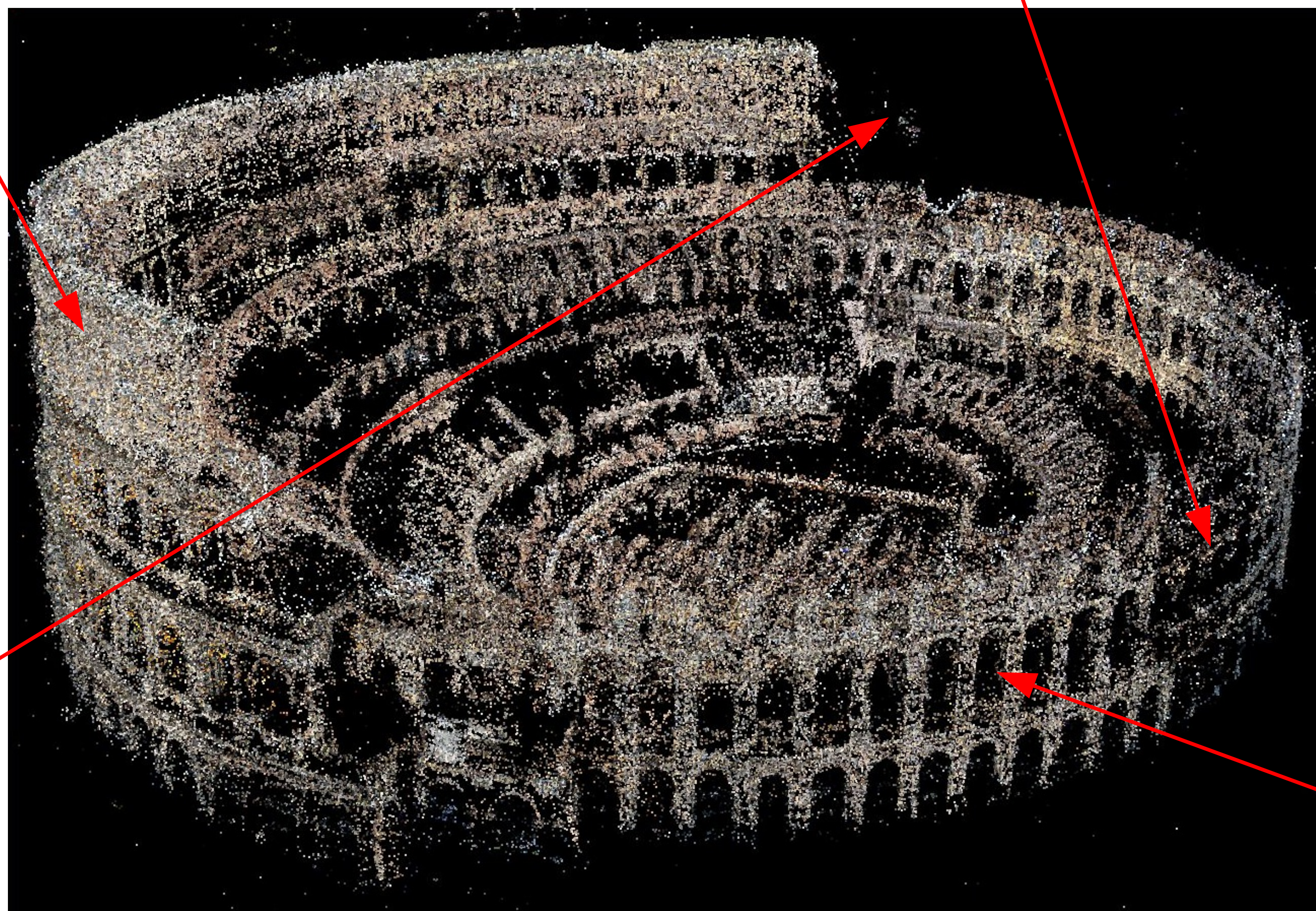
- Many technologies
  - Laser (LIDAR, e.g. Streetview)
  - Infrared (e.g. Kinect)
  - From a collection of photographs (Photosynth, Bundler)
- Many challenges: resolution, occlusion, noise, registration

Acquisition Challenges

Noise → Poor detail reproduction

Low resolution further obscures detail





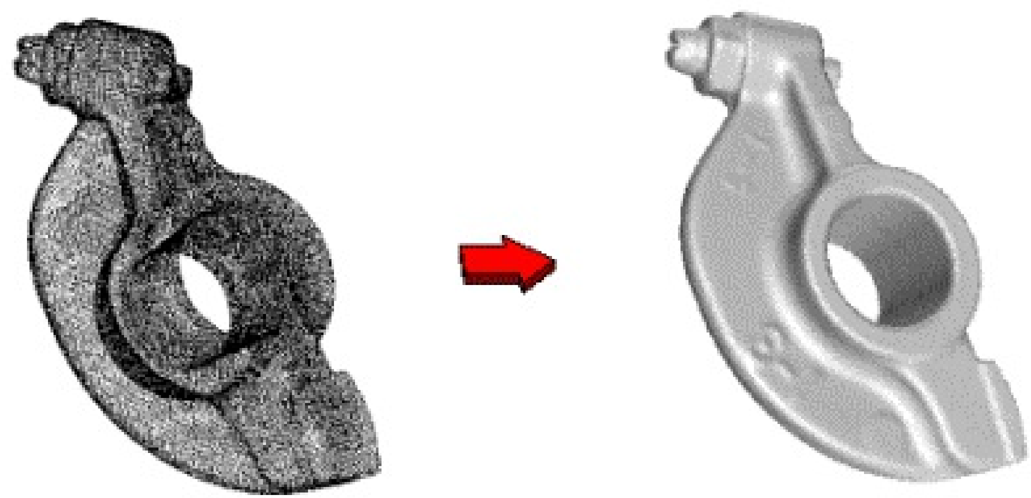
Some data was not properly registered with the rest

Occlusion → Interiors not captured

http://grail.cs.washington.edu/

How can we do better?

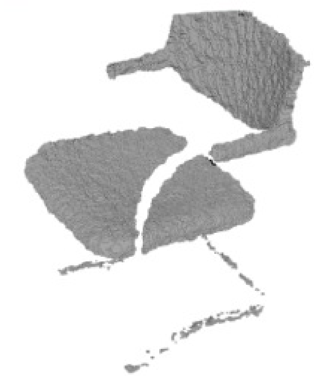
- More data, better hardware
  - Difficult because of cost and limited human resources
- Apply geometry ***priors*** to reconstruct original surface from noisy, low-resolution scan

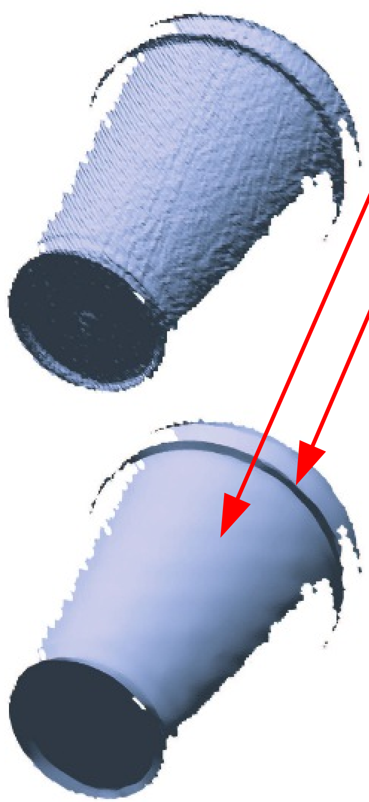


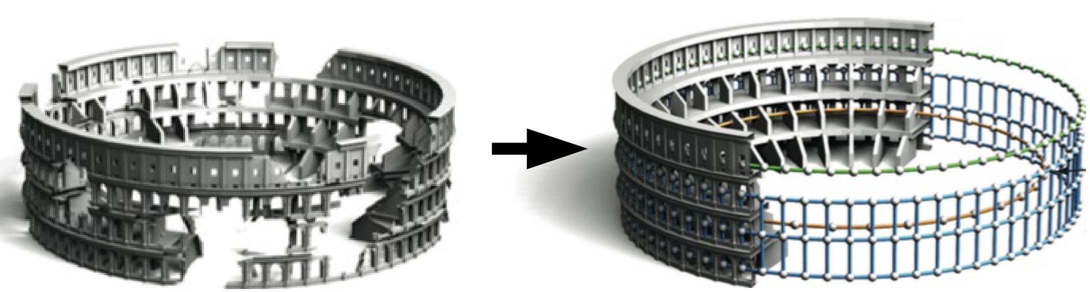
Point cloud data　　Reconstructed surface

Geometry Priors

- Prior: (Typically statistical) model of how the true data is expected to look. E.g.
  - Surface must be smooth
  - Edges must be sharp
  - Detail must be repetitive
  - Shape must resemble exemplars







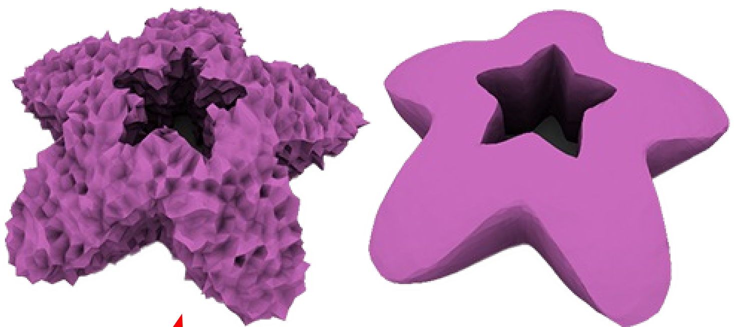
Thought for the Day #2, Revisited

Can you predict the complexity of a shape, without knowing the shape itself?

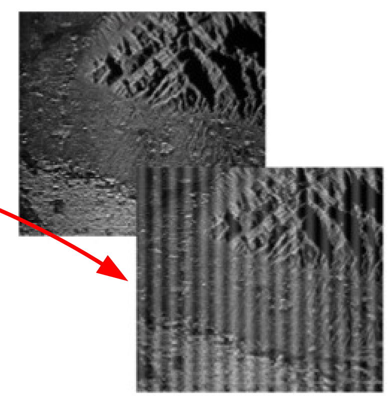
We can make an intelligent guess, using a prior

Some notes on noise

- Noise: Any deviation of the sampled point data from the true surface
  - Can be random (often assumed to be high-frequency Gaussian)
    - A simple smoothing filter helps, but blurs sharp edges and fine detail. Methods like bilateral filtering do better.



  - … or structured/systematic
    - e.g. because of limitations in scanner resolution or calibration, or mount oscillation, or genuine bugs

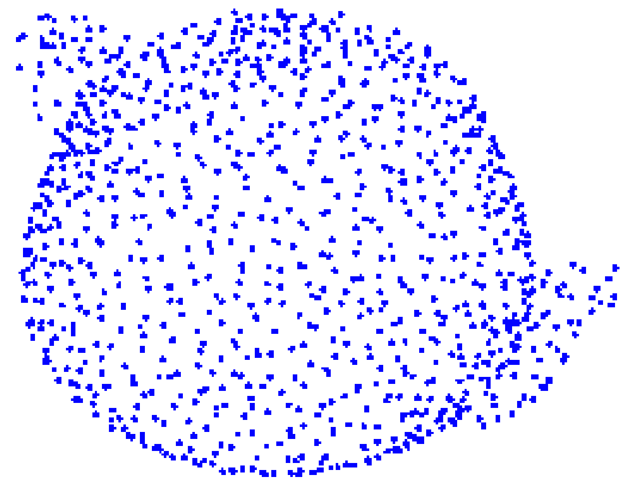


- Reconstruct true signal by removing noise
  - Requires a prior on the true geometry and/or a prior on the structure of the noise (e.g. noise is gaussian, or periodic)

Solomon et al., "A General Framework for Bilateral and Mean Shift Filtering", 2014

Acquiring point clouds

- From existing virtual shapes



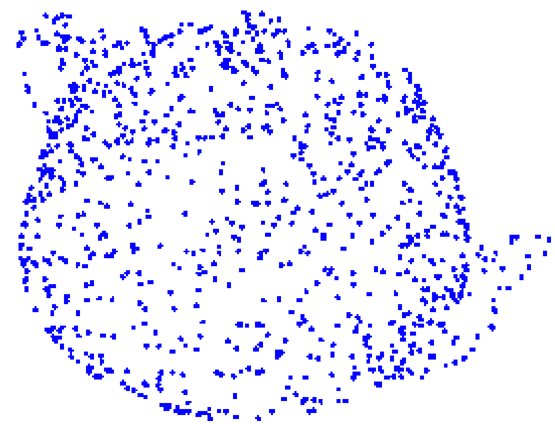
- Why would we want to do this? Don't we already have a better representation of the shape?

Thought for the Day #3

When is a point cloud preferable to more sophisticated/accurate representations?

Sampling points from a shape

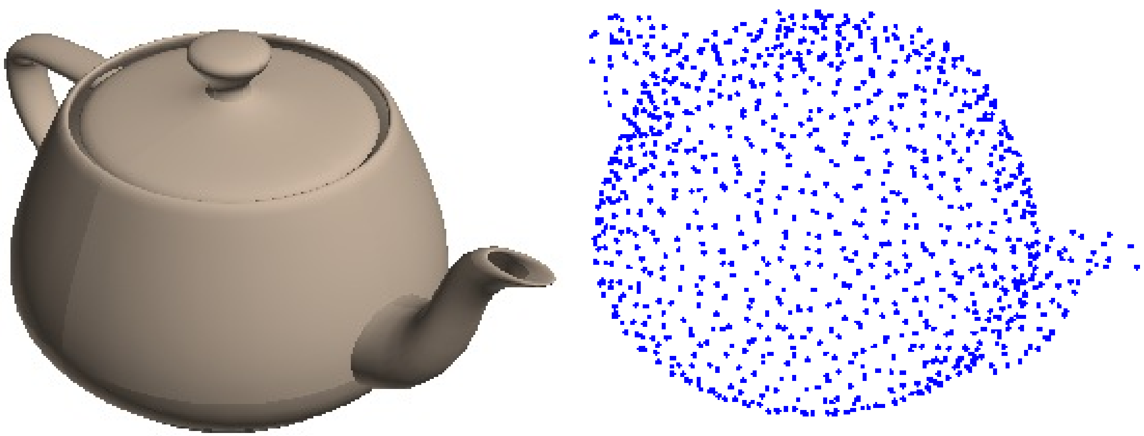
- ***Method 1:*** Independent identically distributed (i.i.d.) samples, by surface area

- Usually the **easiest** to implement:
  1) **Pick** surface element (e.g. mesh triangle) with probability proportional to its area
  2) **Sample** a point uniformly from the element
- ***Problem:*** Irregularly spaced sampling

Sampling points from a shape

- ***Method 2:*** Rejection sampling – reject and re-pick the $k + 1^{\text{th}}$ sample if it is too close to the previous k points

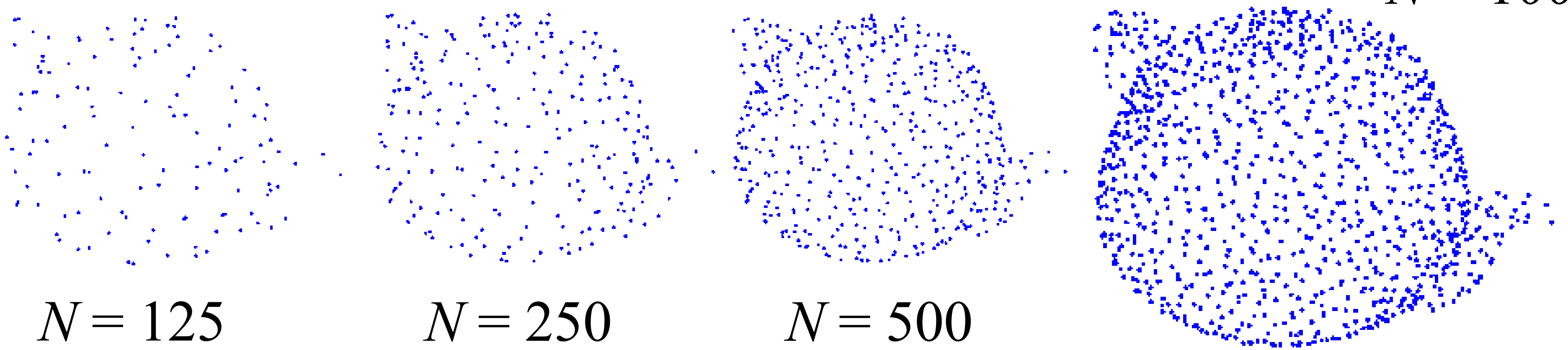


- Also very **easy** to implement
- ***Problem:*** Need to pick spacing threshold perfectly
  - If too big, impossible to pick a large number of points
  - If too small, points are not regularly spaced

Sampling points from a shape

- ***Method 3:*** Furthest point sampling – pick the $k+1^{\text{th}}$ sample as the point furthest from any of the previous k points
- Gives **good results**
- Any **prefix** of the sequence is also regularly spaced!
  - Great for quick downsampling

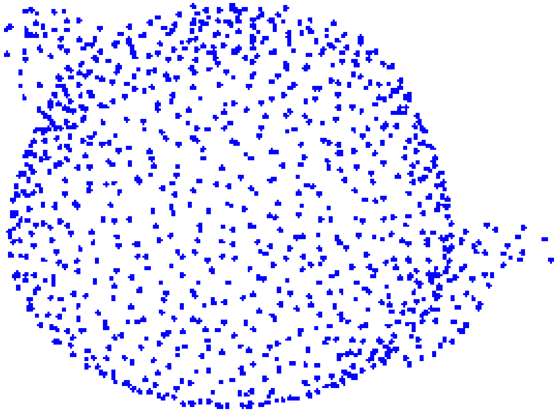
$N = 1000$



$N = 125$ $N = 250$ $N = 500$

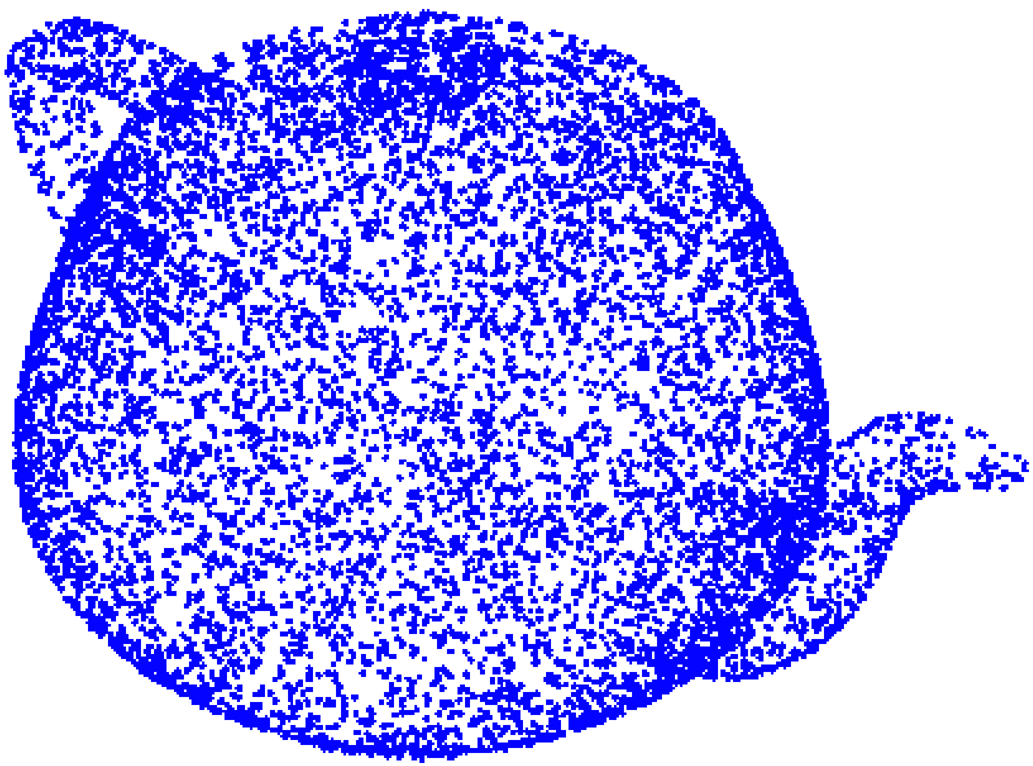
Sampling points from a shape

- ***Method 3:*** Furthest point sampling – pick the $k + 1^{\text{th}}$ sample as the point furthest from any of the previous k points
- Gives **good results**
- Any **prefix** of the sequence is also regularly spaced!
    - Great for quick downsampling
- ***Problems:***
    - Tricky to implement
    - Slow



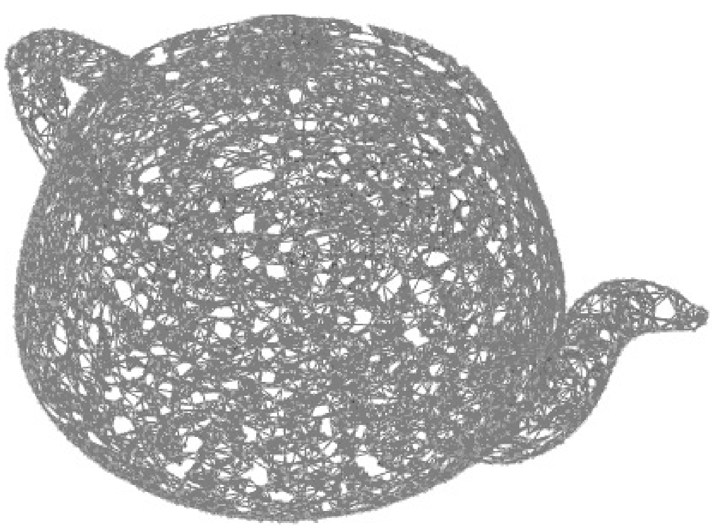
Furthest Point Sampling

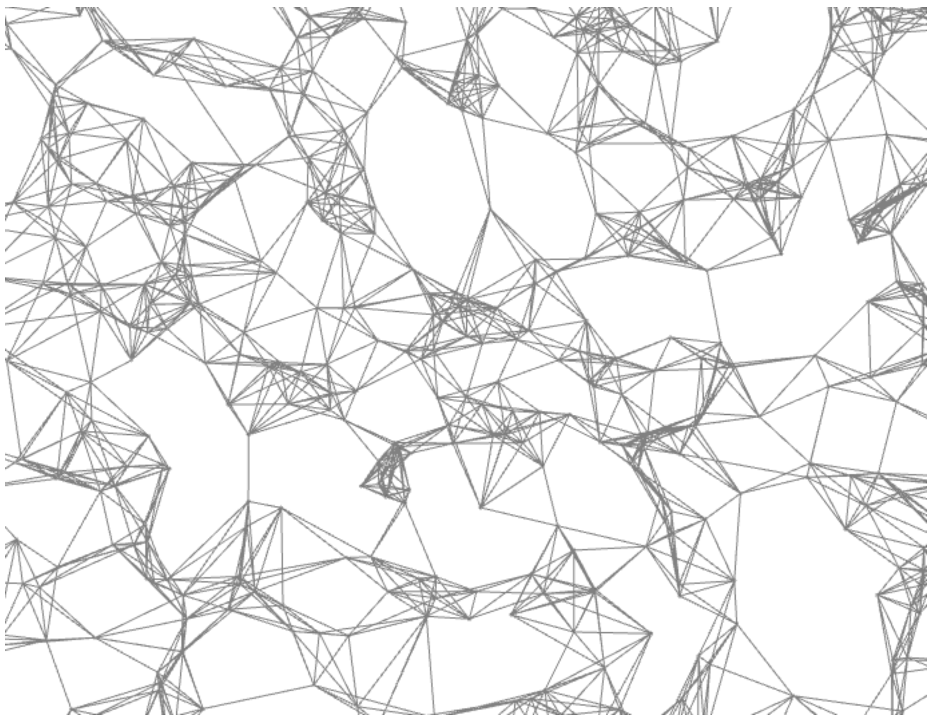
- ***Step 1:*** Oversample the shape by any fast method (e.g. for $N = 1000$ pick $N = 10000$ i.i.d. samples)



Furthest Point Sampling

- ***Step 2:*** Compute a k-Nearest Neighbors graph on the points (e.g. $k = 8$)





Furthest Point Sampling

- ***Step 3:***

  S = empty set

  for k = 1 to N

  compute distances from S to all graph vertices

  add the furthest vertex to S