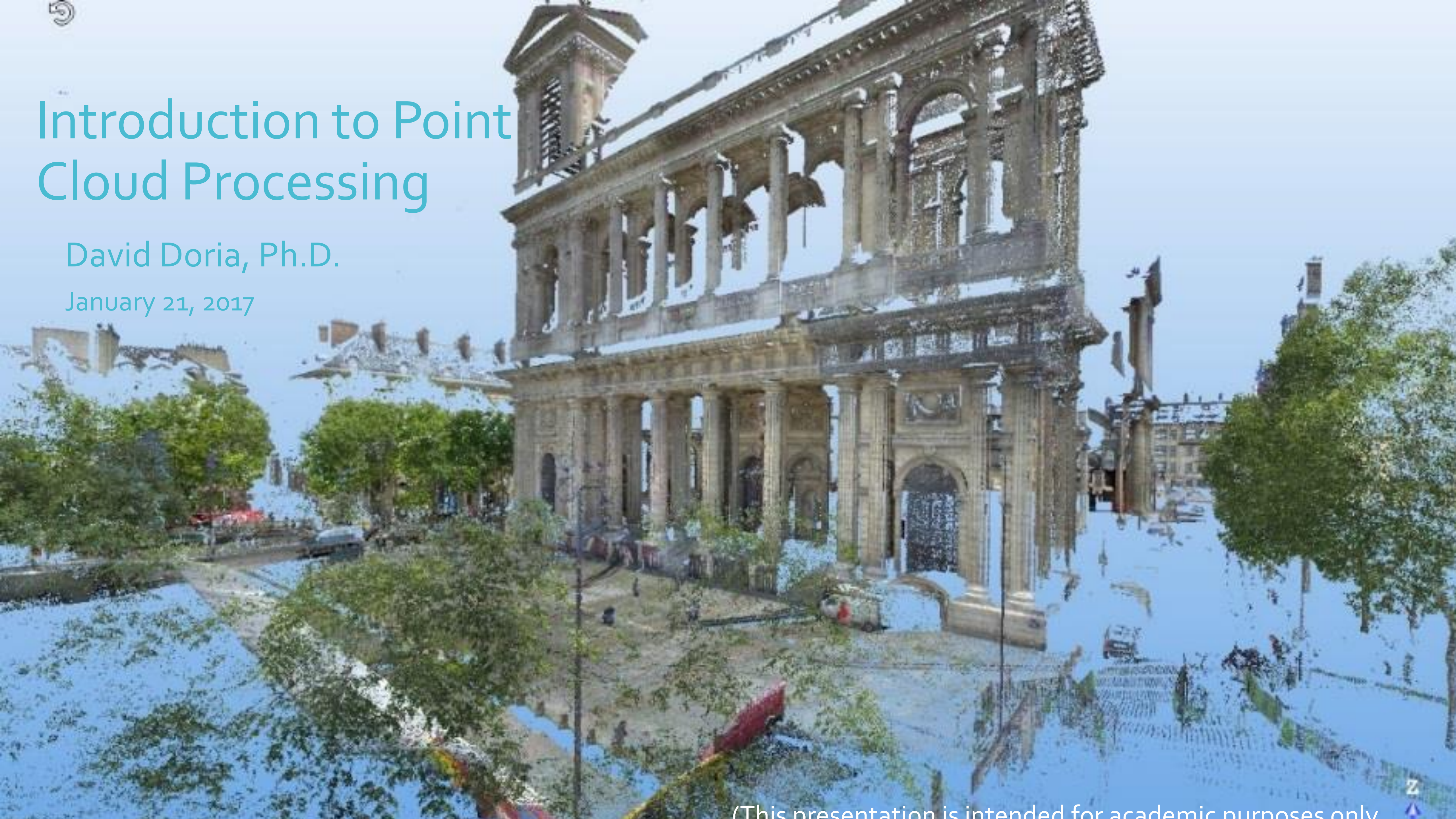
# Introduction to Point Cloud Processing

David Doria, Ph.D.

January 21, 2017



(This presentation is intended for academic purposes only

# What is Point Cloud Processing?

- What is a point?
- What is a "cloud" of points?
- What is "processing"?

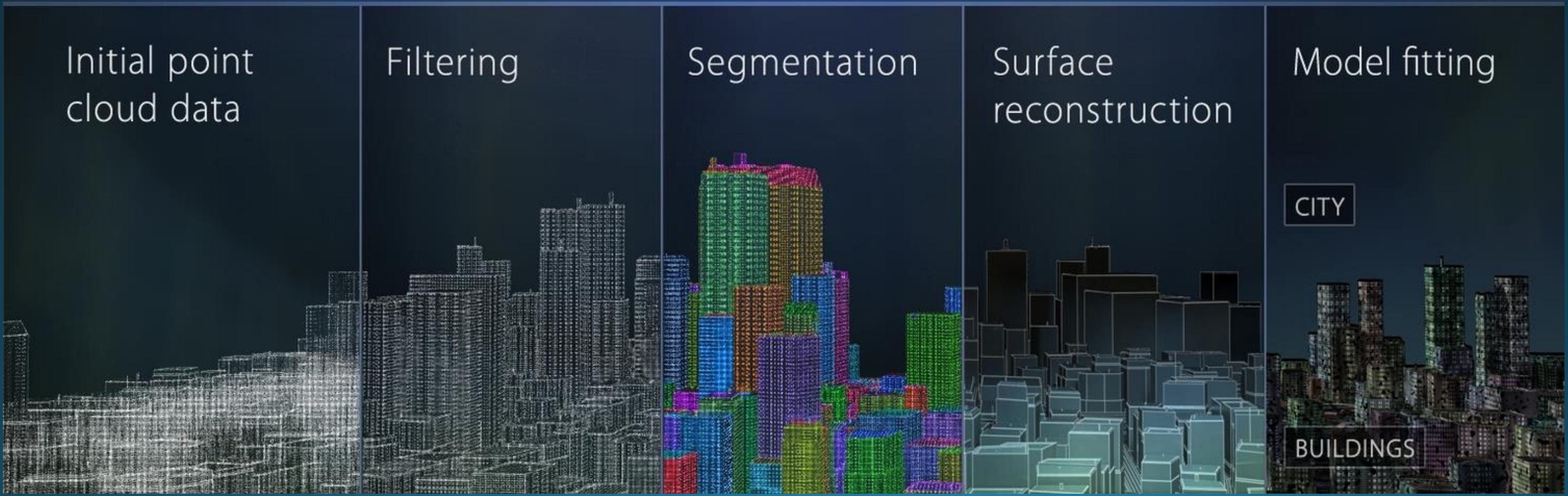
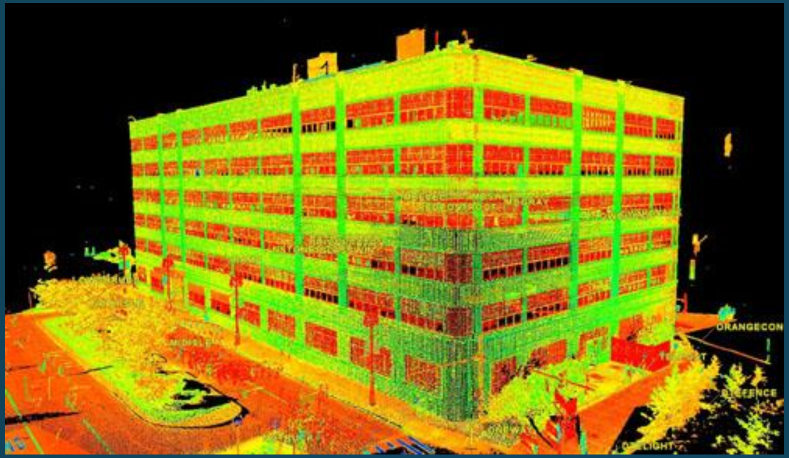
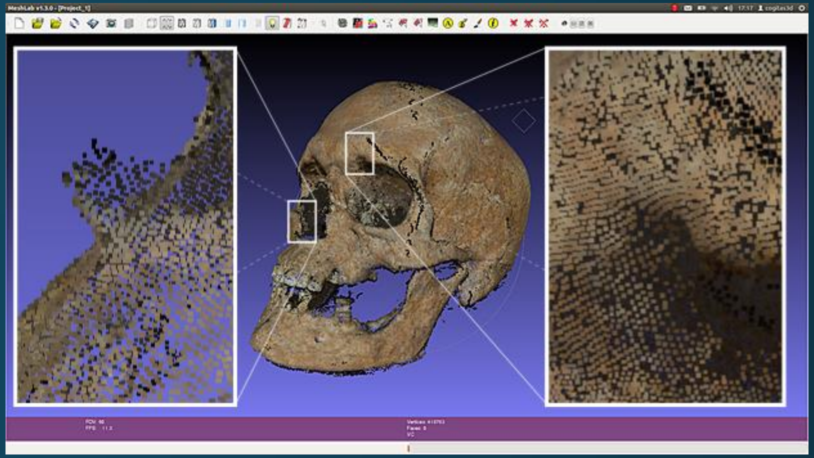


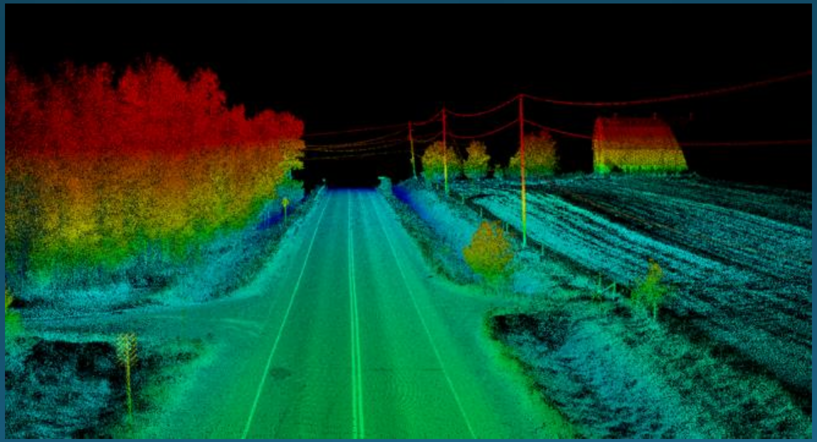
Image from pointclouds.org

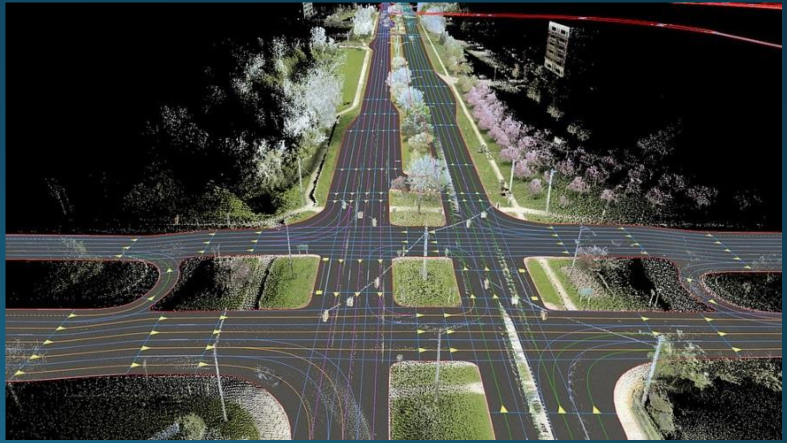
# What Are Point Clouds Used For?

- Architecture
- Robotics
- Manufacturing/construction
- Historical documentation
- Road modeling
- Many more…









# Manipulation, Viewer, and I/O Software

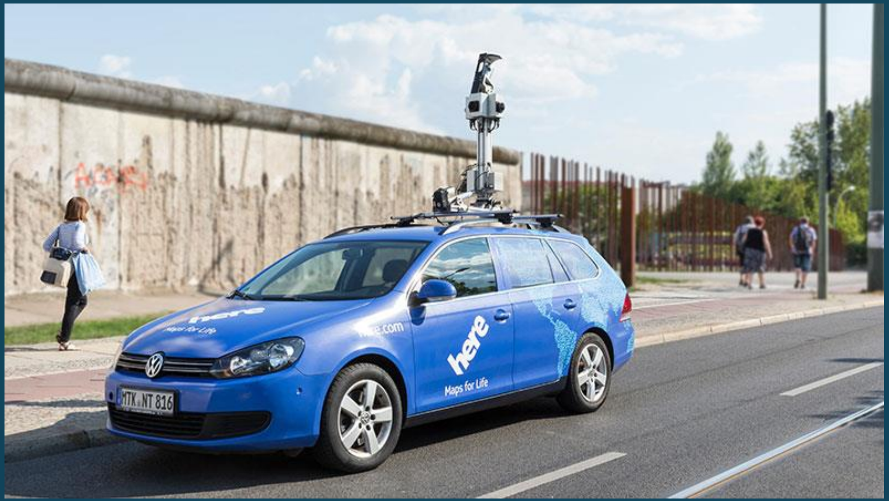
- Visualization Toolkit
- Point Cloud Library
- Liblas
- Paraview

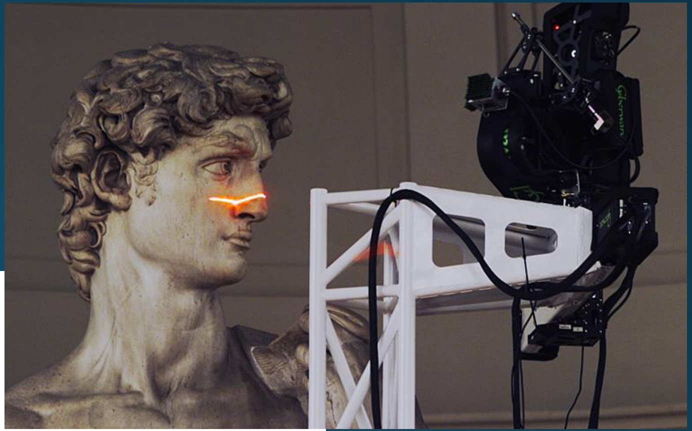
- CloudCompare
- MeshLab
- ArcGIS
- Many more…

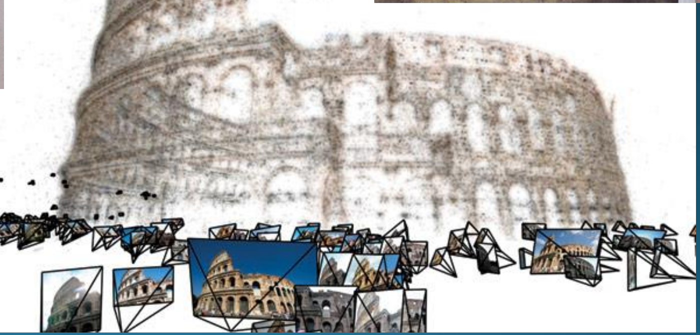
File formats: .las (standard), .pcd (PCL), .vtp (VTK), .csv (x, y, z), many more…

# Acquisition Methods

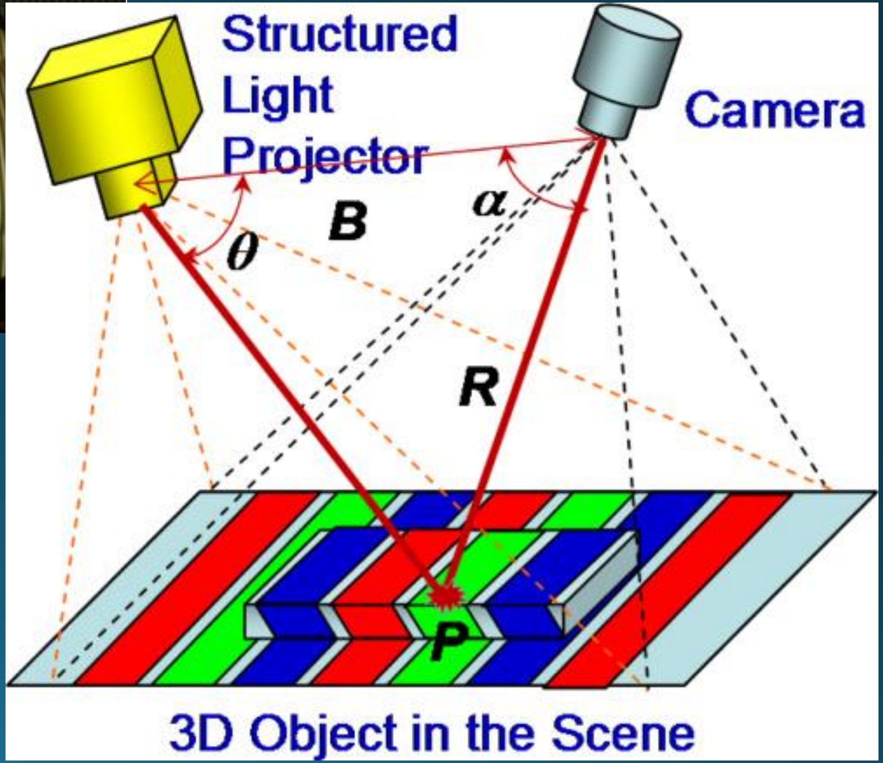
- Passive (Stereo vision)
- Active
  - structured light
  - LiDAR
  - physical touch
- Cost vs Speed vs Resolution vs Intrusiveness



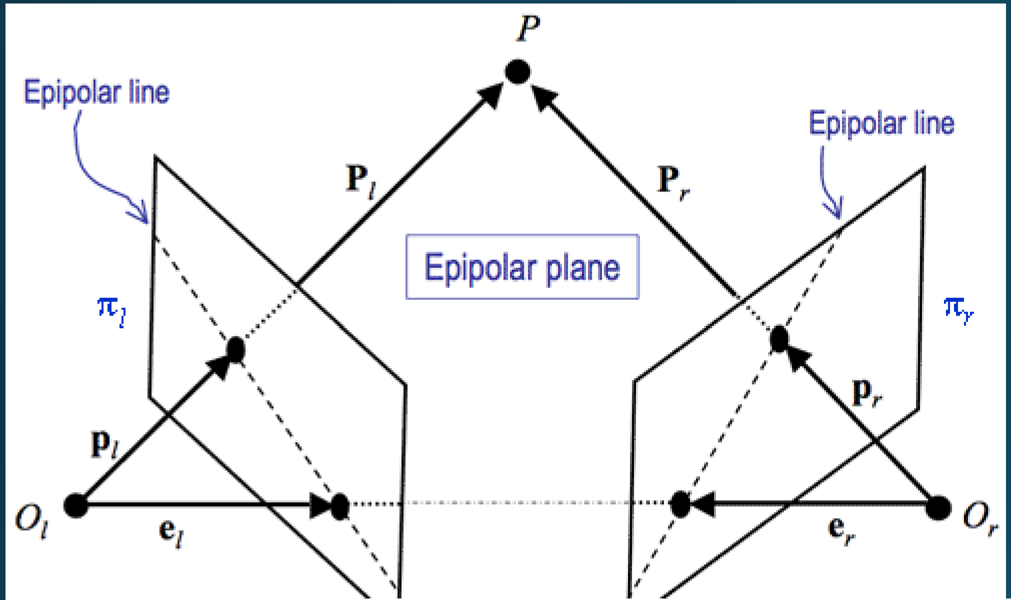


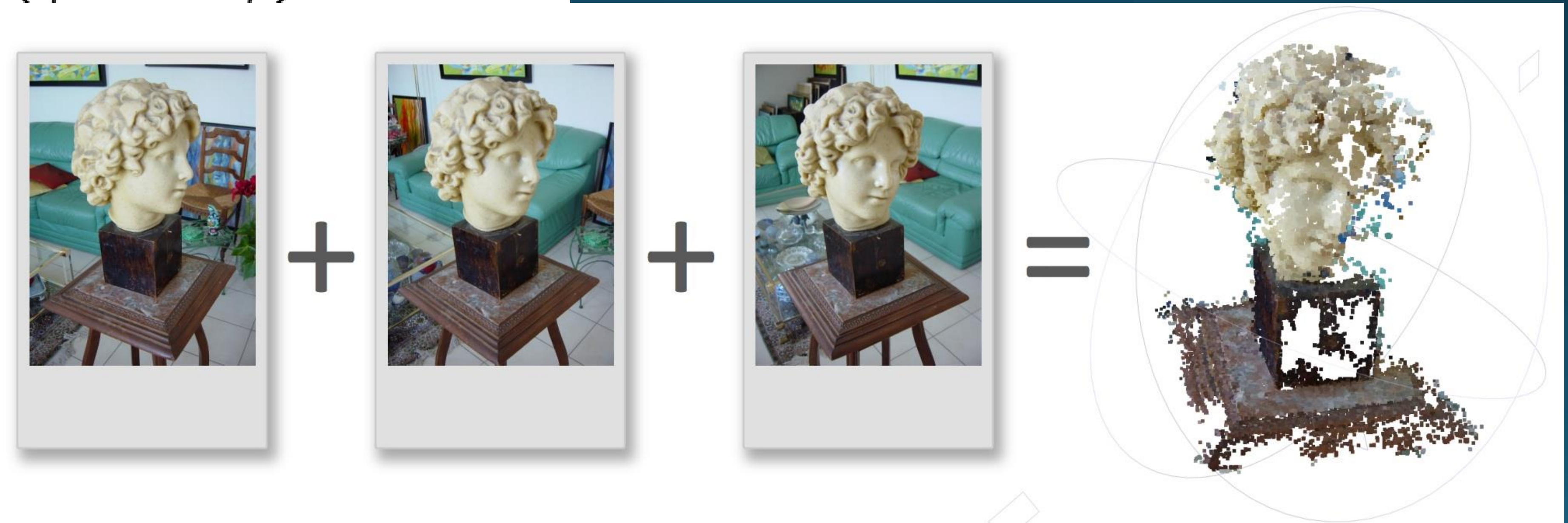


# Structured Light Scanning

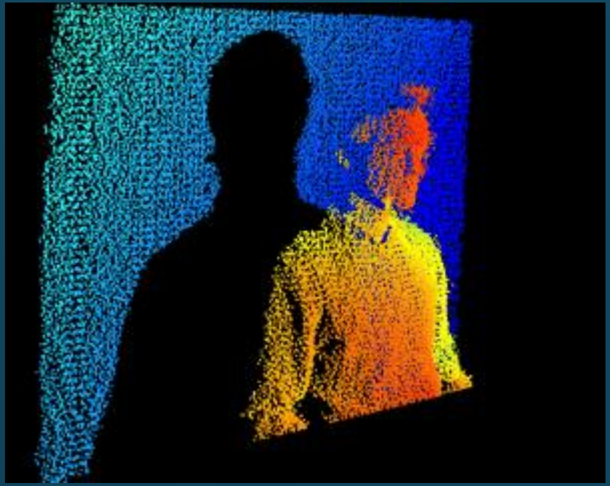


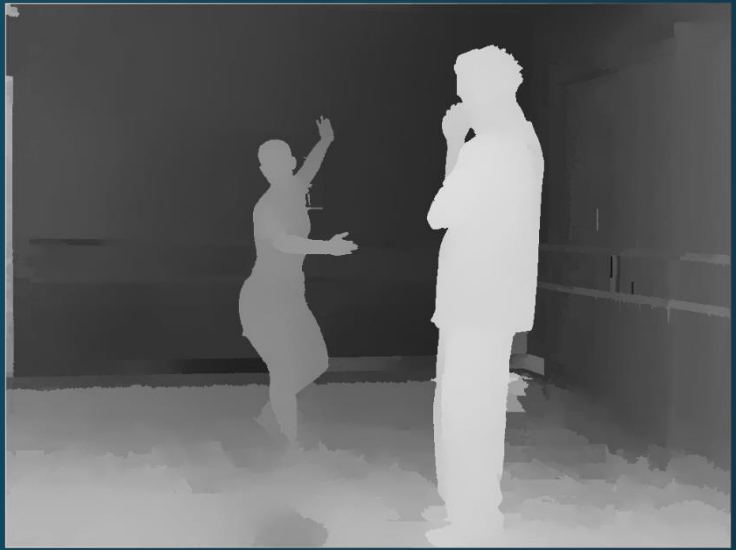
# Stereo Vision (aka 3D Reconstruction, aka Stereo Reconstruction)



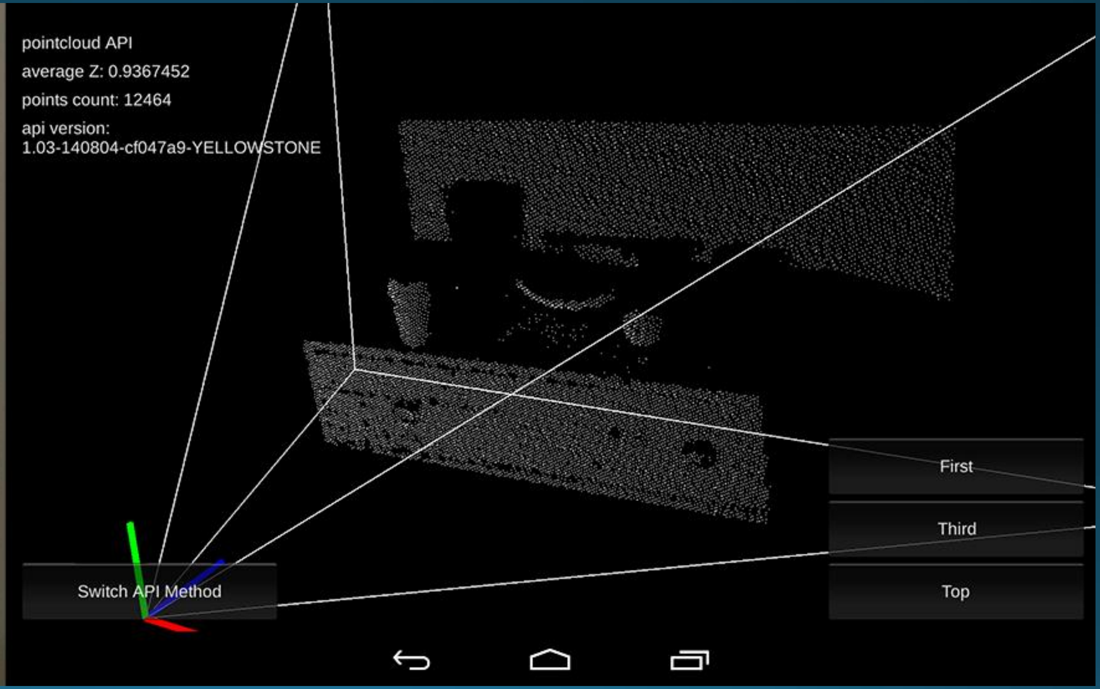


# Depth Images





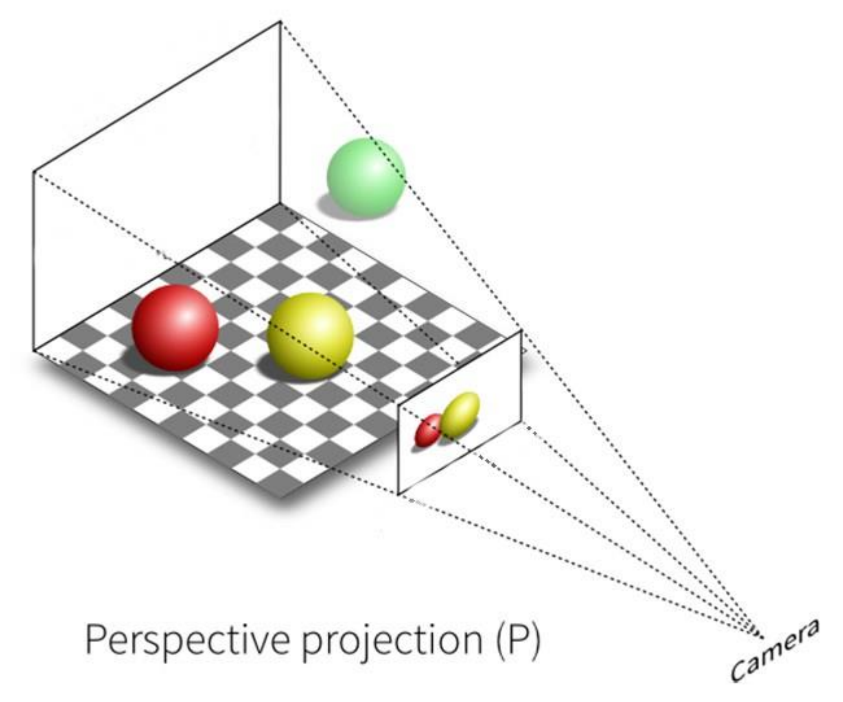
- “Structured” 3D data
- Single view point

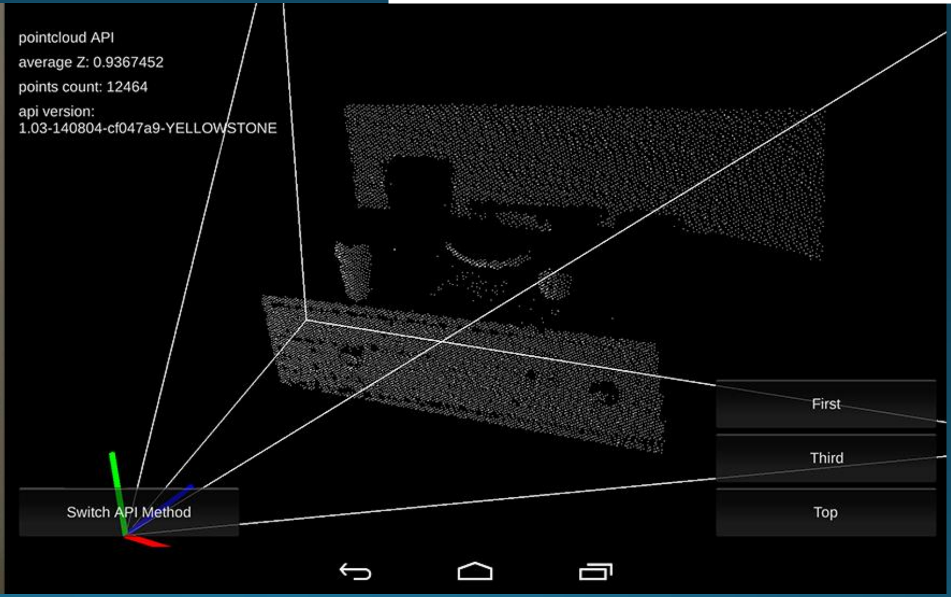


```
pointcloud API
average Z: 0.9367452
points count: 12464
api version:
1.03-140804-cf047a9-YELLOWSTONE
```

# Relationship Between Images and Points

- Relationship between images and points
- 3D->2D and 2D->3D projection/mapping

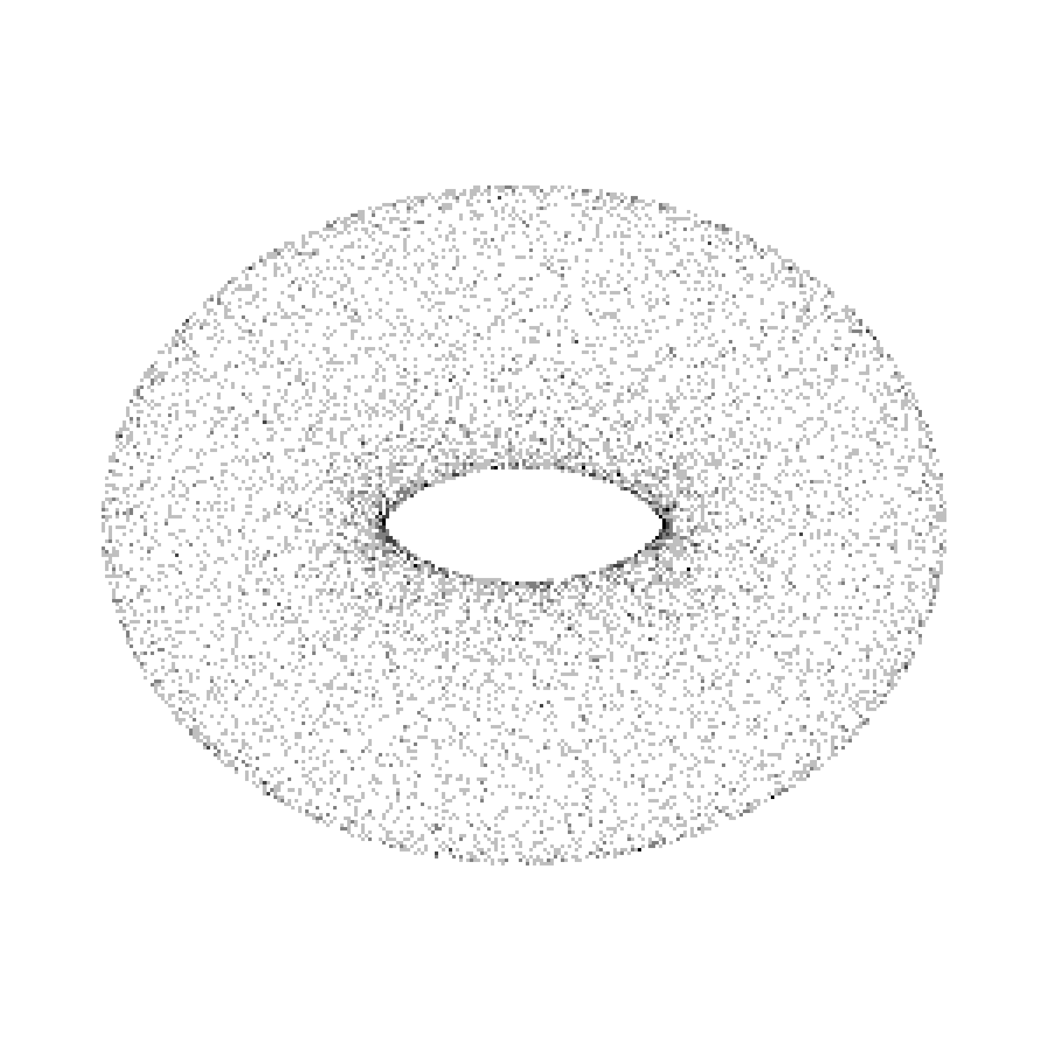



```
pointcloud API
average Z: 0.9367452
points count: 12464
api version:
1.03-140804-cf047a9-YELLOWSTONE
```

# Unorganized Point Clouds

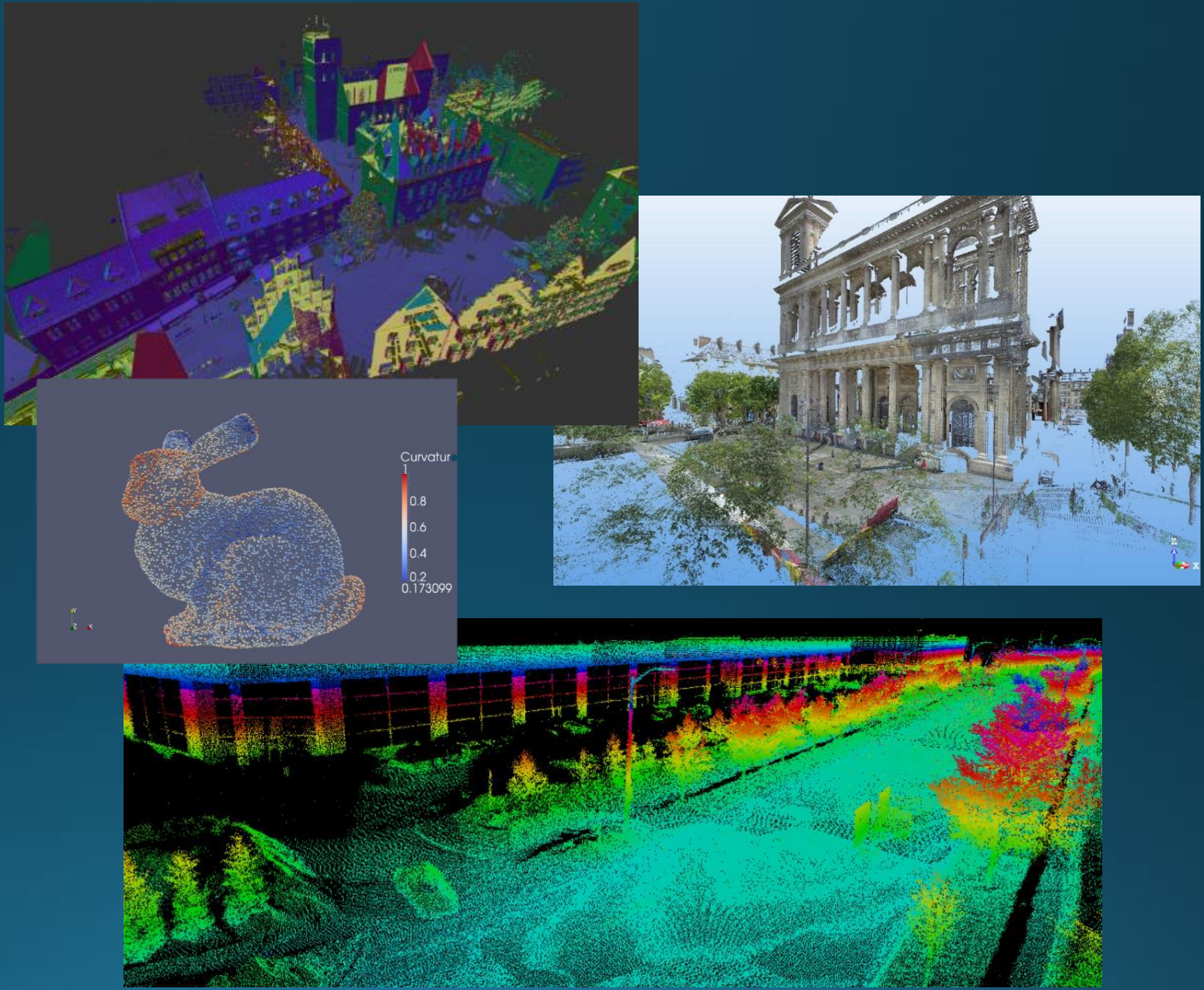
- No viewpoint

- Easy to acquire/store, hard to process

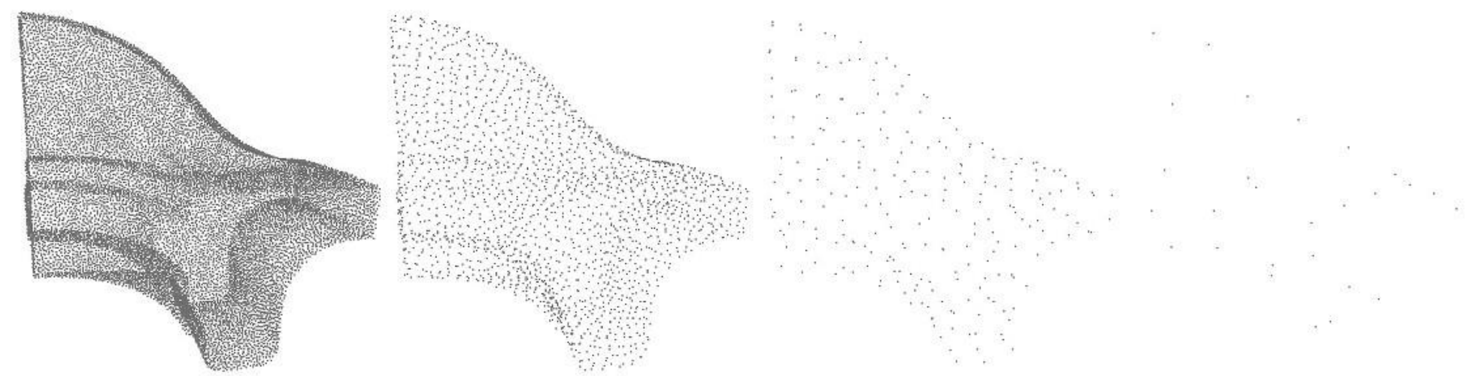


# Attributes on Points

- Color
- Intensity (reflectance)
- Anything else (curvature, cluster id, etc.)
- Very useful for visualization and thresholding



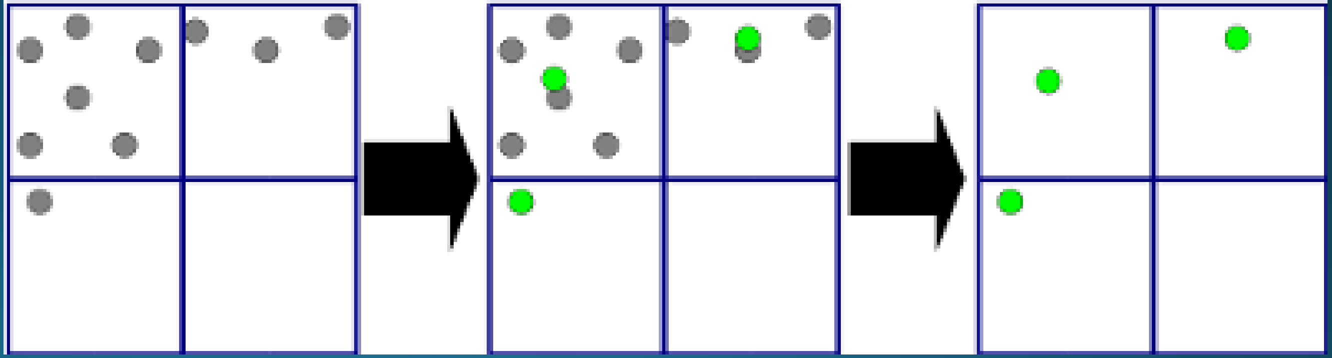
## Downsampling



- Remove unnecessary data

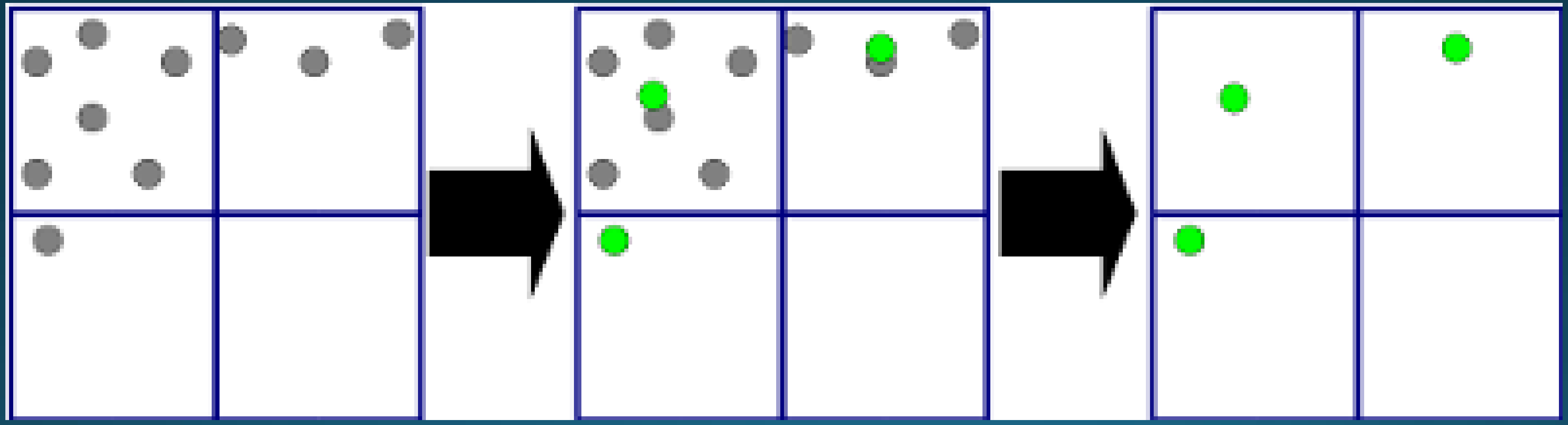
- Volumetric grids

- Honor “budgets” (processing time, storage space, etc.)
  - Usually represented as a # points

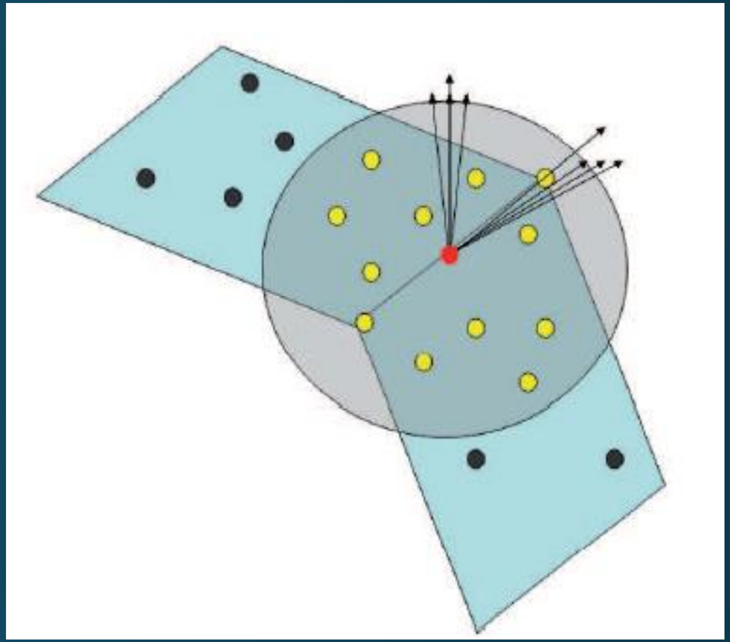


# Downsampling - simple

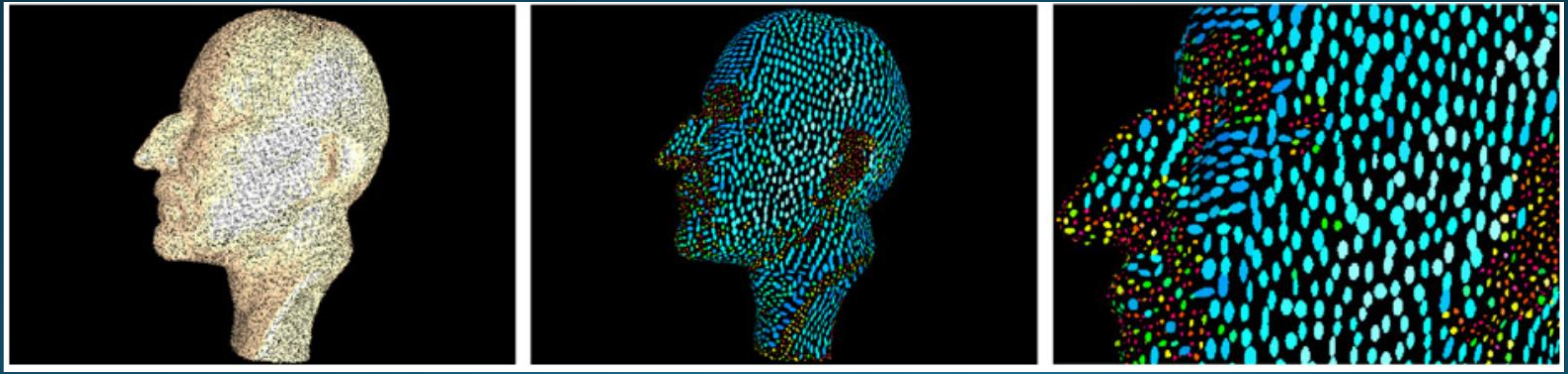
- Volumetric grids



# Downsampling

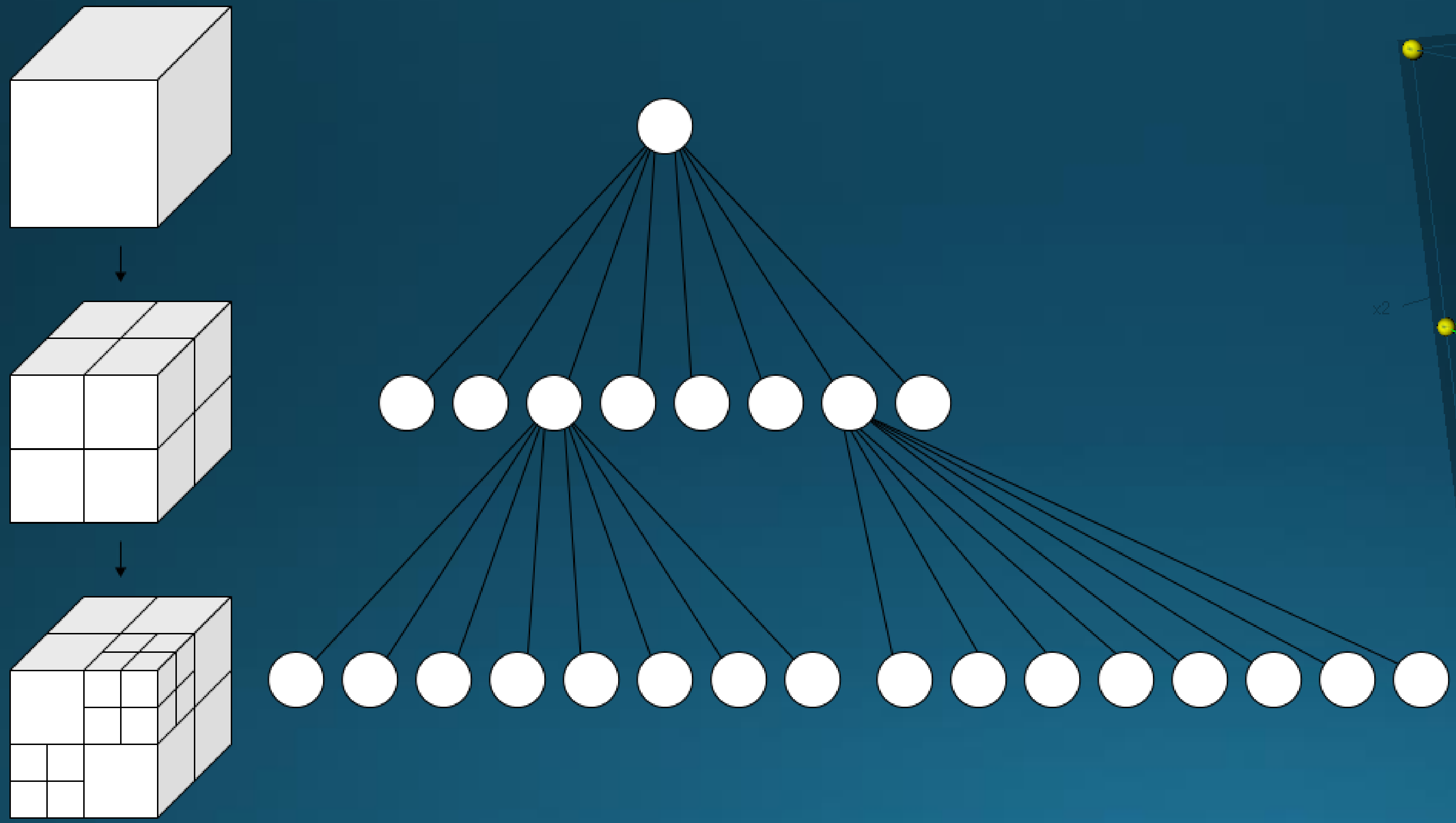


- Which points do we keep?
- Remove points only where they don’t add information
- “Curvature-aware”/”adaptive”

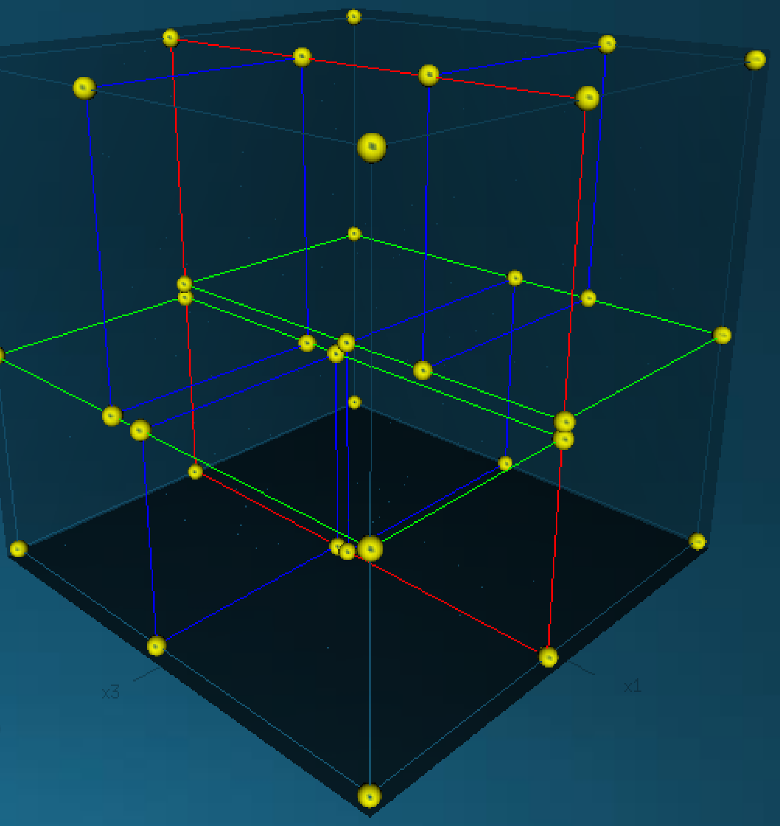


# Spatial Data Structures

- Efficient storage and computations
- Removes need to traverse over entire large data set



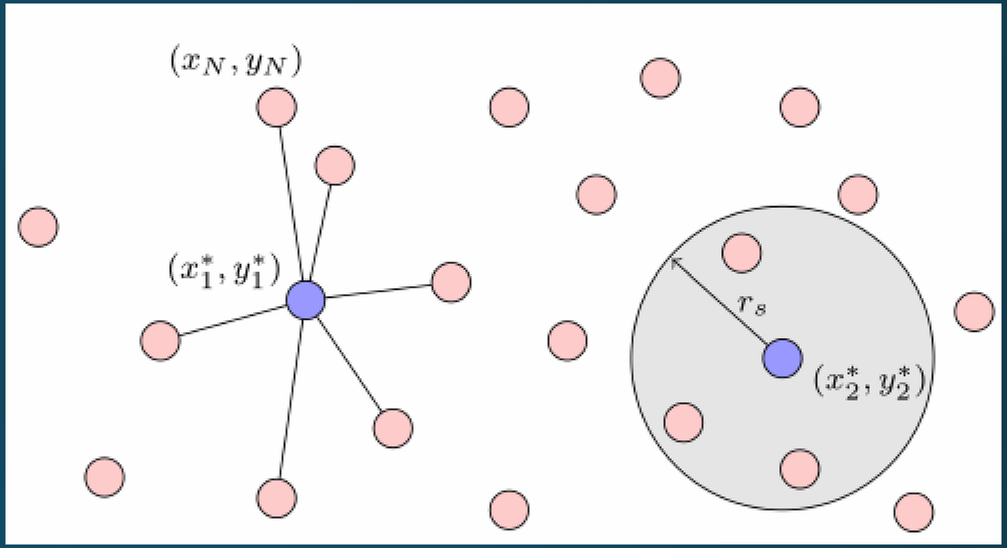
Octree

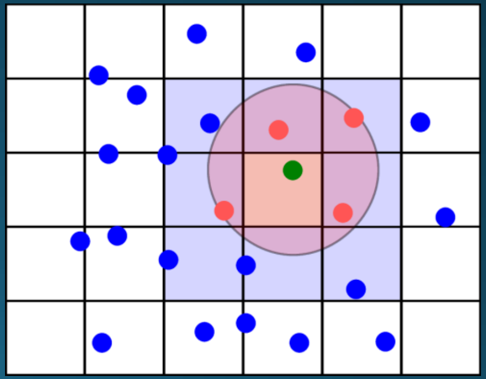


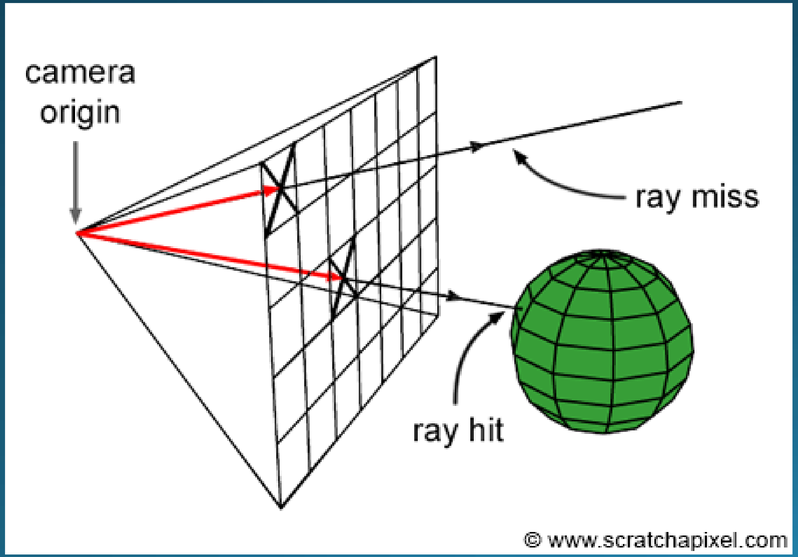
KD-tree

# Operations with Spatial Trees

- Operations:
  - Radius neighbor search
  - K-nearest neighbor search
  - Line-mesh intersection (for ray tracing)
  - Etc.



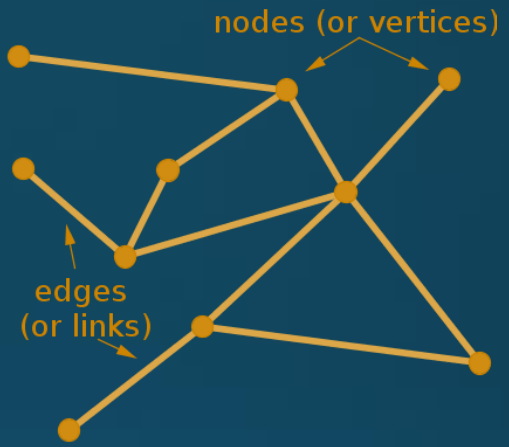


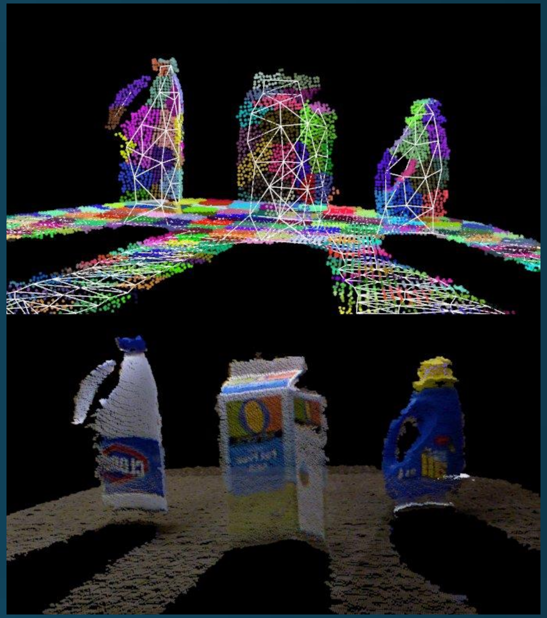


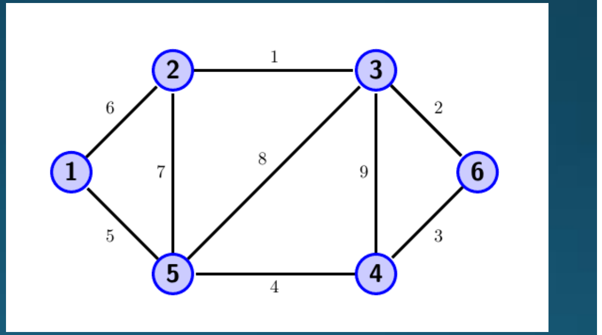
© www.scratchapixel.com

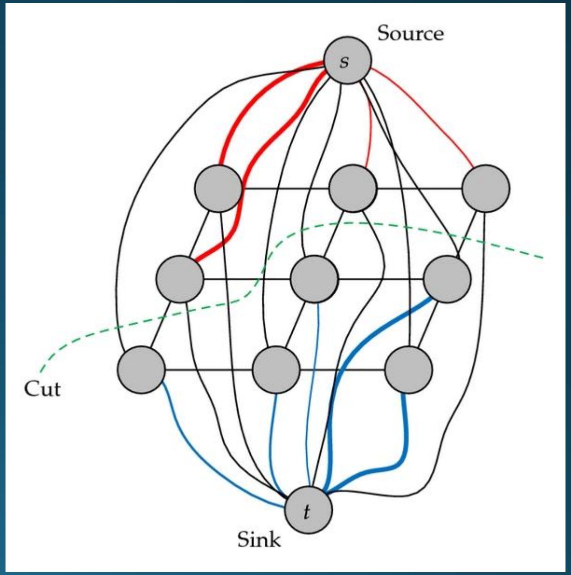
# Point Cloud Graphs

- Can use standard graph algorithms to do domain specific tasks simply by specifying connectivity and edge weights
  - Breadth first search (traversal)
  - Flood fill/paint bucket tool (connected components/region growing)
  - Segmentation (graph cuts)
  - Etc.





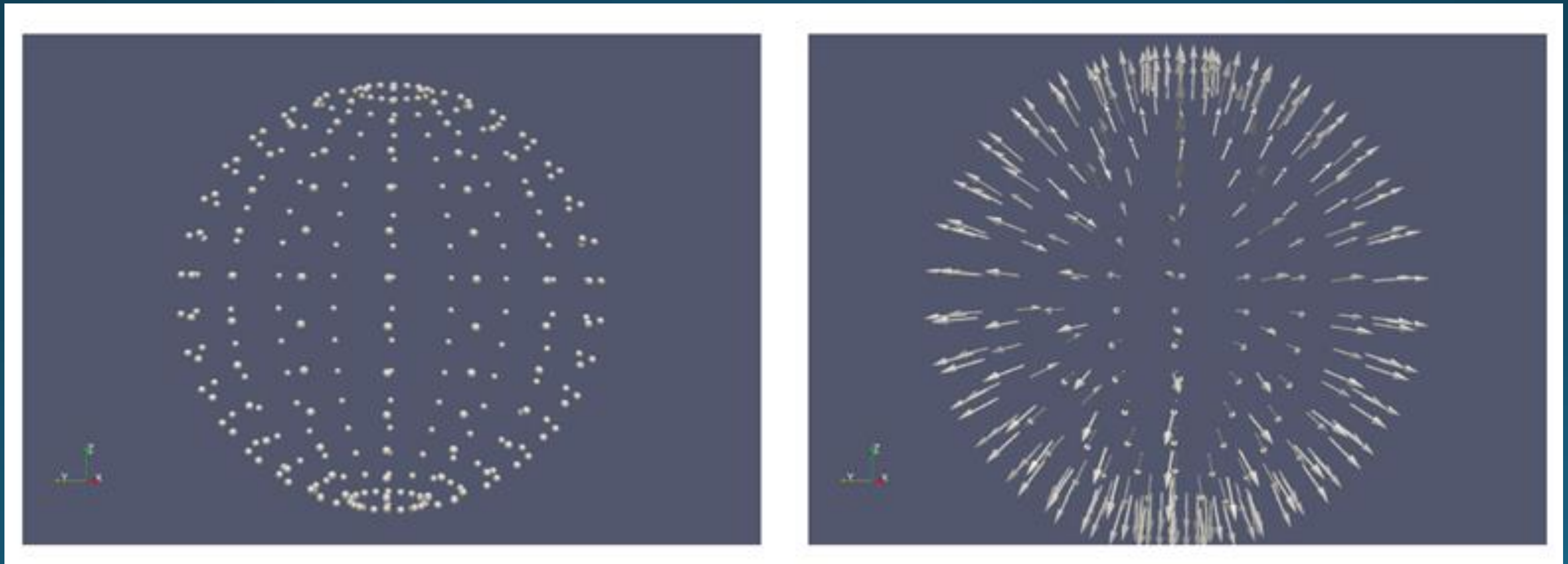




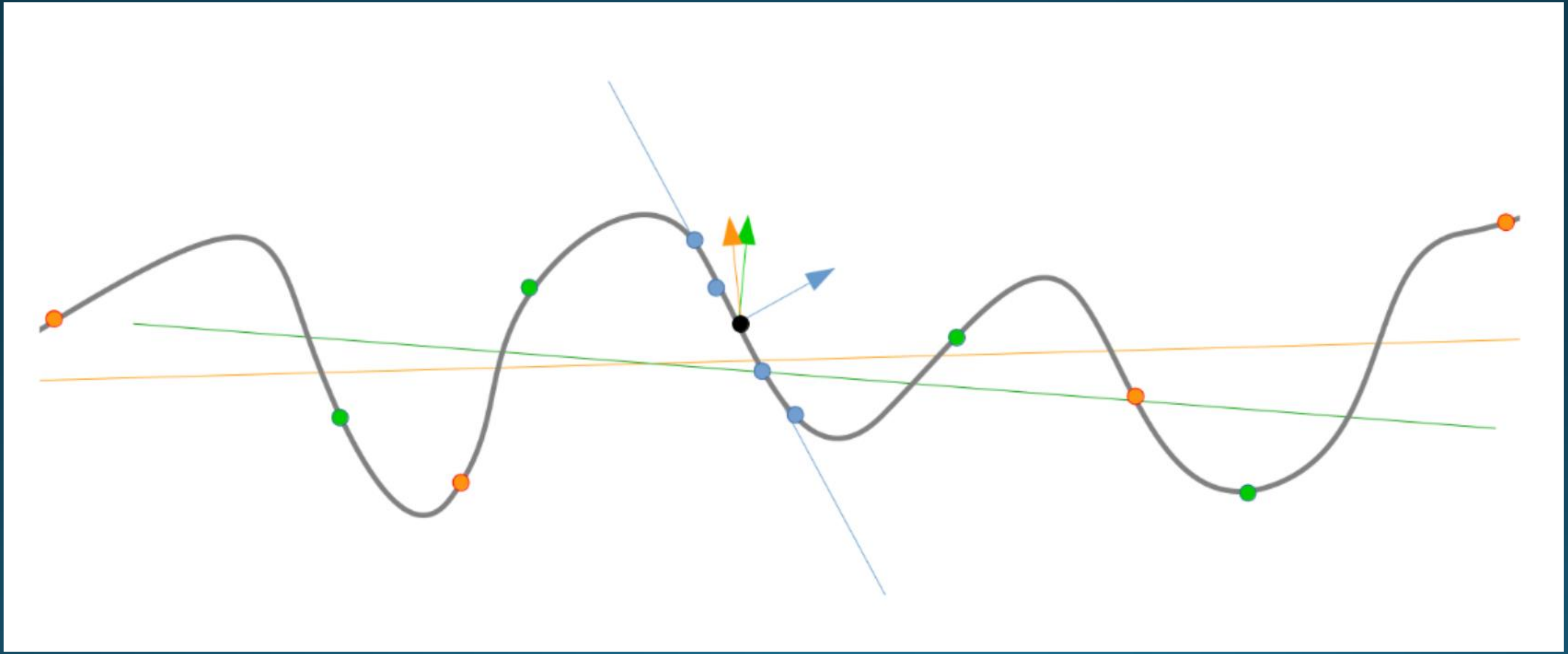
# Local Surface Properties

- Without reconstructing the surface (much, much harder), what would the surface look like locally?

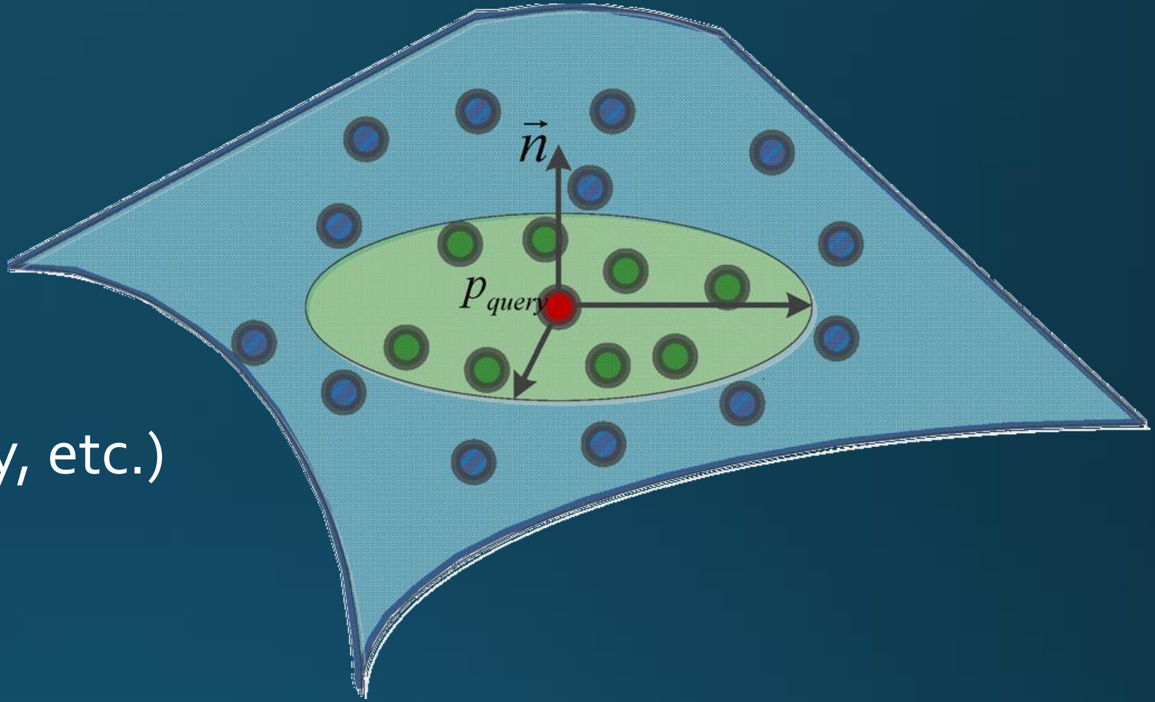
- Surface normals, curvature, etc.



# Scale is Critical to Analysis



# Principal Component Analysis (PCA)



- Surface normals
- Geometric descriptors (linearity, planarity, etc.)
- Eigen vectors of covariance matrix
  - How one coordinate changes with respect to the others

$$S_{jk} = \frac{1}{N-1}\sum_{i=1}^{N}(X_{ij} - \bar{X}_j)\,(X_{ik} - \bar{X}_k) \longleftrightarrow S = X^T X$$

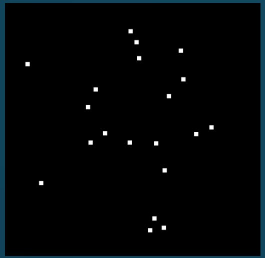
$$S = Q\Lambda Q^{-1}$$

Diagonal of $\Lambda$: Eigen values

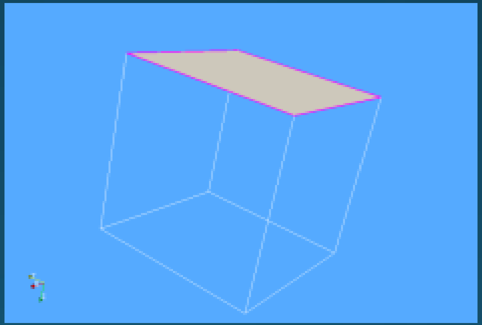
Columns of Q : Eigen vectors

# Surface/Mesh reconstruction

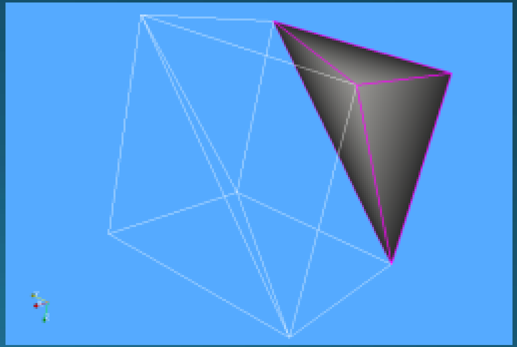
- Topology
  - Just points (0D)



  - Line segments (1D)



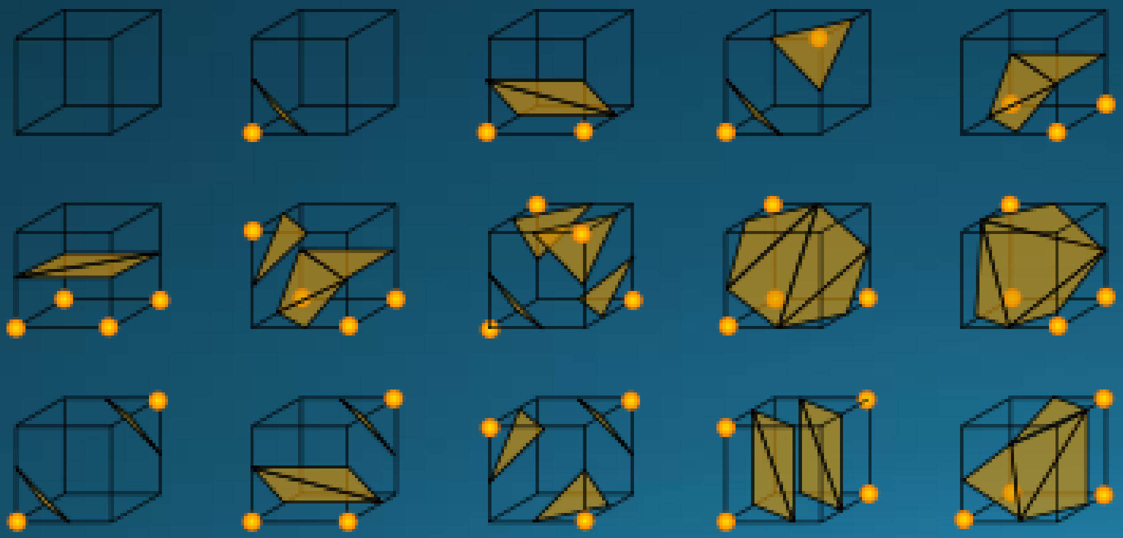
  - Polygons (usually triangles (2D)
  - Volumetric primitives (usually tetrahedra) (3D)

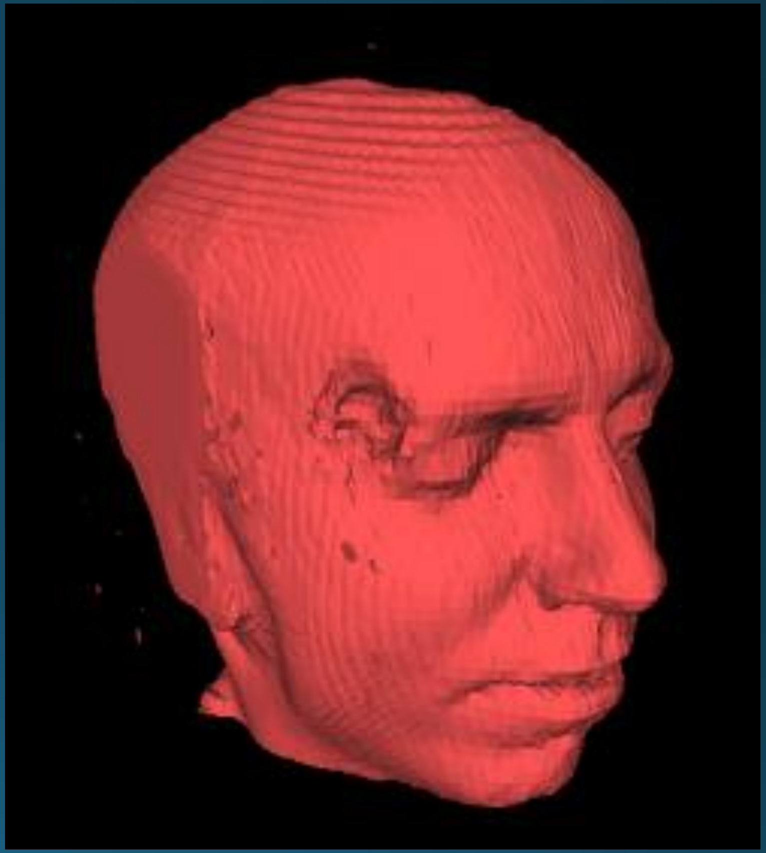


# Marching Cubes

- Construct a uniform grid and mark occupied voxels

- Determine which one of a pre determined set of configurations neighboring vertices are in



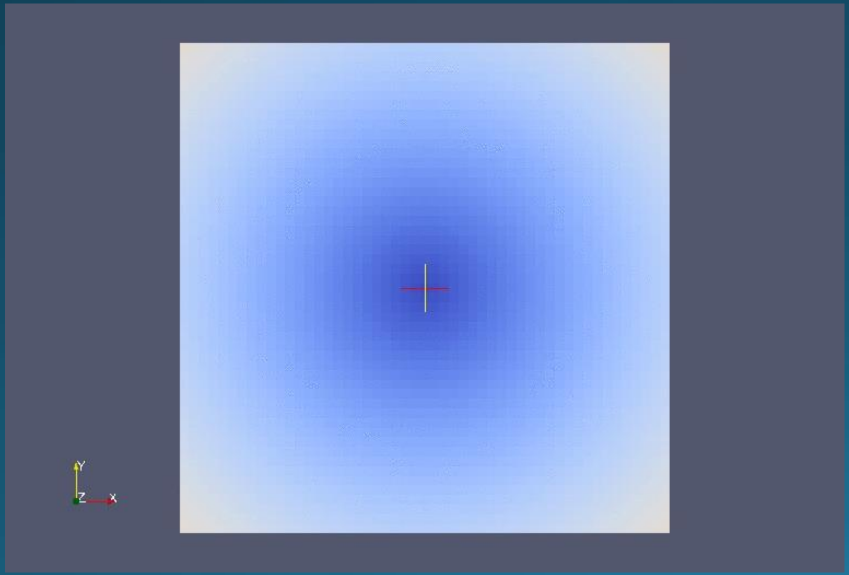


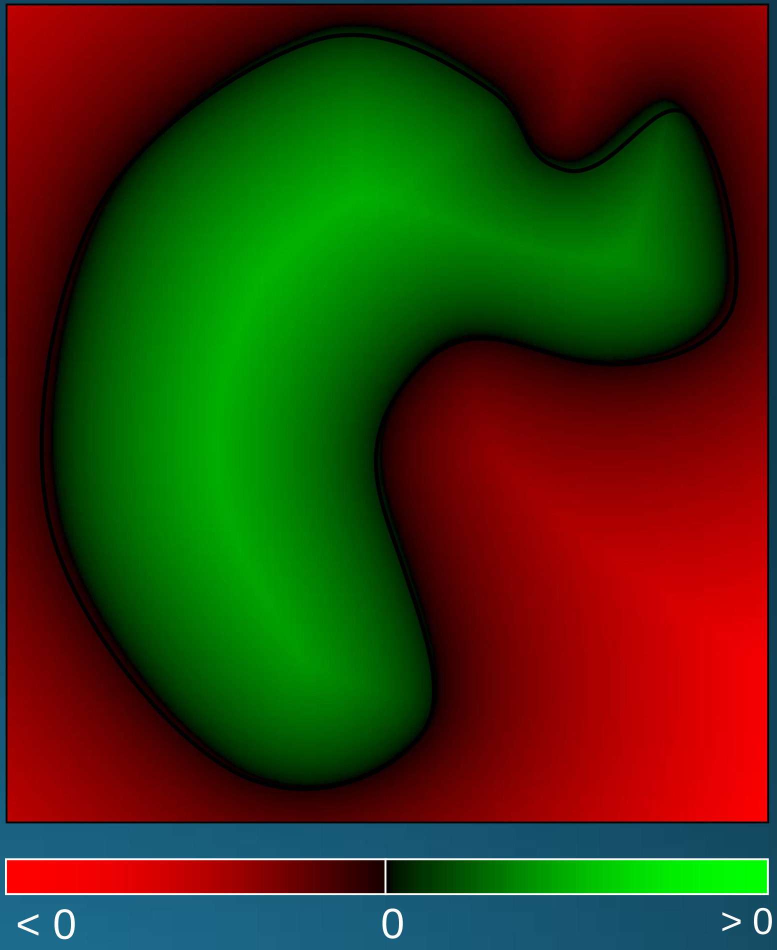
# Distance Fields

- Isocontours ("iso" from Greek word meaning 'equal')

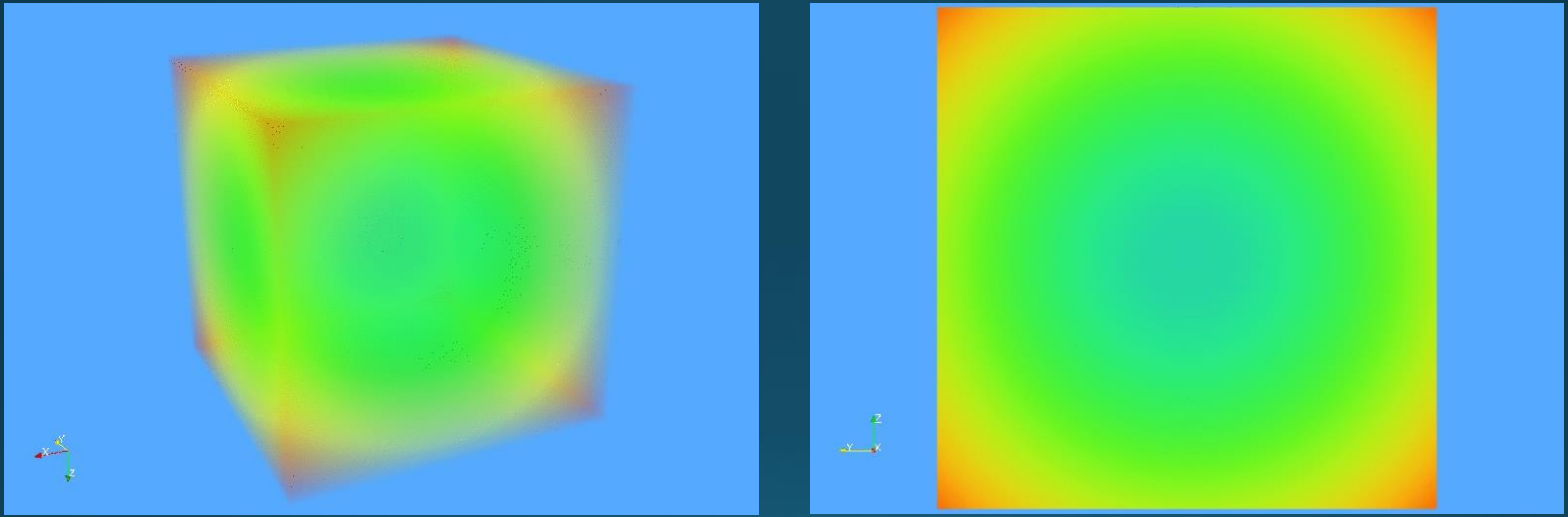
- Determine everywhere in a data set that the data equals a specified value

Shows isocontours of increasing values





# Isosurfaces



Two views of an isosurface and a slice through the volume+isosurface of increasing values over time.

# Poisson reconstruction

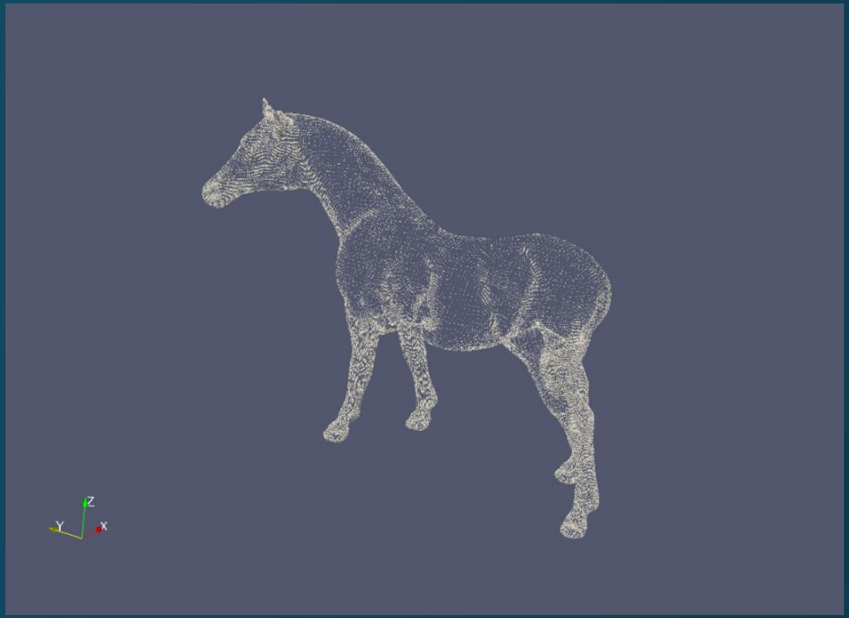
- Similar to an isosurface
- Name comes from solving a Poisson equation setup by making the estimated surface normal field, N, the gradient of the indicator function (zero level set) to find

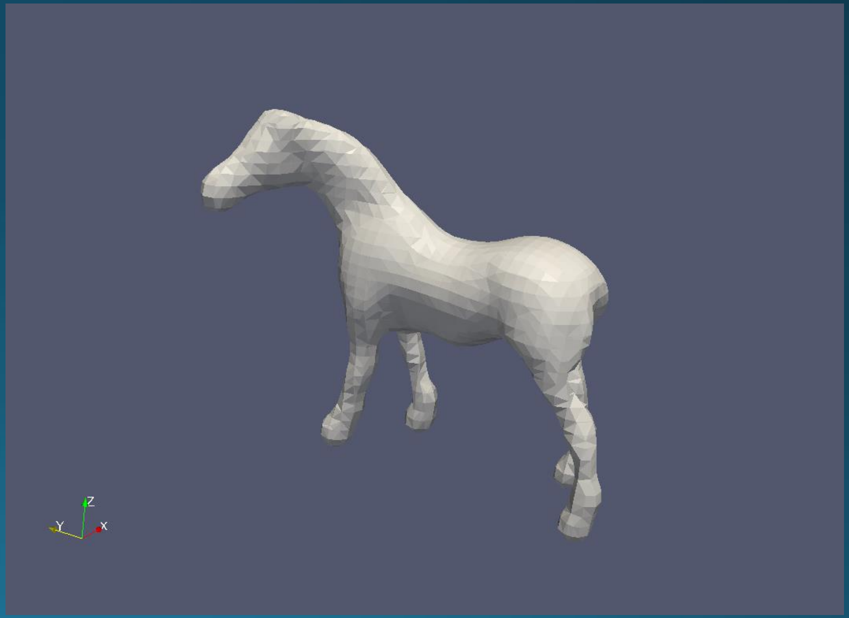
$$X' = \min_{X} ||\nabla X - N||$$

Equivalent to

$$\Delta X = \nabla N$$

Which has a linear solution.

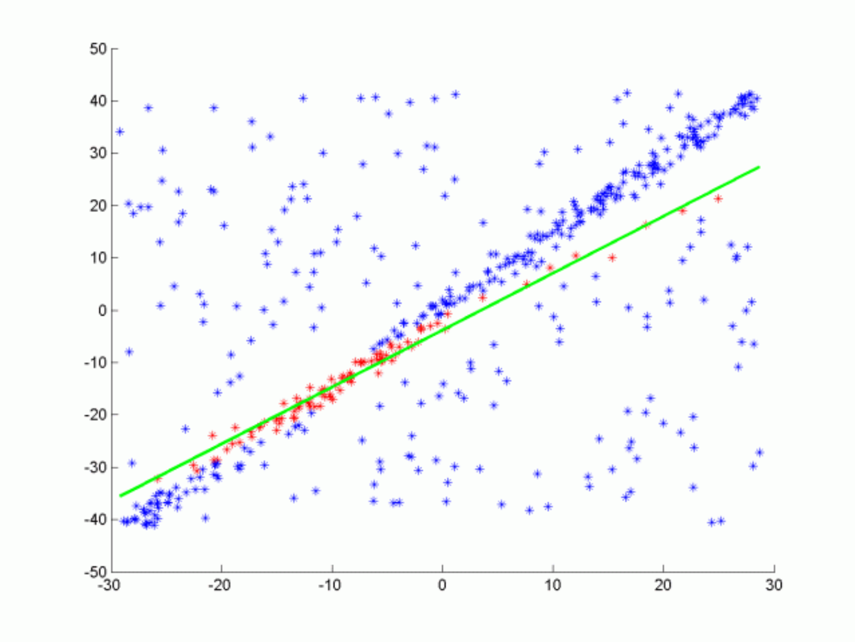


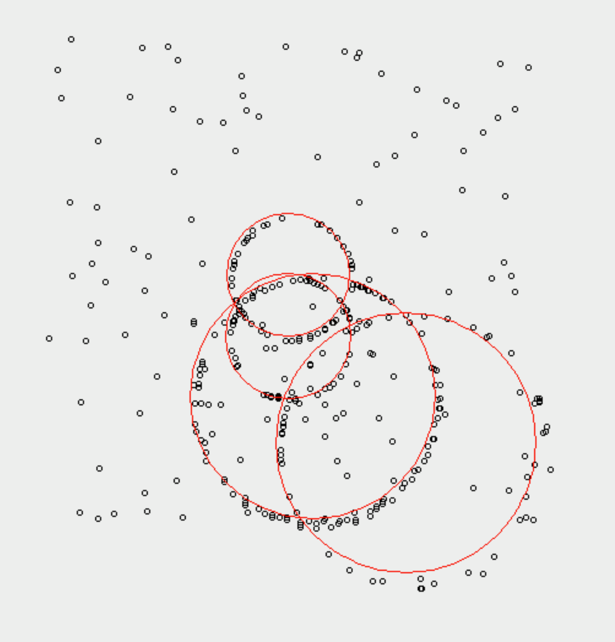


# Model Fitting

- Random Sample Consensus (RANSAC)
  - fitting models to subsets of data with outliers

- Lines, planes, circles, cylinders, etc.

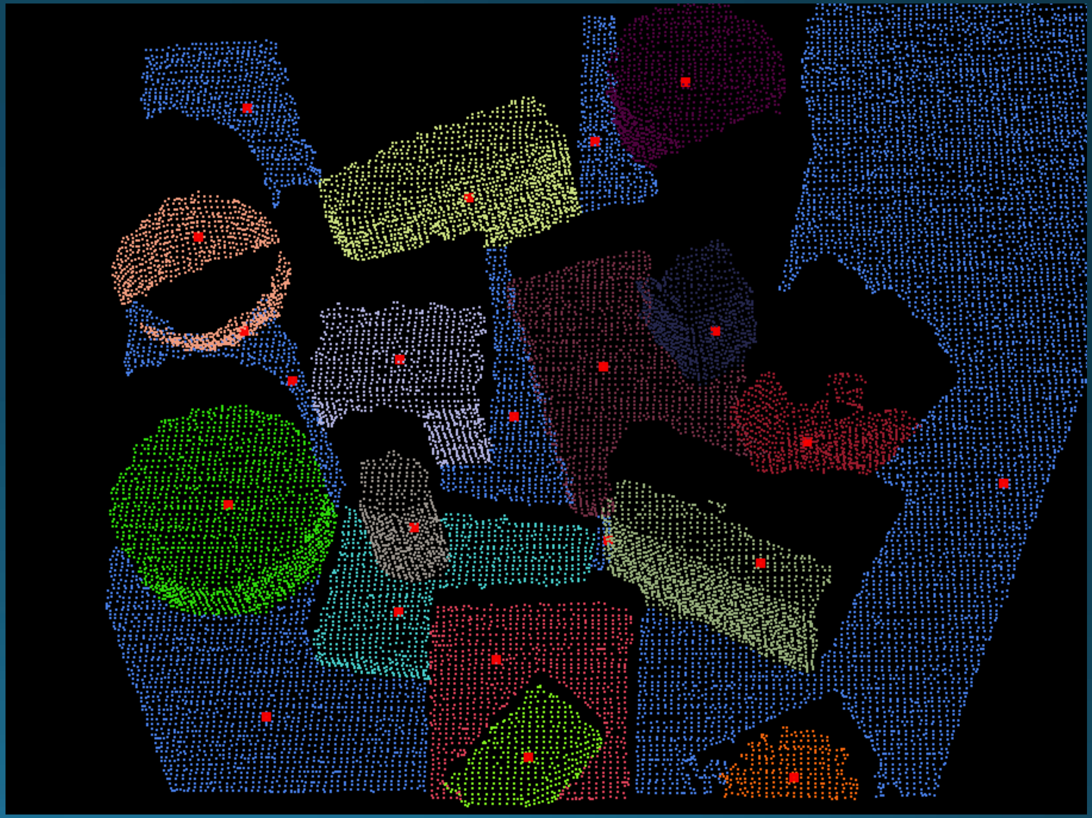


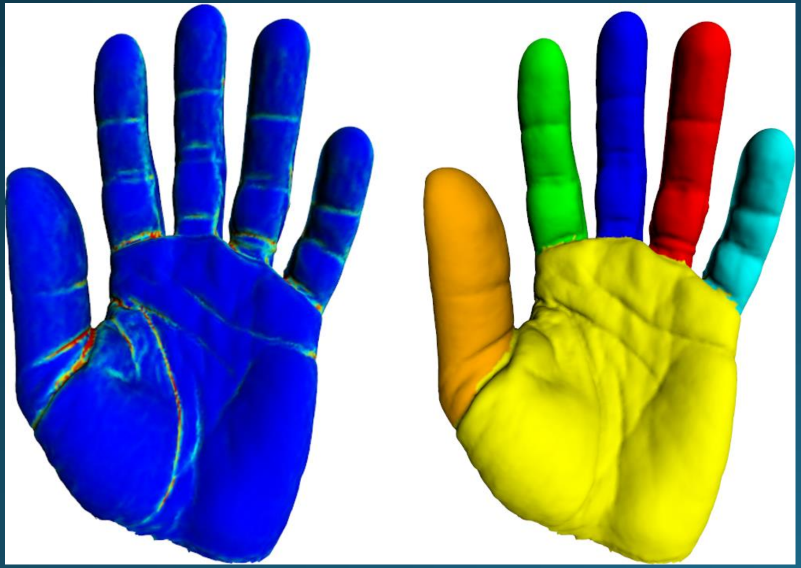


# Segmentation

- Find individual objects (foreground/background segmentation)

- Find parts of objects



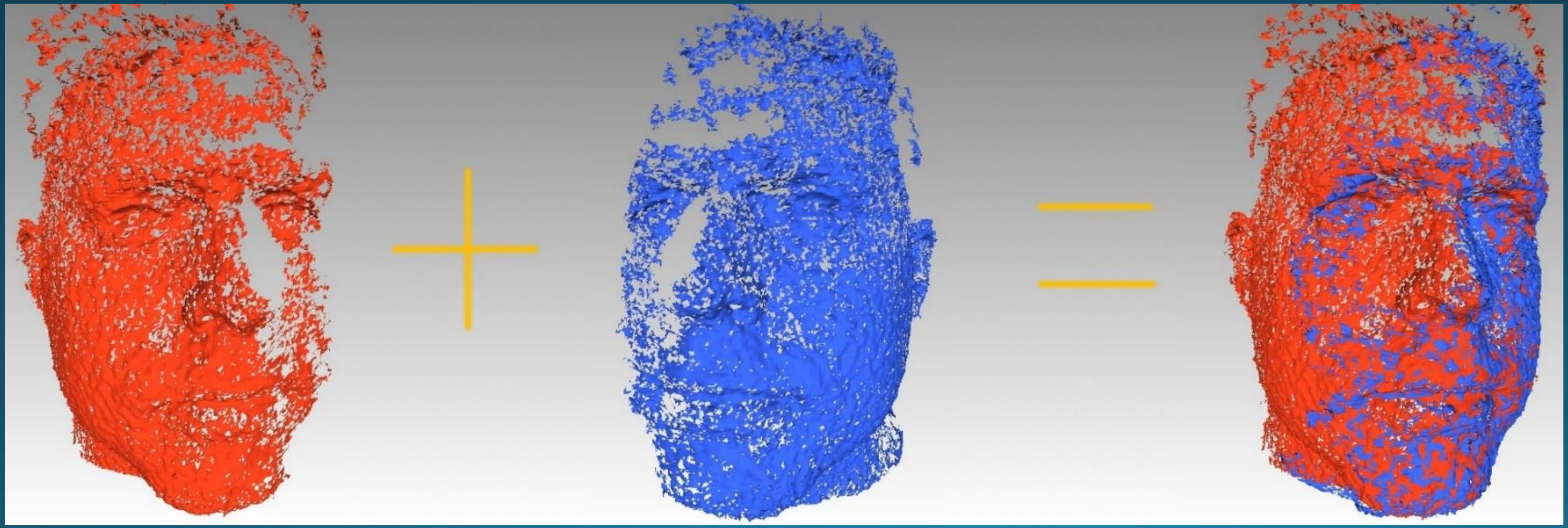


# Point Cloud Alignment

- Align = Register

- Take two (or more) data sets that were recorded in different coordinate frames and transform one (or more) so that they are all in the same frame

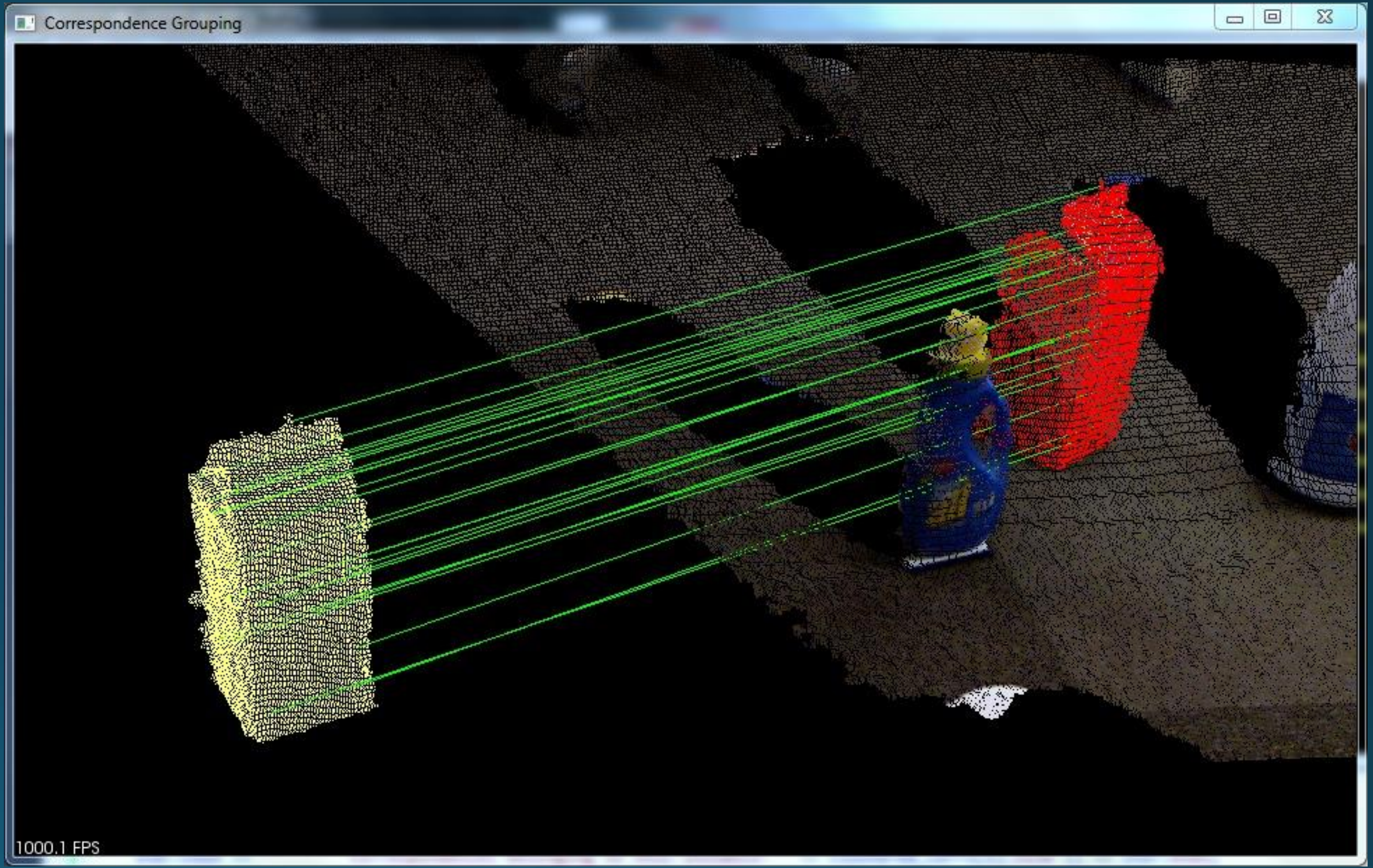
- Fill in missing data + reduce sparsity



# Landmark/Correspondence Transform

- Manually or automatically determine point-wise correspondences

- Compute a least squares transform between the correspondences

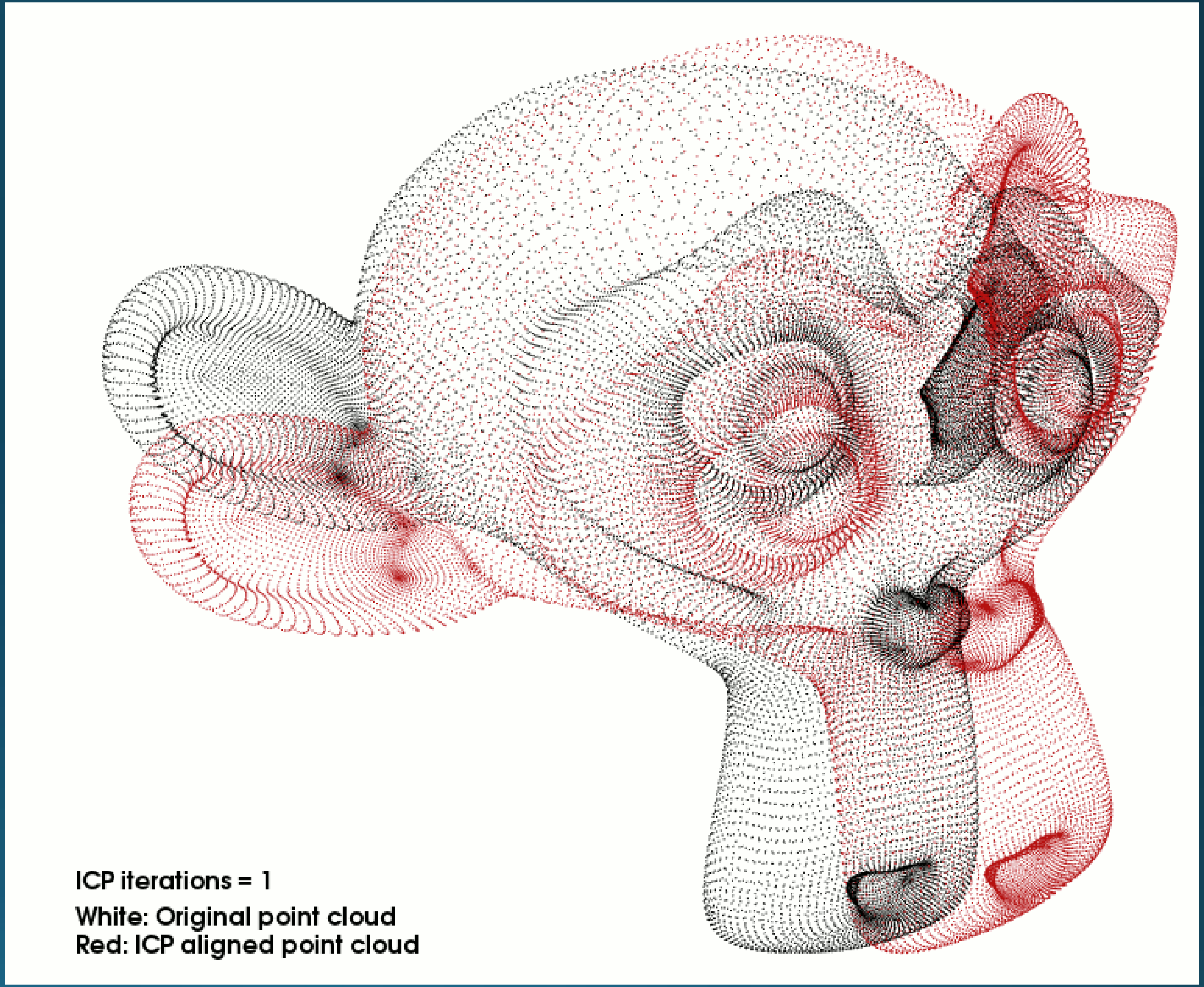


$$(R', t') = \min_{R,T} \frac{1}{N} \sum_{i=1}^{N} \|(Ra_i + t) - b_i\|^2$$

# Iterative Closest Points (ICP)

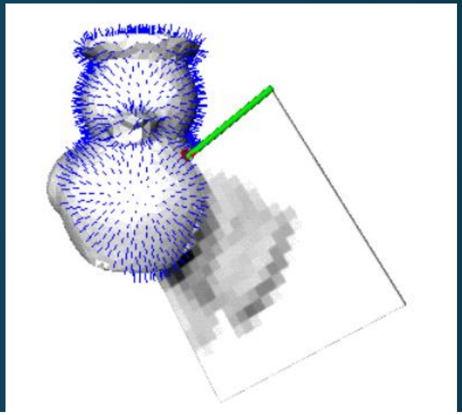
- Allows registration without explicit known correspondences
  - Assume nearest neighbors are correspondences
  - Perform landmark transform
  - Iterate

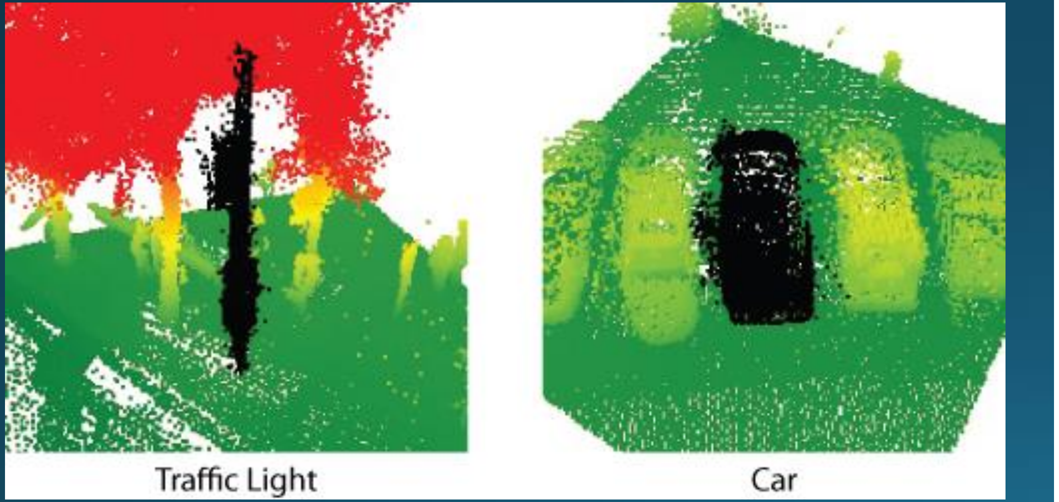
- Often used as a “fine tuning” step even when correspondences have been explicitly provided



# Object detection/recognition

- Classifier on feature vectors
- Many feature vectors proposed
- Classic example: spin images





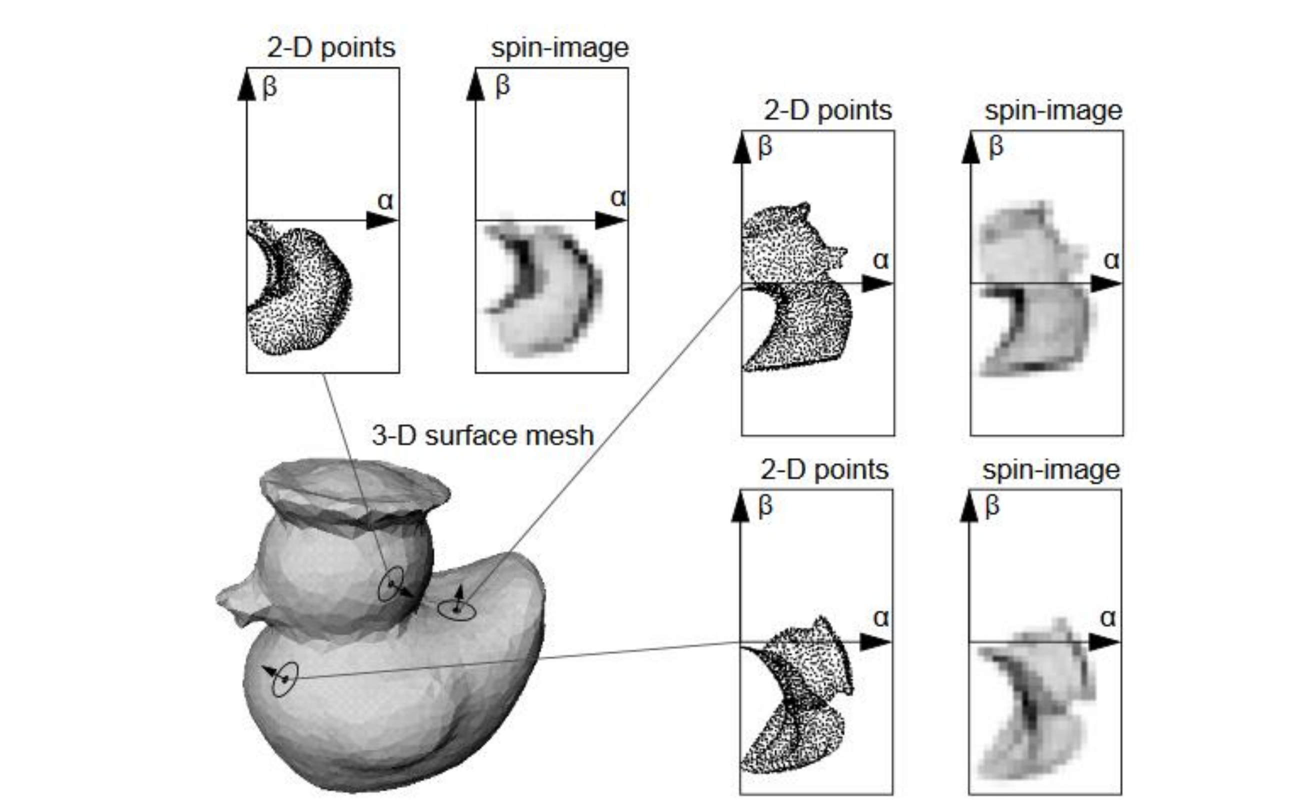
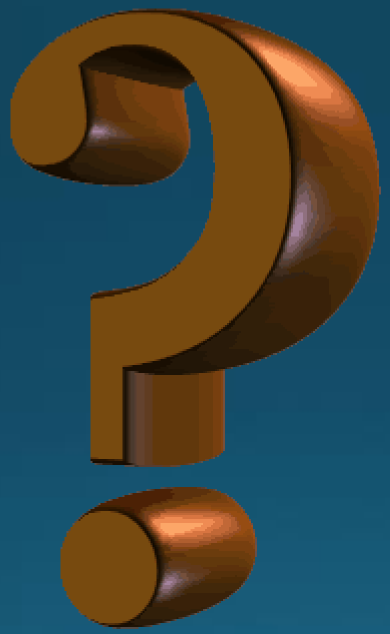


**Figure 2: Spin-images of large support for three oriented points on the surface of a rubber duck model.**

# Questions?

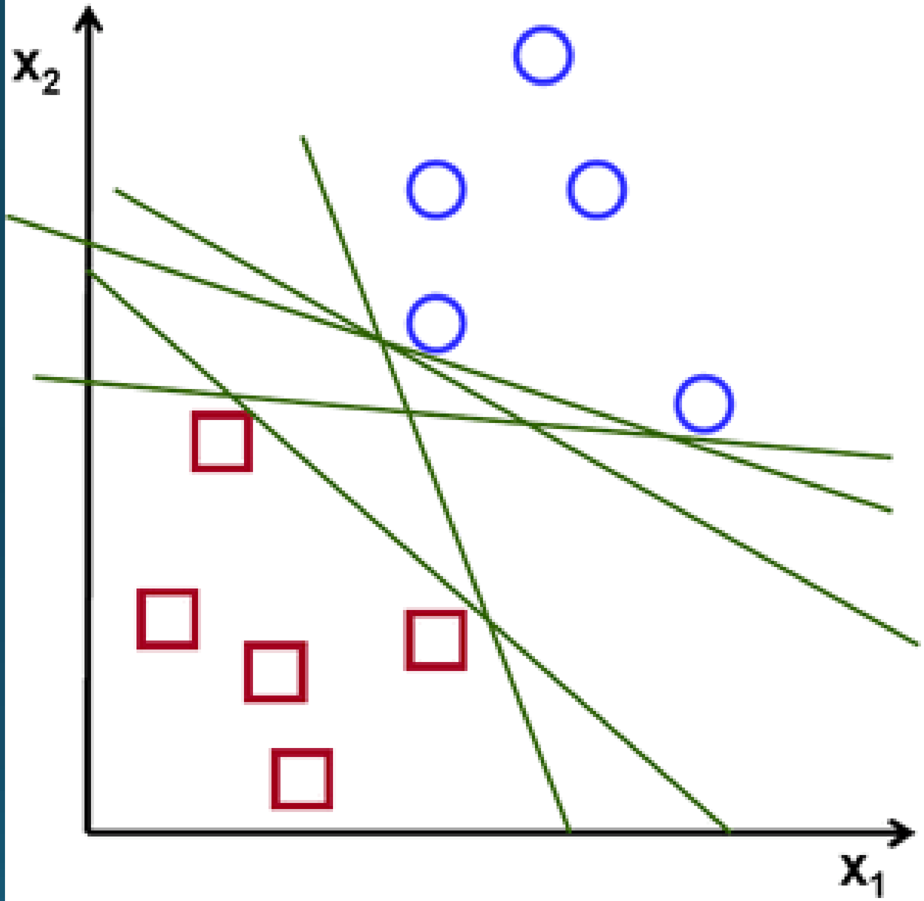


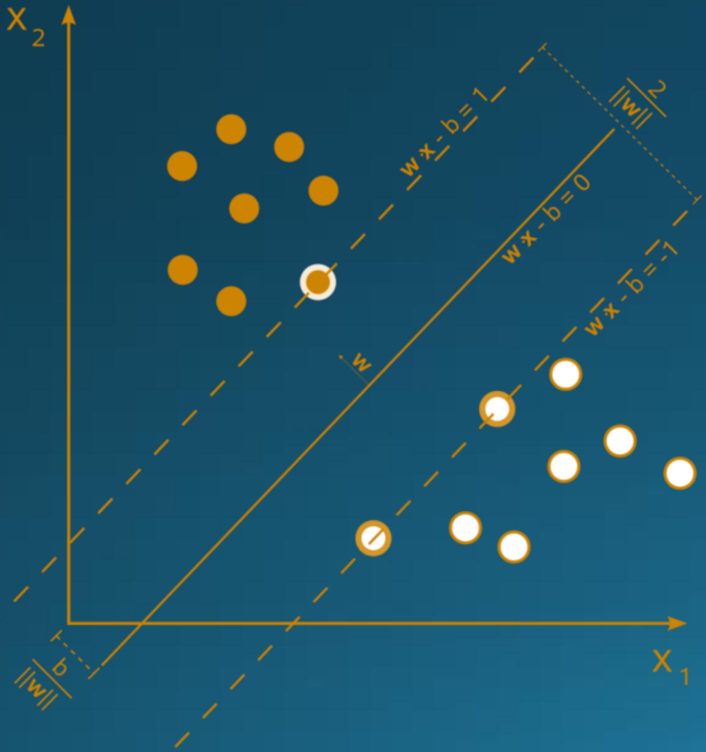
# Detail Slides

# Support Vector Machine

- Divides a features space

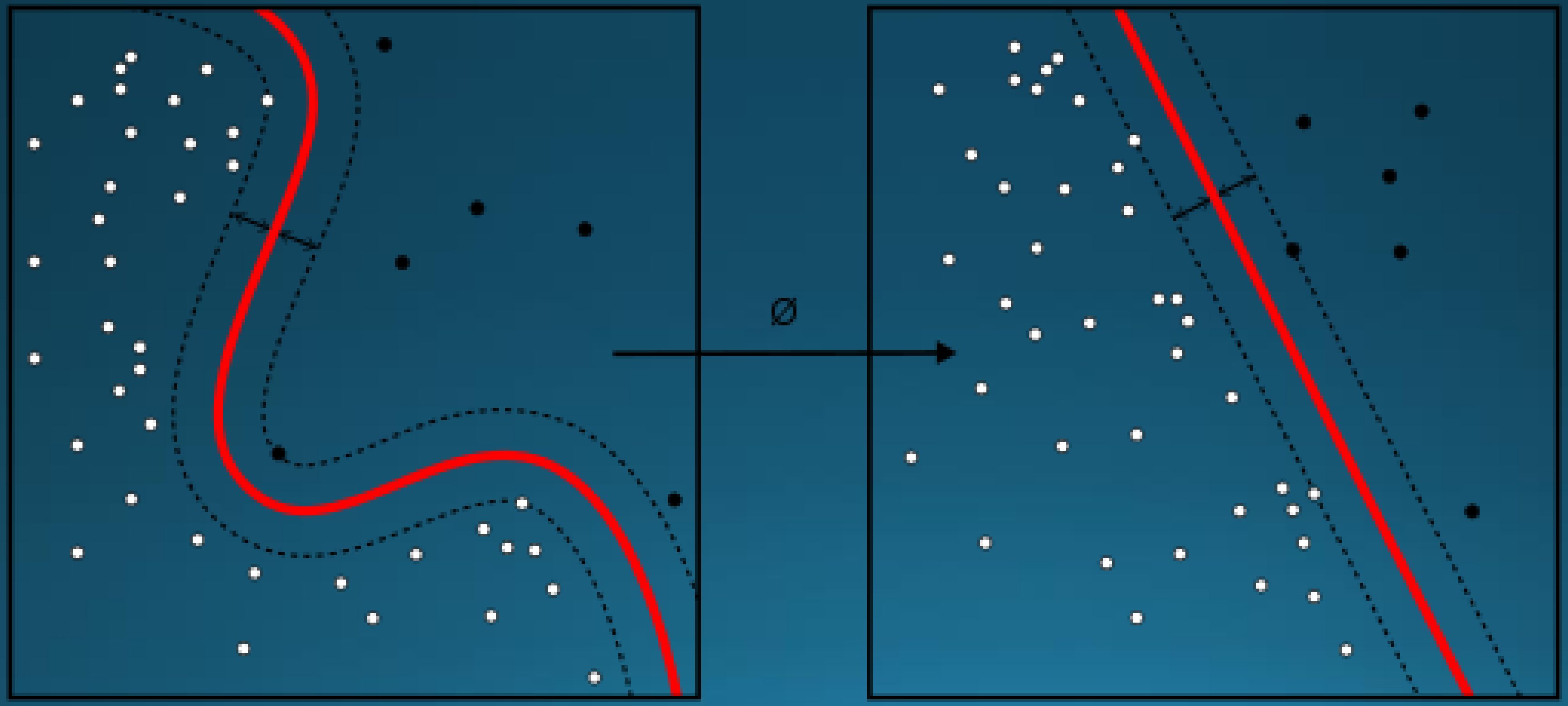
- Any sections will do… but these are the “best”





# Support Vector Machine

- Higher dimensional embedding – allows linear separation of non-linear features



# Support Vector Machine

- Multi-class classifiers can be constructed