Analysis of massive 3D data

Florent Lafarge
Inria Sophia Antipolis - Mediterranee

Materials available at https://team.inria.fr/titane/teaching/

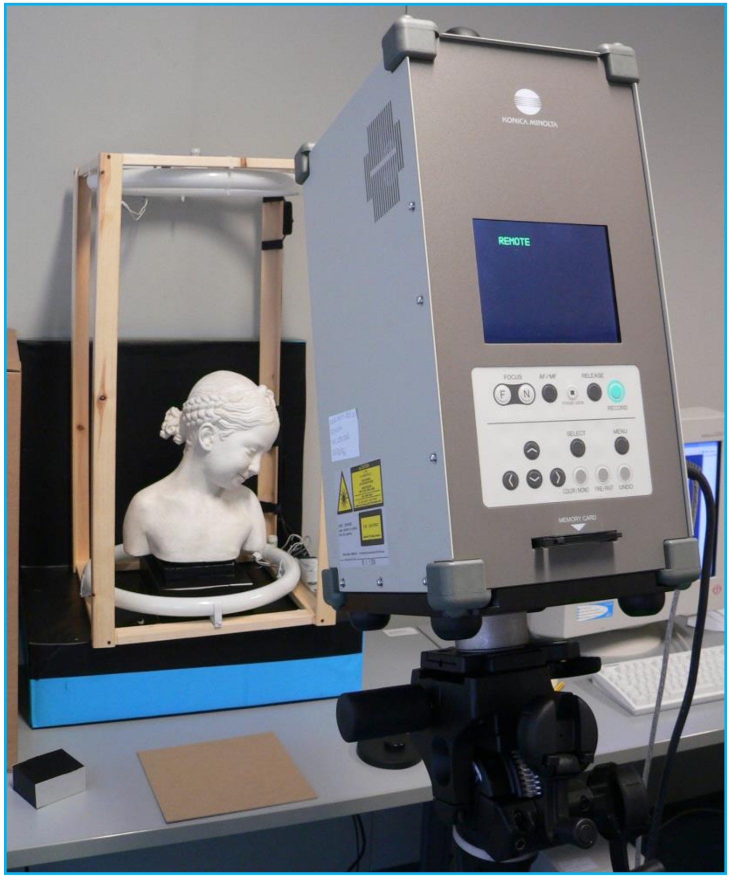
Contents

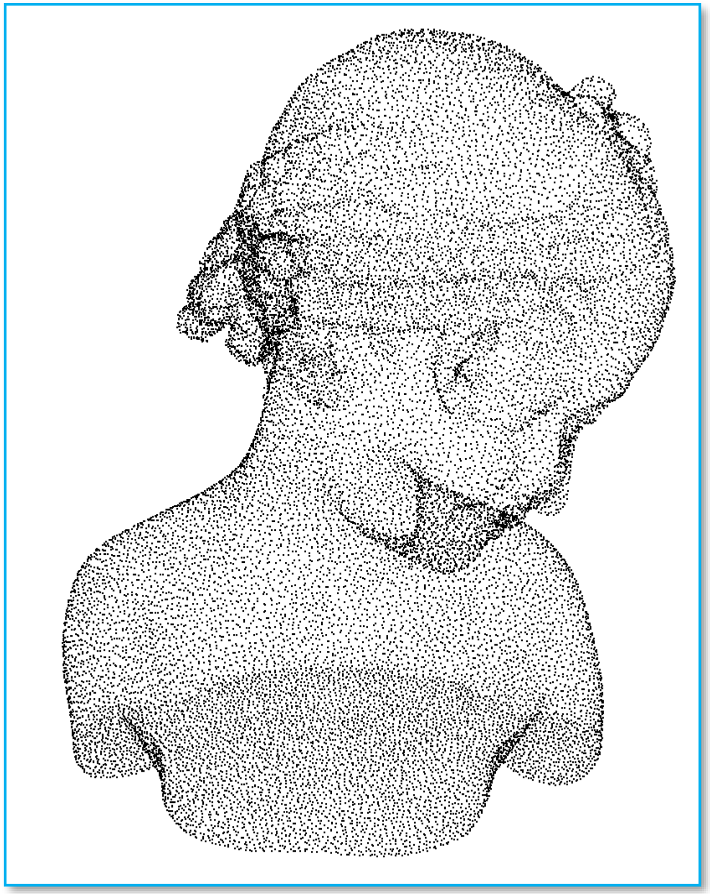
- Types of 3D data
- Local analysis of 3D data
- Scene classification

Types of 3D data

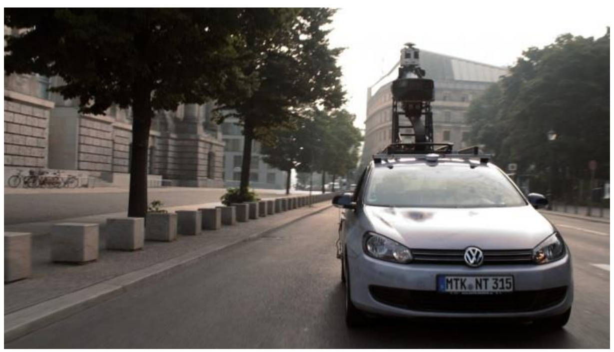
- Point cloud
- Surface mesh

Laser scanning

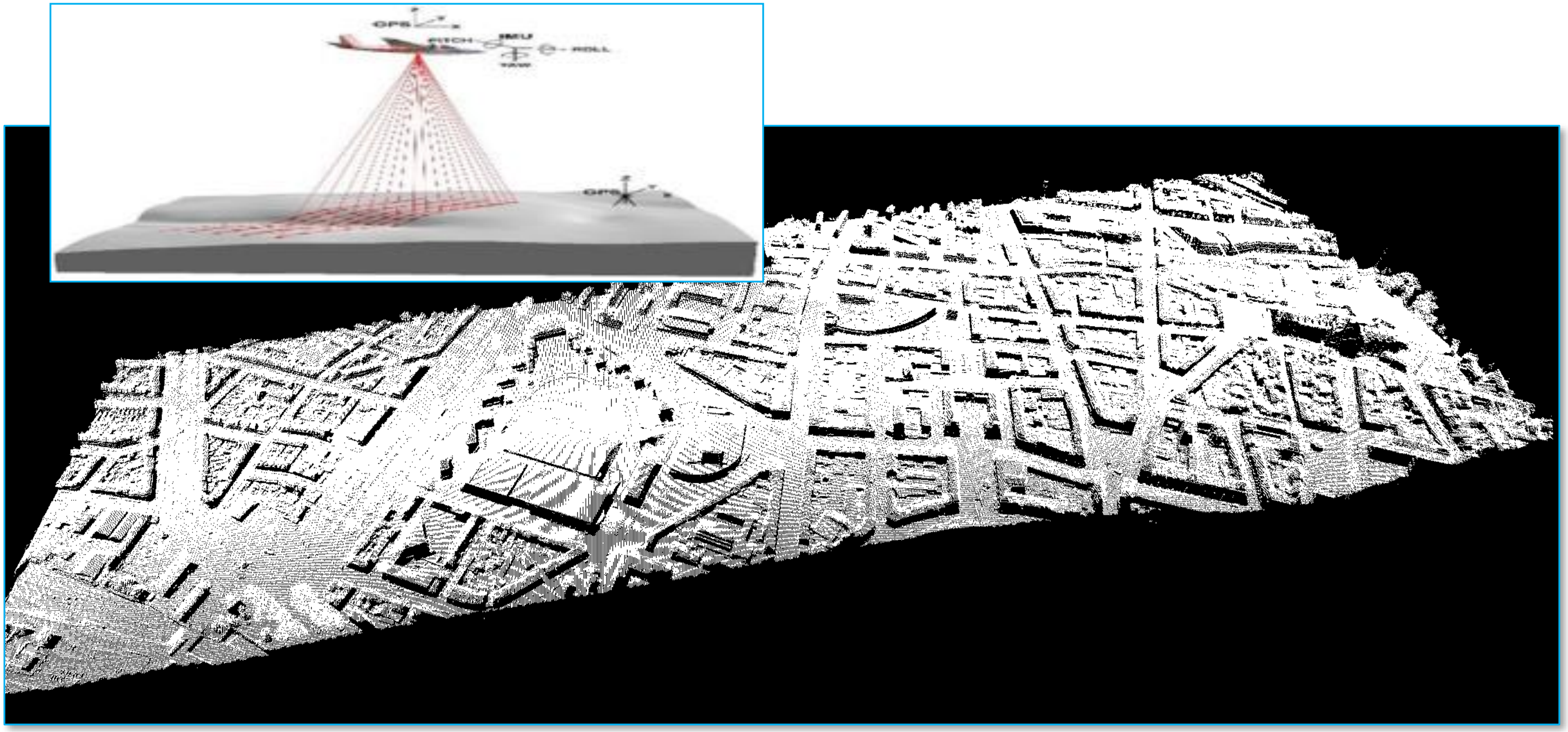




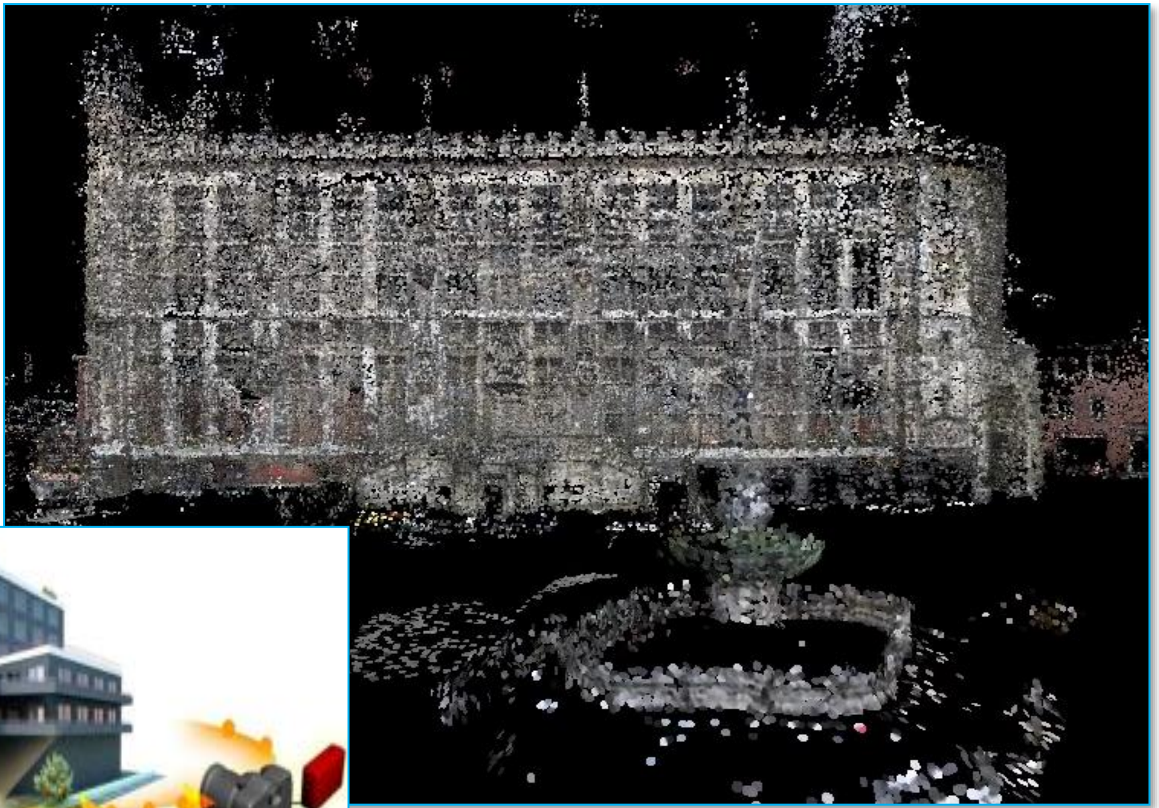
Car-based Laser



Airborne Lidar



Multi-View Stereo (MVS)



RGB-D sensors

Is a point just its spatial coordinates x-y-z ?

- Not necessarily: additionnal attributes can exist

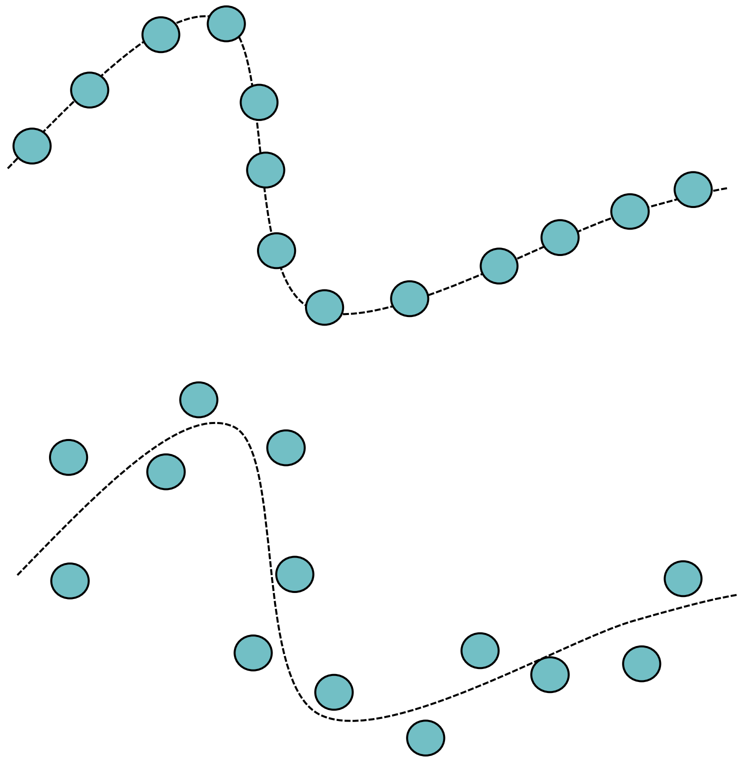
Is a point just its spatial coordinates x-y-z ?

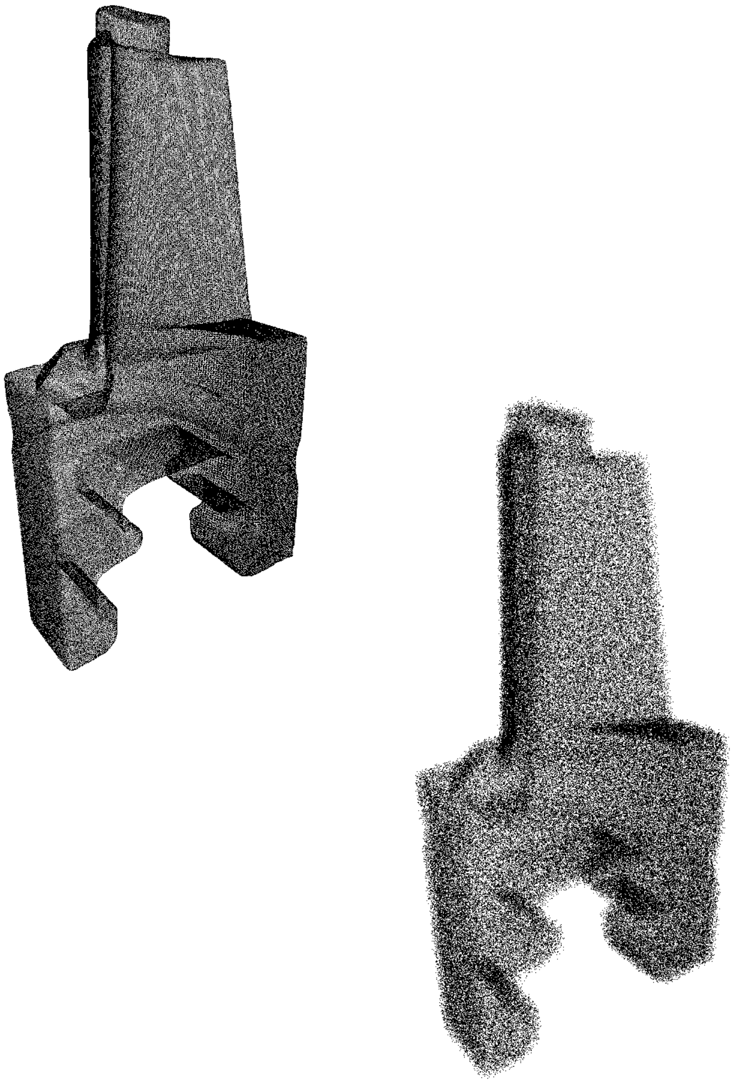
- Not necessarily: additionnal attributes can exist
- For each point :
  - Normal
  - Color
  - Confidence (MVS)
  - Camera index (MVS)

Defects in the point sets

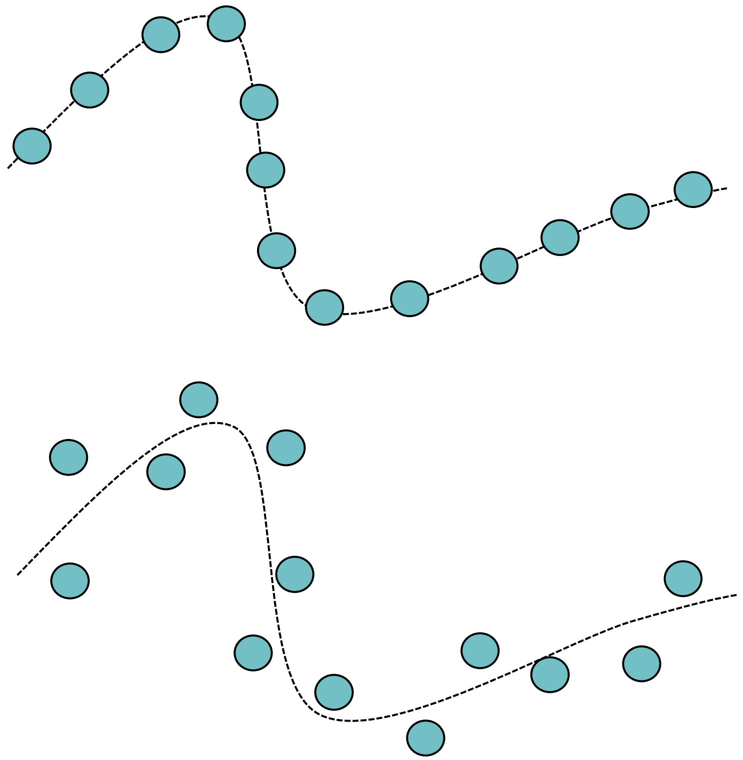
- noise
- outliers
- heterogeneous sampling
- missing data
- ..

Noise



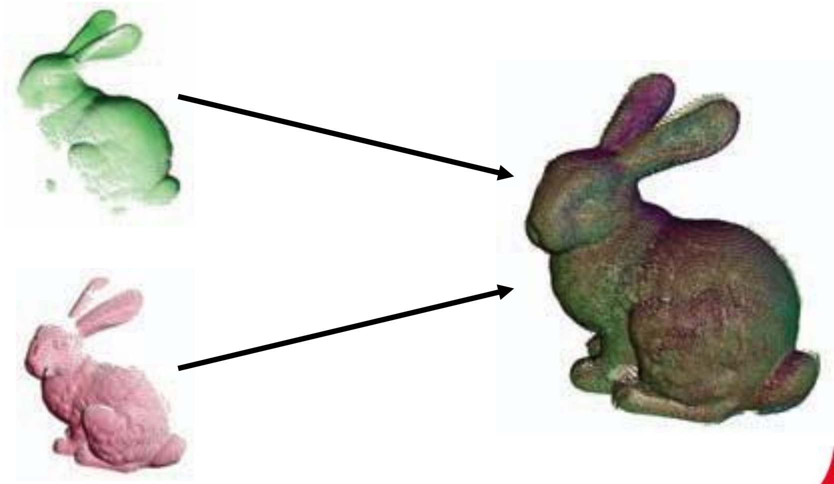


Noise

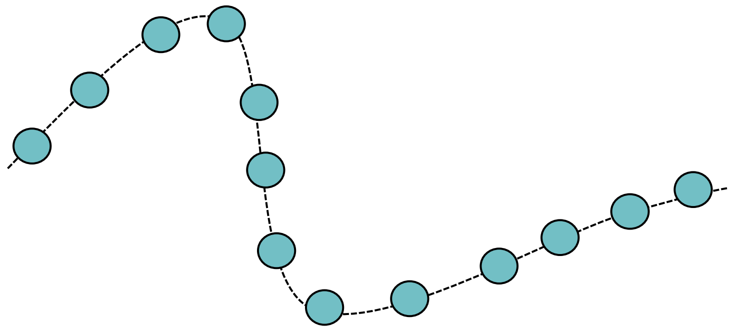


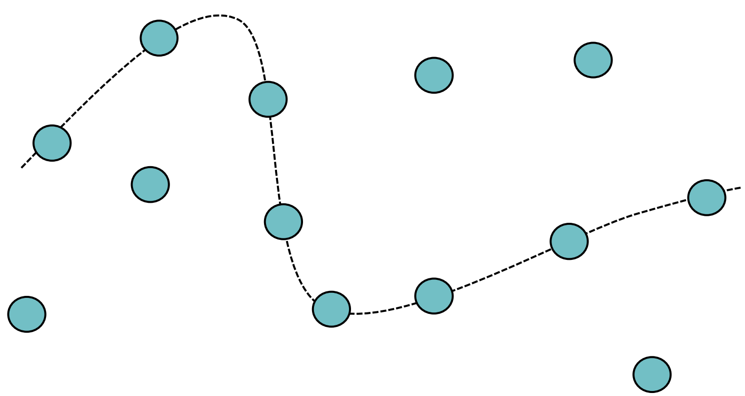
Inaccuracy

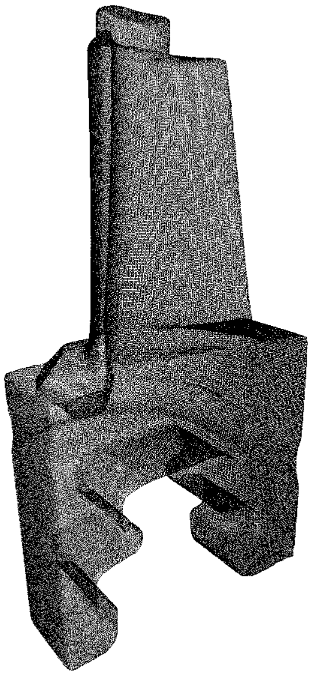
- acquisition system
- registration
- ..

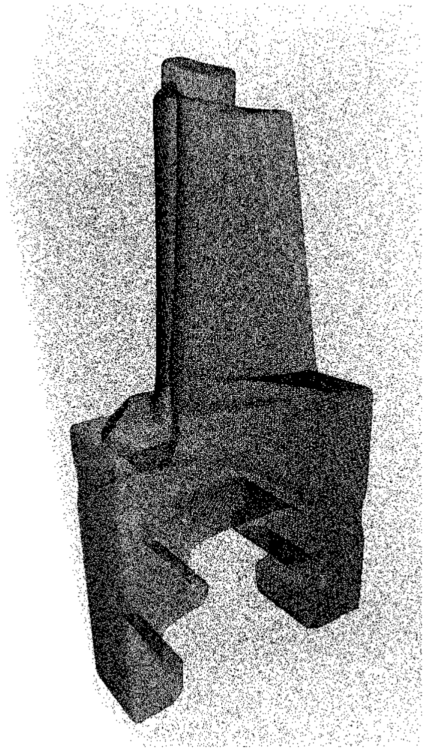


Outliers

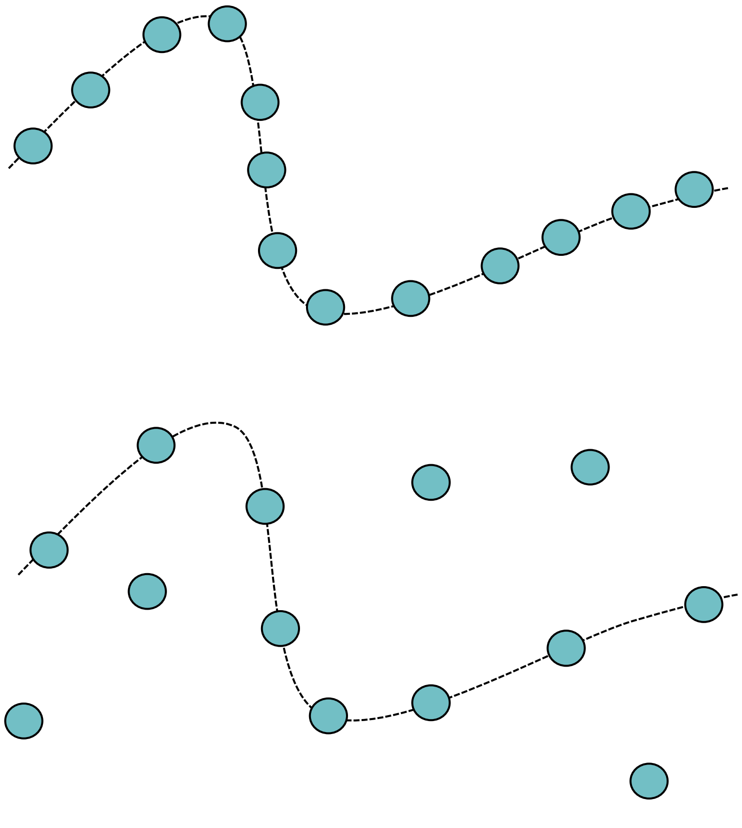




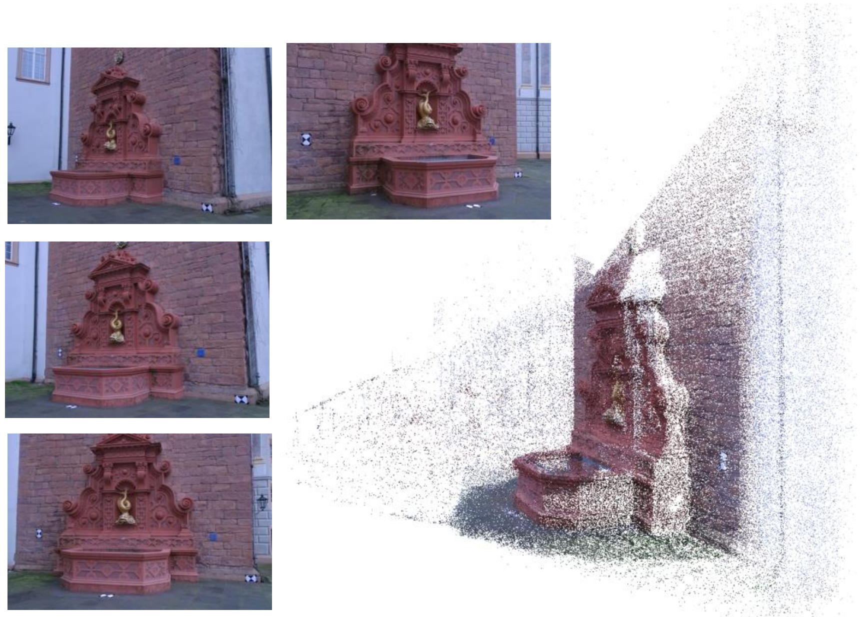




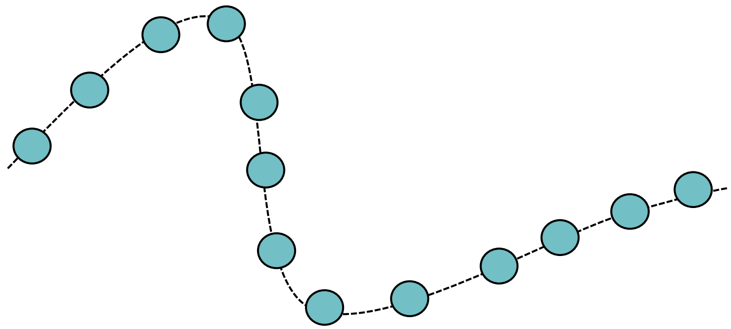
Outliers

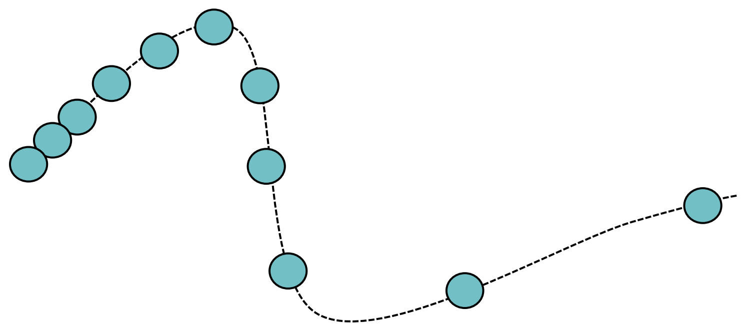


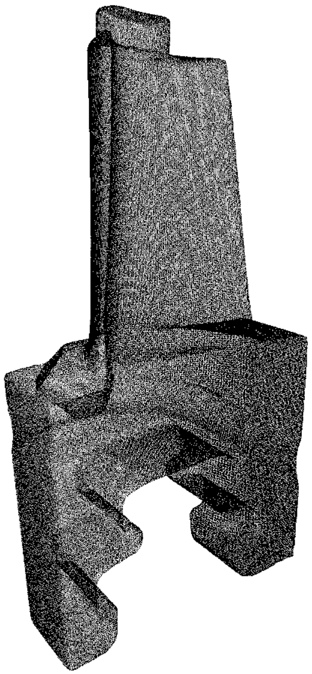
- Typical problem from multi-view stereo



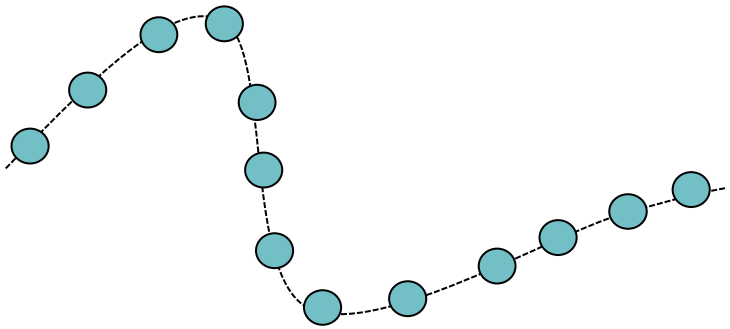
Heterogeneous sampling

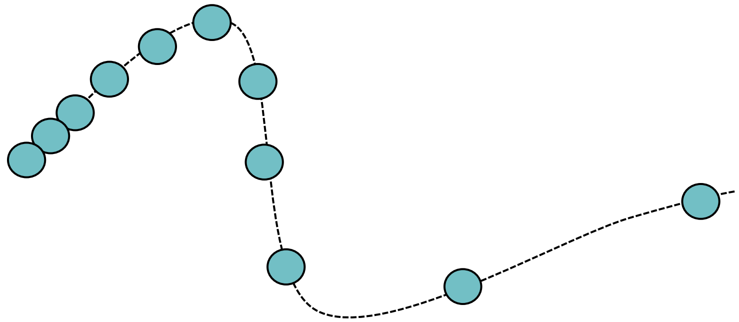




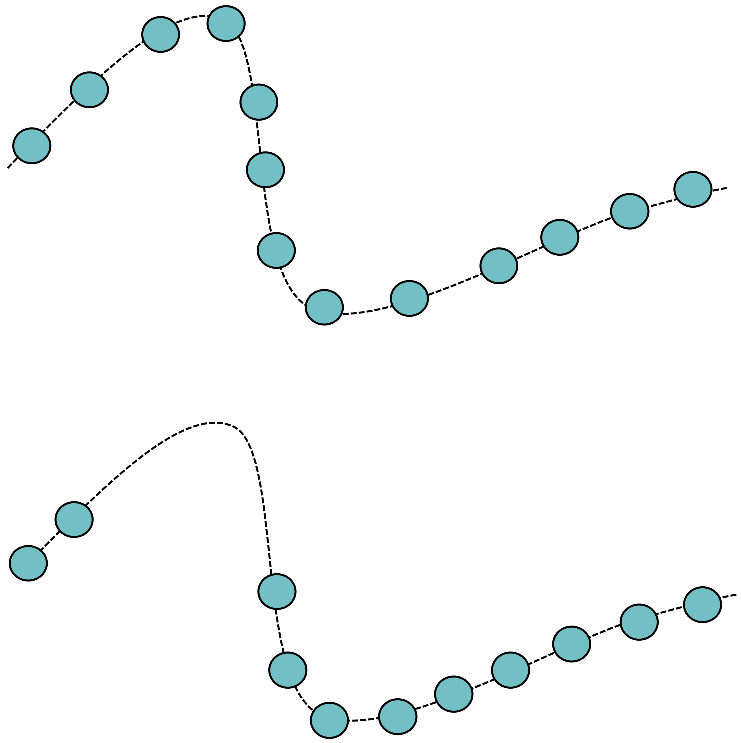


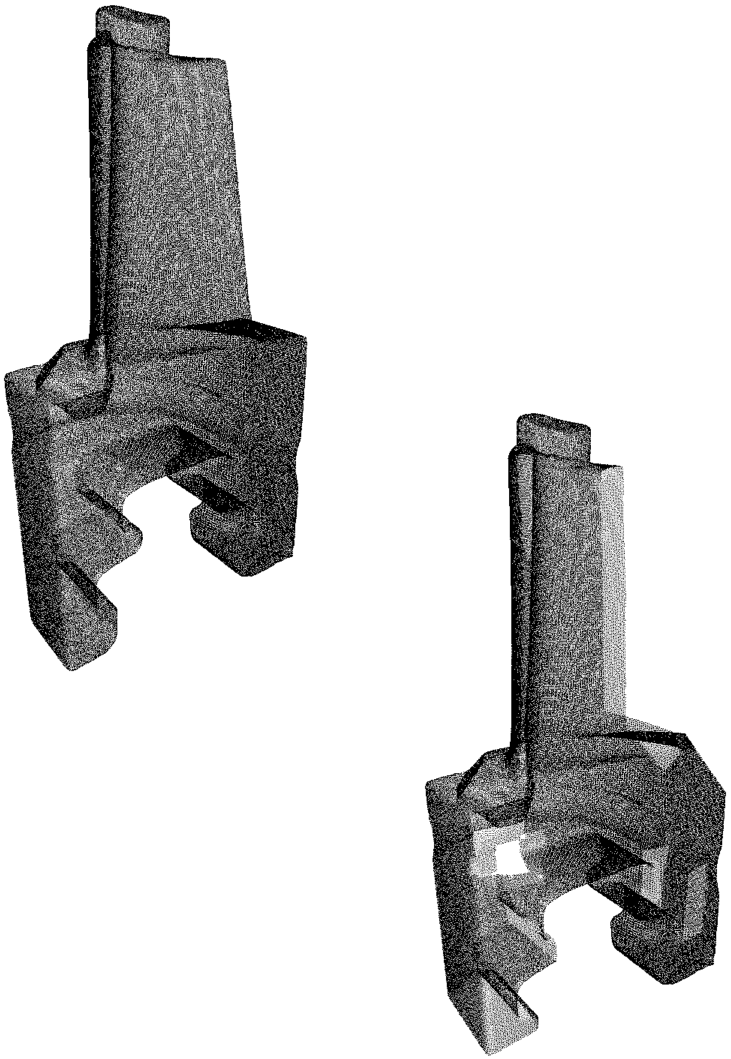
Heterogeneous sampling



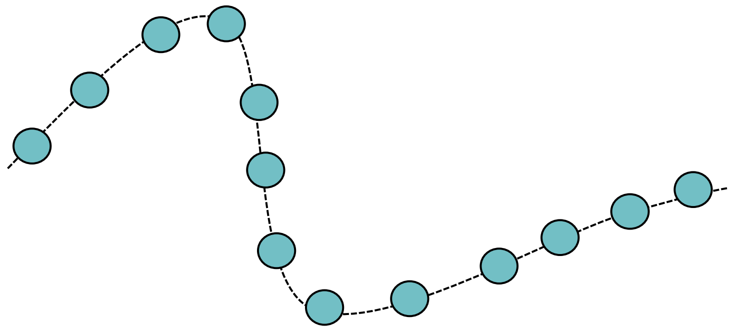


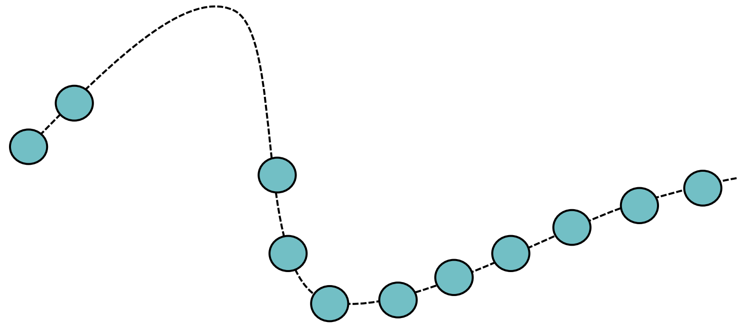
Missing data

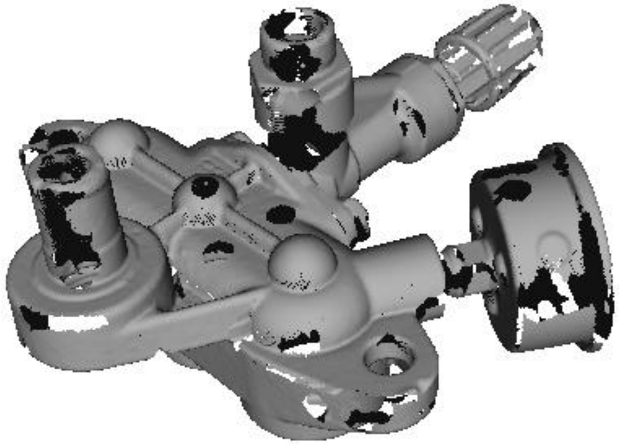




Missing data

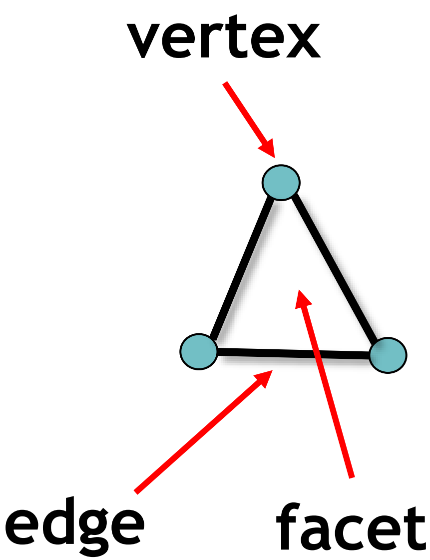






3D data as a surface mesh

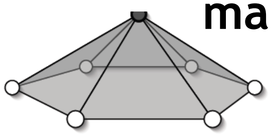
- the most common is the triangular mesh



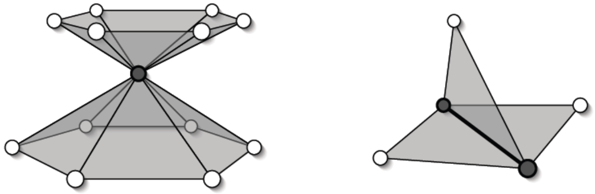
Properties of surfaces

- Manifold / Non-manifold

manifold

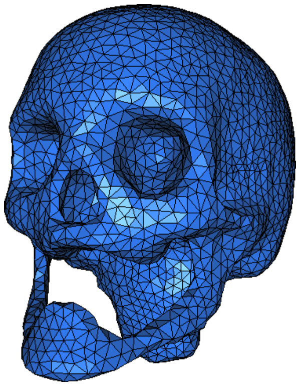


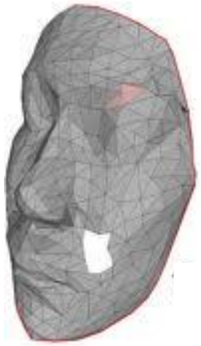
Non-manifold



Properties of surfaces

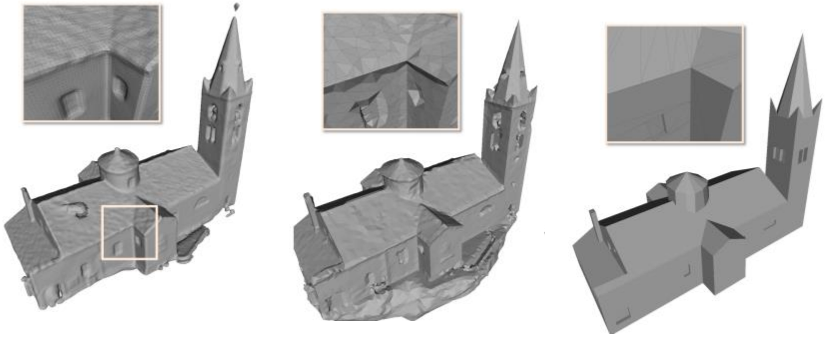
- Manifold / Non-manifold
- Watertight / with boundary





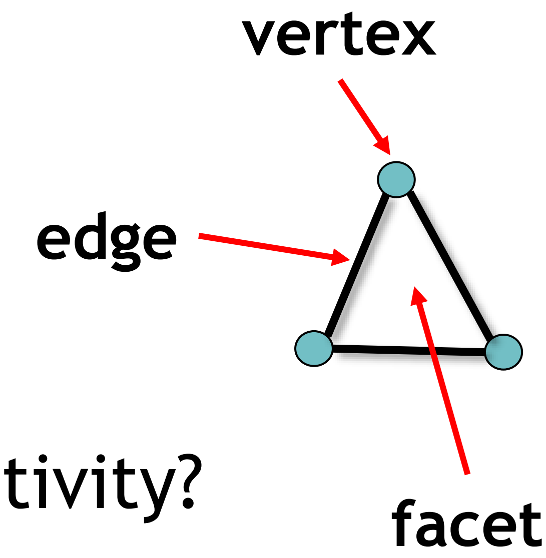
Properties of surfaces

- Manifold / Non-manifold
- Watertight / with boundary



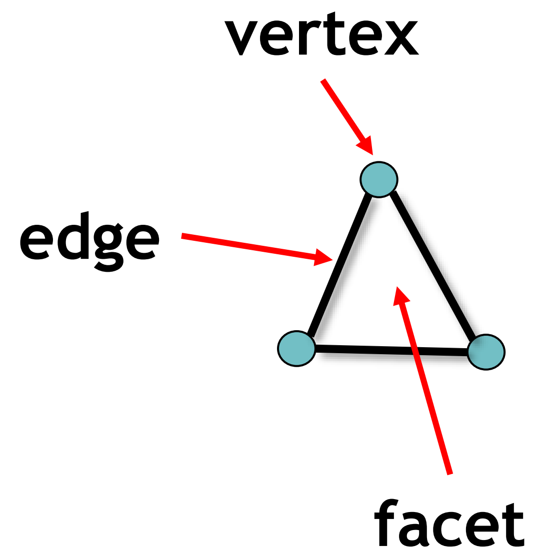
- Smooth/piecewise smooth/primitive-based

Mesh data structure

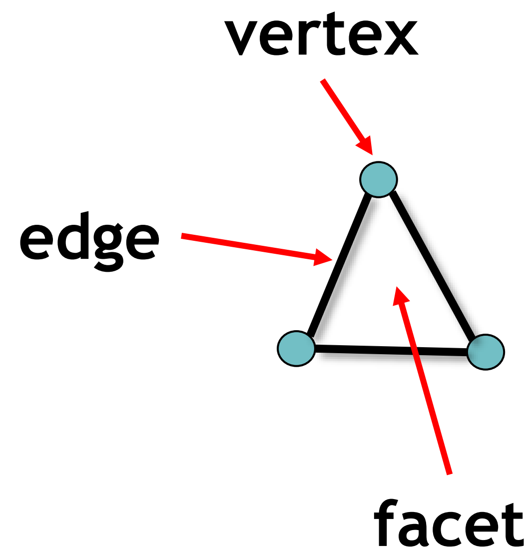


- How to store geometry and connectivity?
- Compact storage (File formats)
- Efficient algorithms on meshes
  - Identify time-critical operations
  - All vertices/edges of a face
  - All incident vertices/edges/faces of a vertex

Mesh data structure



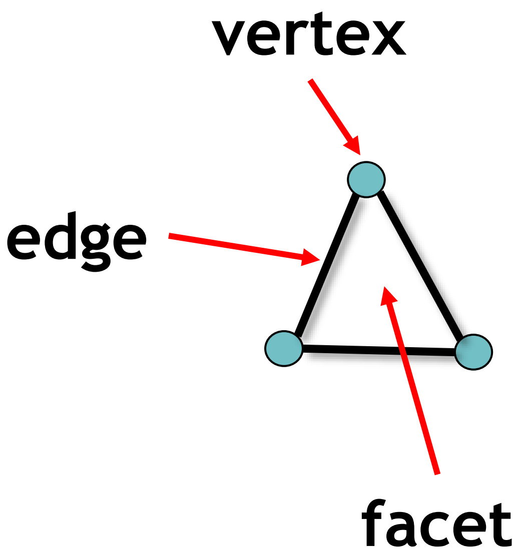
- What should be stored?
  - Geometry: 3D coordinates
  - Attributes
    - e.g. normal, color, texture coordinate
    - Per vertex, per face, per edge
  - Connectivity: What is adjacent to what



Mesh data structure

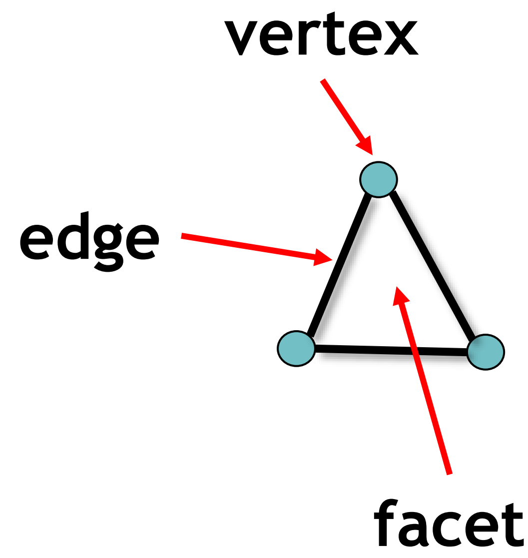
- What should it support?
  - Rendering
  - Queries
    - What are the vertices of face #3?
    - Is vertex #6 adjacent to vertex #12?
    - Which faces are adjacent to face #7?
  - Modifications
    - Remove/add a vertex/face, vertex split..

Mesh data structure



- How good is a data structure?
  - Time to construct (preprocessing)
  - Time to answer a query
  - Time to perform an operation

Facet set (STL)

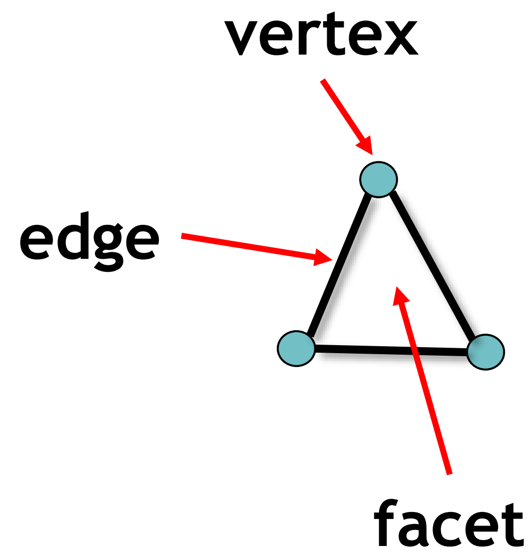


- Face:
  - 3 positions

Triangles		
x_{11} y_{11} z_{11}	x_{12} y_{12} z_{12}	x_{13} y_{13} z_{13}
x_{21} y_{21} z_{21}	x_{22} y_{22} z_{22}	x_{23} y_{23} z_{23}
...	...	...
x_{F1} y_{F1} z_{F1}	x_{F2} y_{F2} z_{F2}	x_{F3} y_{F3} z_{F3}

- 9 float / facets
- no connectivity!

Shared vertex (PLY,OBJ,OFF)



- Indexed Face List
  - Vertex: position
  - Face: vertex indices

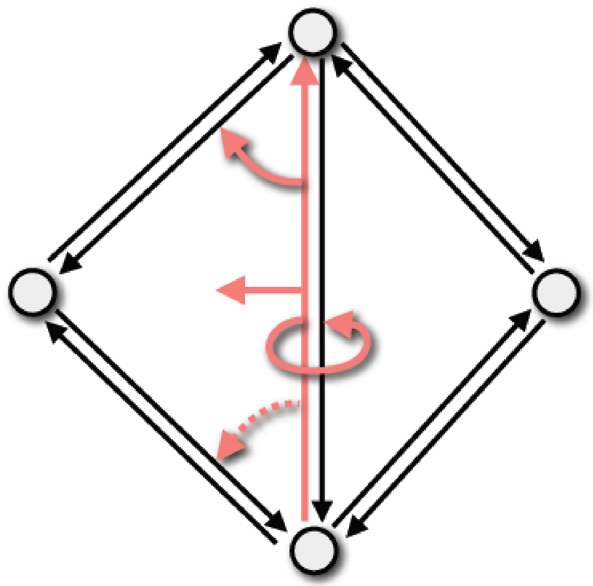
Vertices
x_1 y_1 z_1
...
x_V y_V z_V

Triangles
i_{11} i_{12} i_{13}
...
...
...
...
i_{F1} i_{F2} i_{F3}

- 3 float/vertex + 3 int/facet
- no neighborhood info

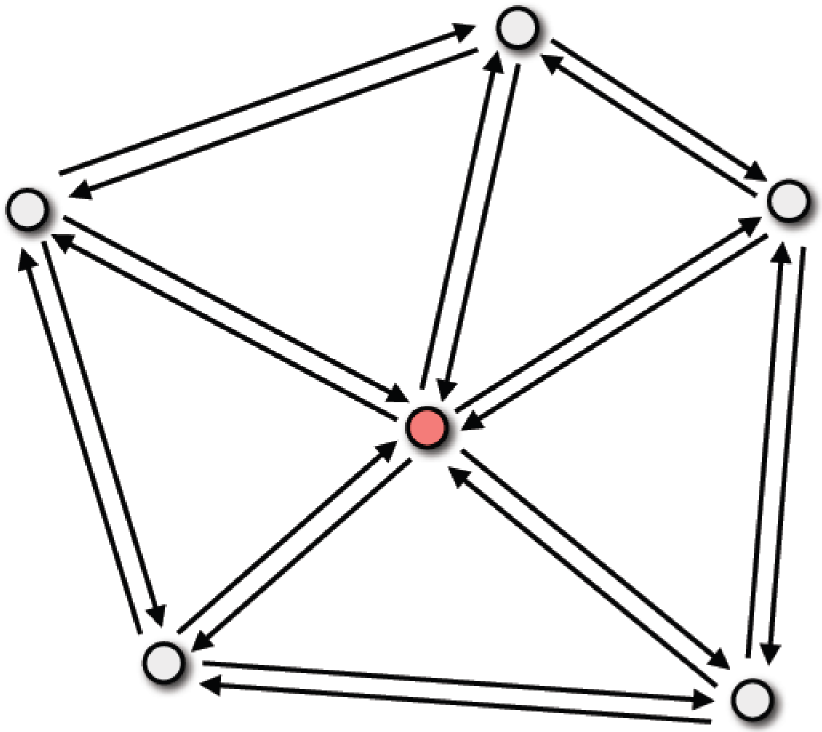
Halfedge-based connectivity

- Vertex:
  - position
  - 1 halfedge
- Halfedge:
  - 1 vertex
  - 1 facet
  - 1,2 or 3 halfedges
- Facet
  - 1 halfedge



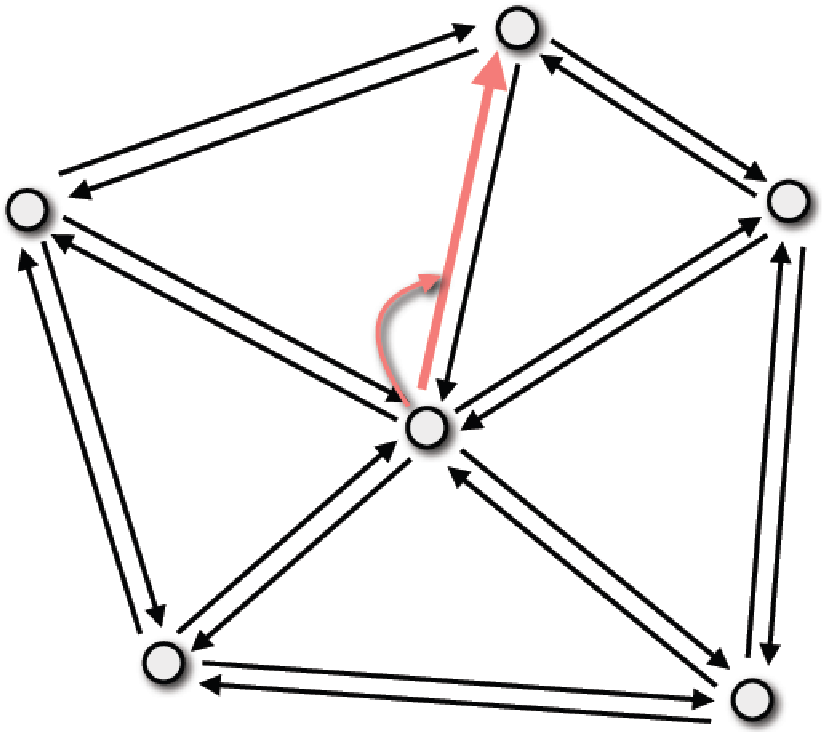
One-ring traversal

1. Start at vertex



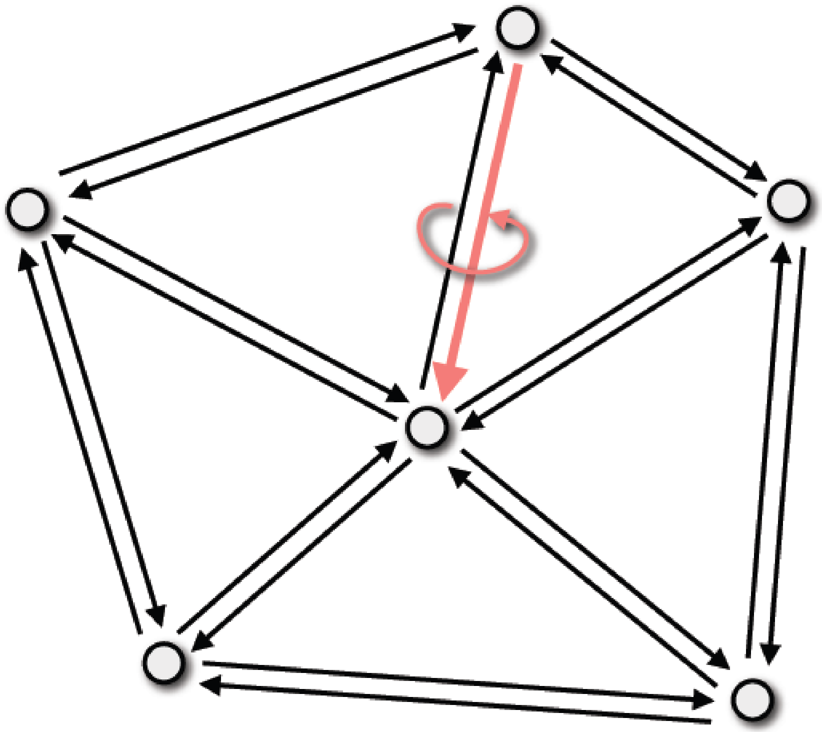
One-ring traversal

1. Start at vertex
2. Outgoing halfedge



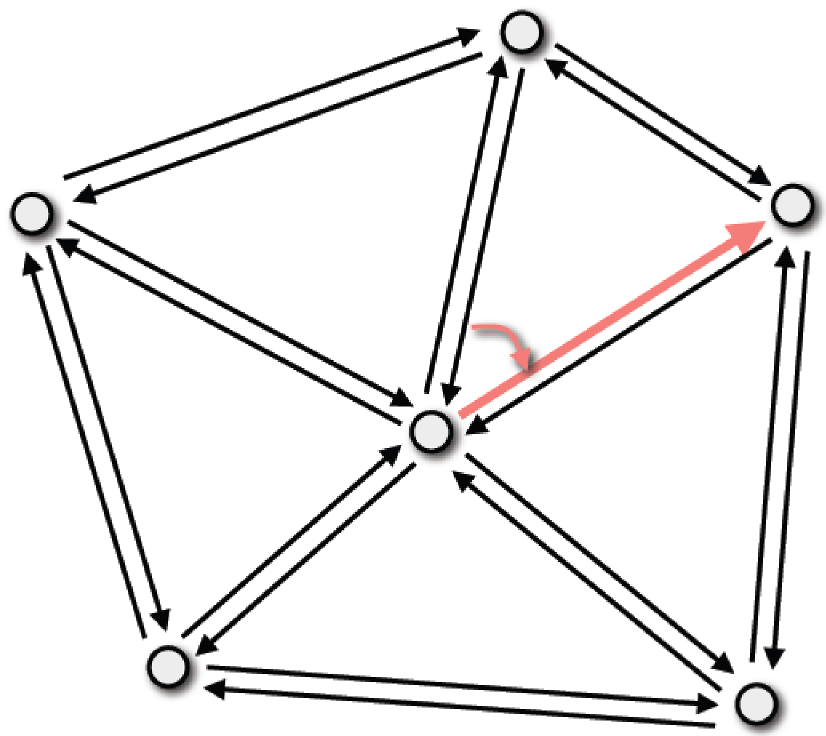
One-ring traversal

1. Start at vertex
2. Outgoing halfedge
3. Opposite halfedge



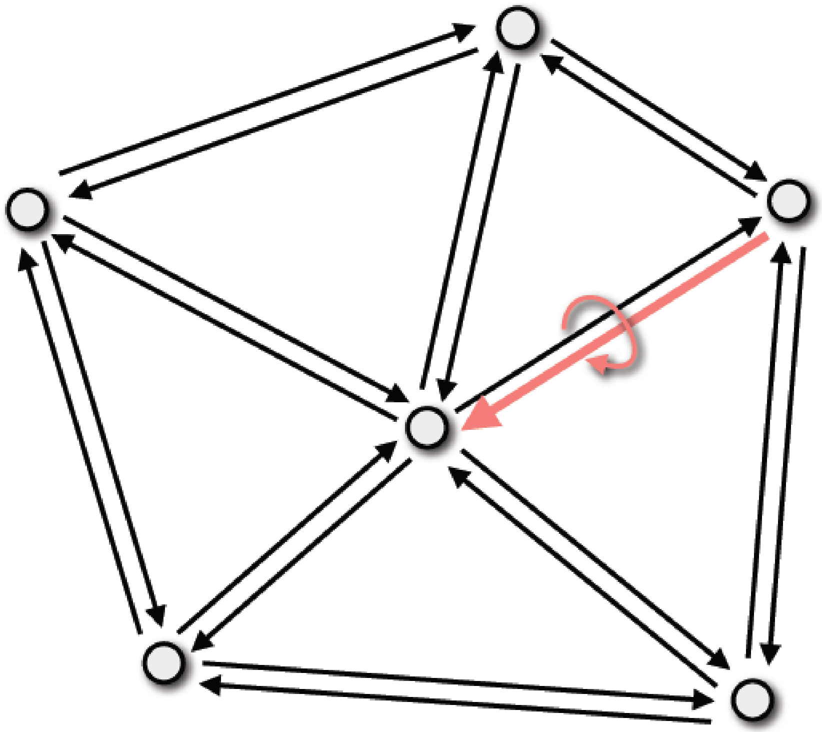
One-ring traversal

1. Start at vertex
2. Outgoing halfedge
3. Opposite halfedge
4. Next halfedge



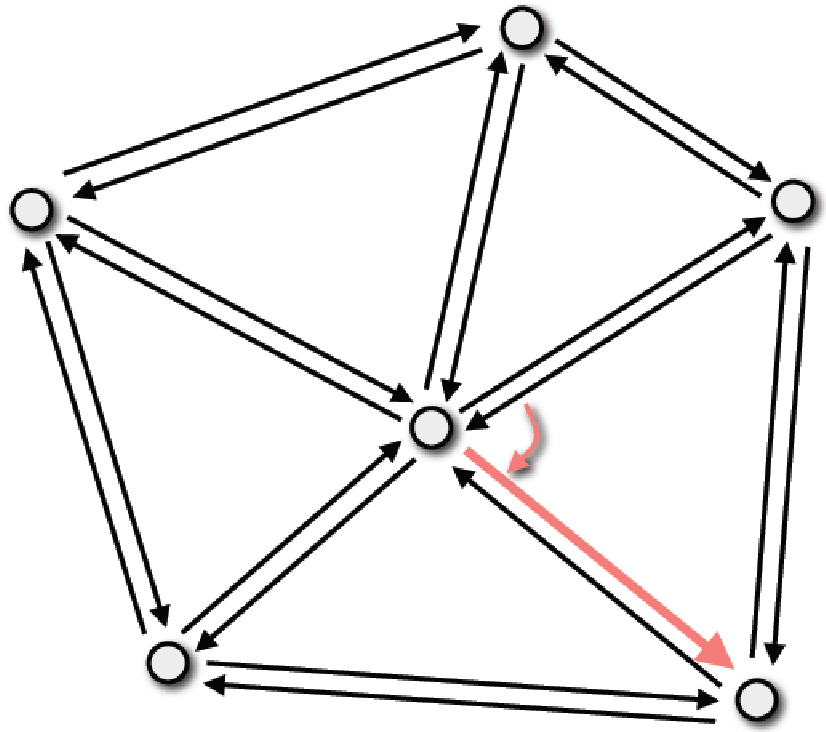
One-ring traversal

1. Start at vertex
2. Outgoing halfedge
3. Opposite halfedge
4. Next halfedge
5. Opposite



One-ring traversal

1. Start at vertex
2. Outgoing halfedge
3. Opposite halfedge
4. Next halfedge
5. Opposite
6. Next
7. ...



Halfedge-based libraries

- CGAL
  - www.cgal.org
  - Computational geometry
  - Free for non-commercial use

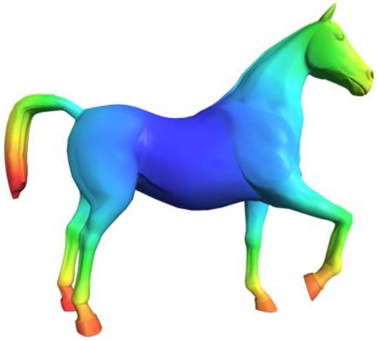
- OpenMesh
  - www.openmesh.org
  - Mesh processing
  - Free, LGPL licence

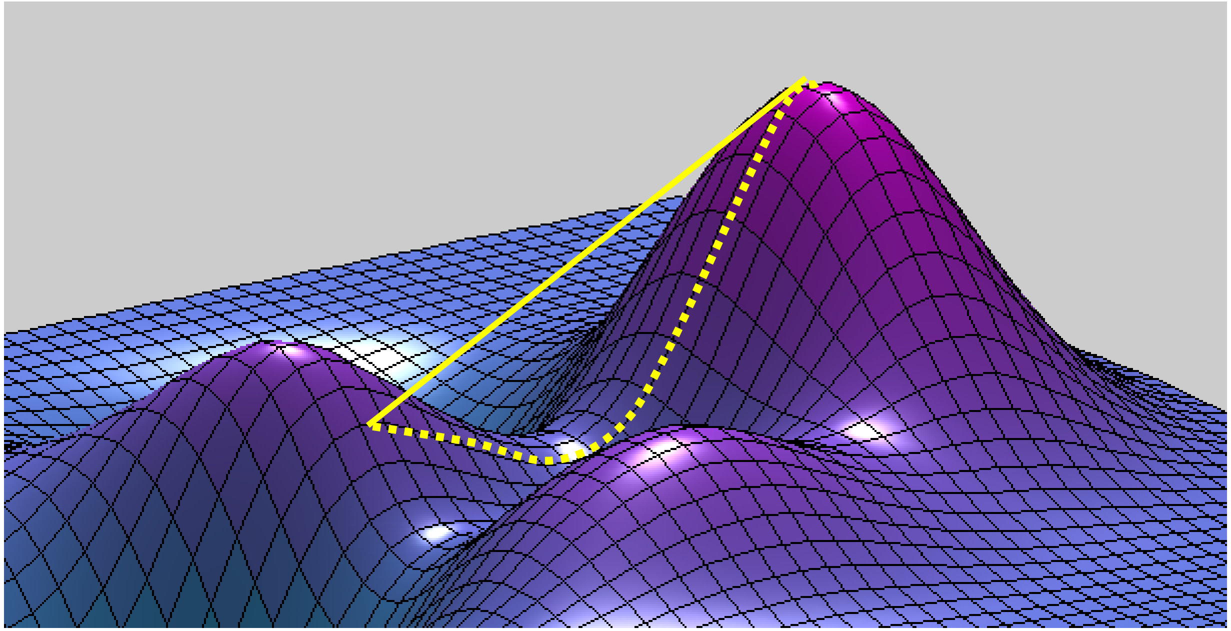
Local analysis of 3D data

- Geometric attributes
- Advanced 3D descriptors

Attributes

- Distance and Geodesic distance



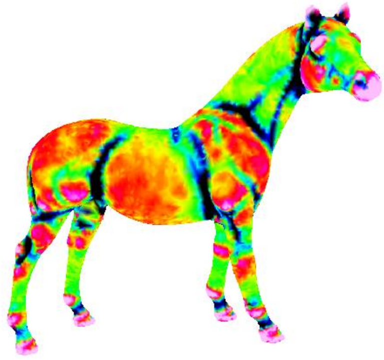


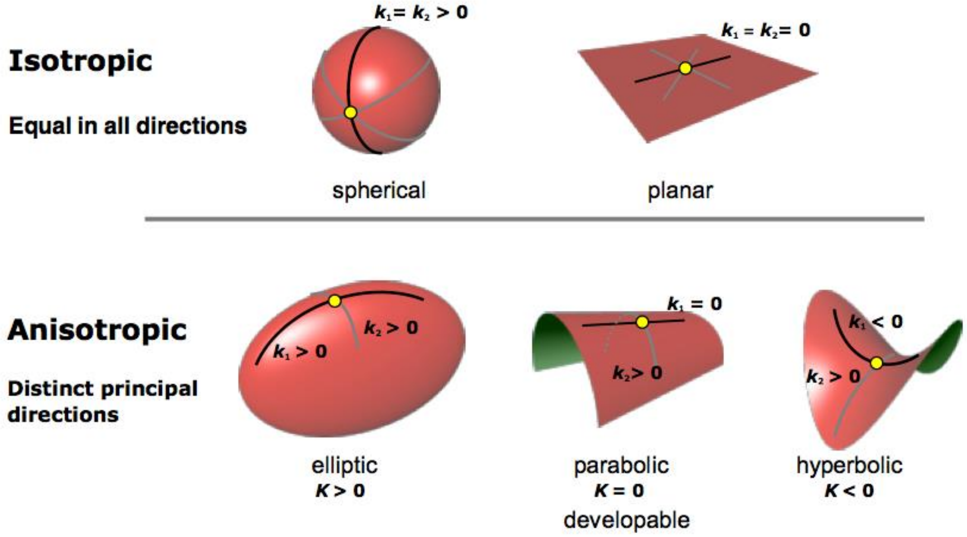
Attributes

- Distance and Geodesic distance
- Planarity, normal direction

Attributes

- Distance and Geodesic distance
- Planarity, normal direction
- Smoothness, curvature





Attributes

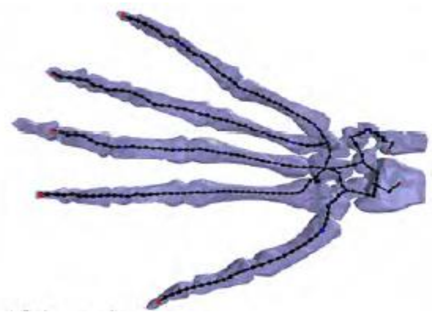
- Distance and Geodesic distance
- Planarity, normal direction
- Smoothness, curvature
- Distance to complex geometric primitives

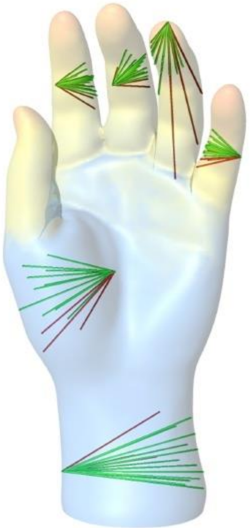
Attributes

- Distance and Geodesic distance
- Planarity, normal direction
- Smoothness, curvature
- Distance to complex geometric primitives
- Symmetry

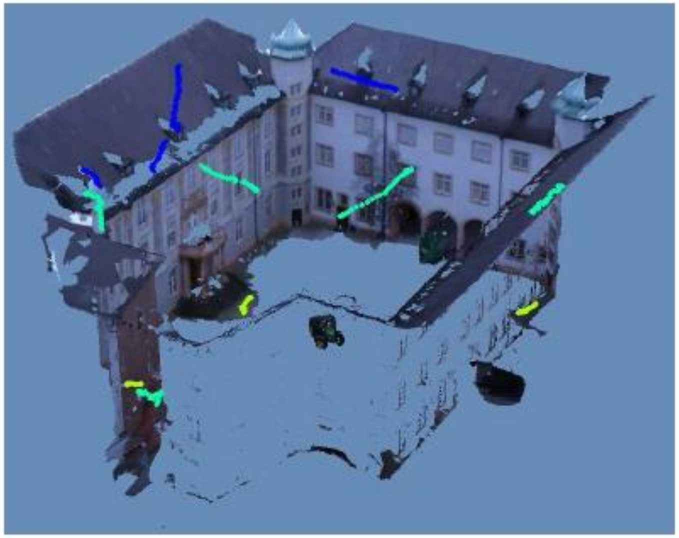
Attributes

- Distance and Geodesic distance
- Planarity, normal direction
- Smoothness, curvature
- Distance to complex geometric primitives
- Symmetry
- Medial Axis, Shape diameter

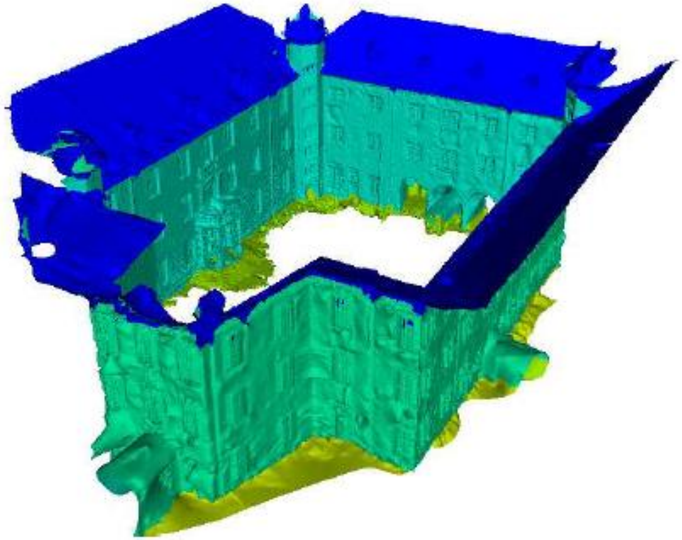




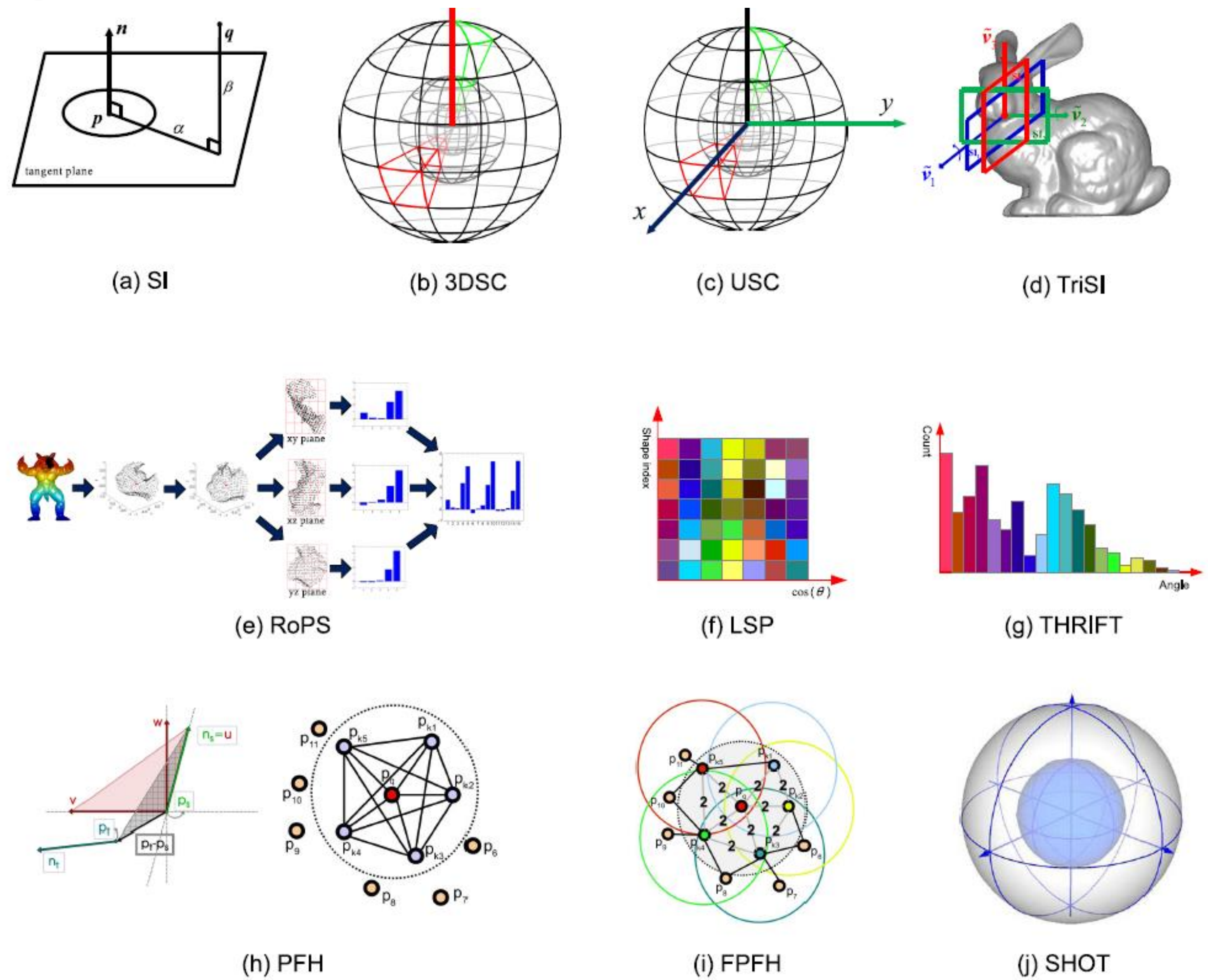
Attributes



- Distance and Geodesic distance
- Planarity, normal direction
- Smoothness, curvature
- Distance to complex geometric primitives
- Symmetry
- Medial Axis, Shape diameter
- Texture
- …



3D descriptors



Scene classification

- Unsupervised (MRF)
- Machine learning (Random Forest)
- Deep learning (PointNet)

Markov Random Fields (MRF)

set of random variables having a Markov property described by an undirected graph

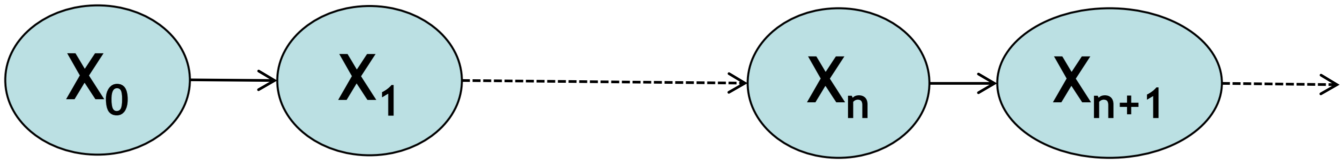
Let V be the set of nodes in the graph

Card(V) = number of random variables in the MRF

Markov property

- in 1D (Markov chain)

$$\Pr(X_{n+1} = x | X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = \Pr(X_{n+1} = x | X_n = x_n).$$

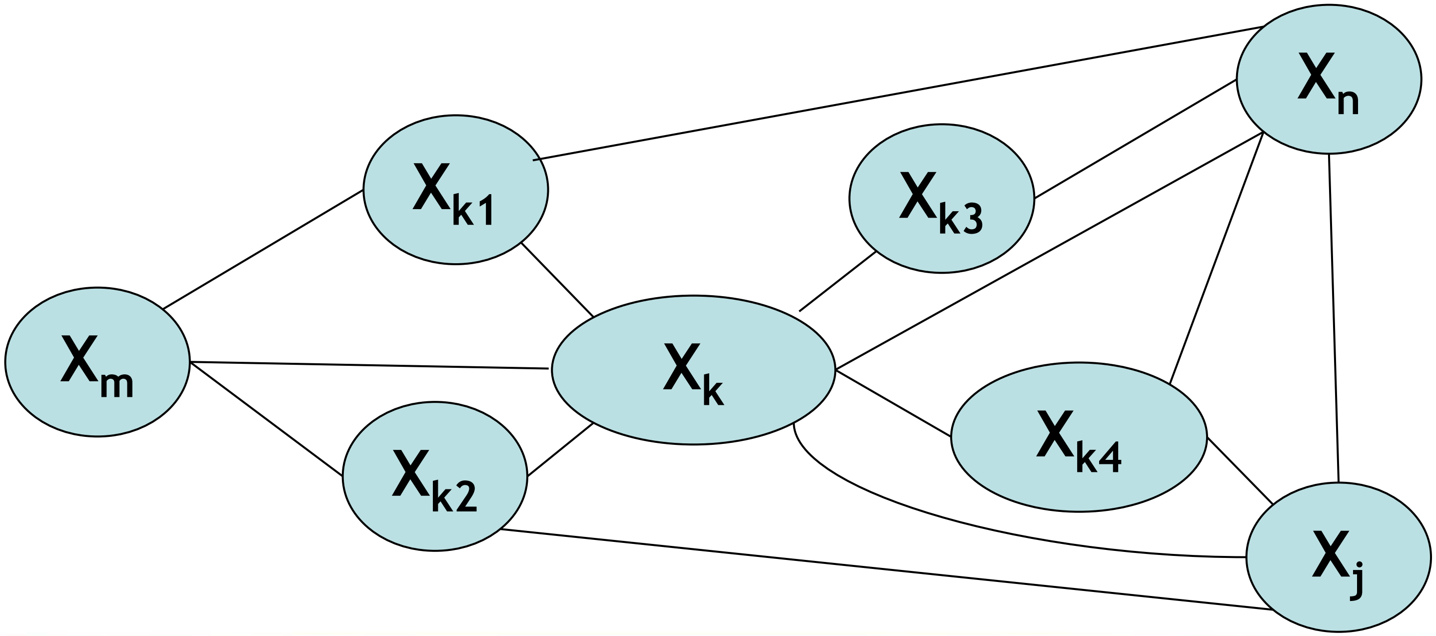


n usually corresponds to time

Markov property

- in 2D or on a manifold in 3D (Markov field)

$$P[\, X_k \mid X - \{Xk\} \,] = P[\, X_k \mid (X_{n(k)}) \,] \quad \text{with } n(k) \text{ neighbors of } k$$

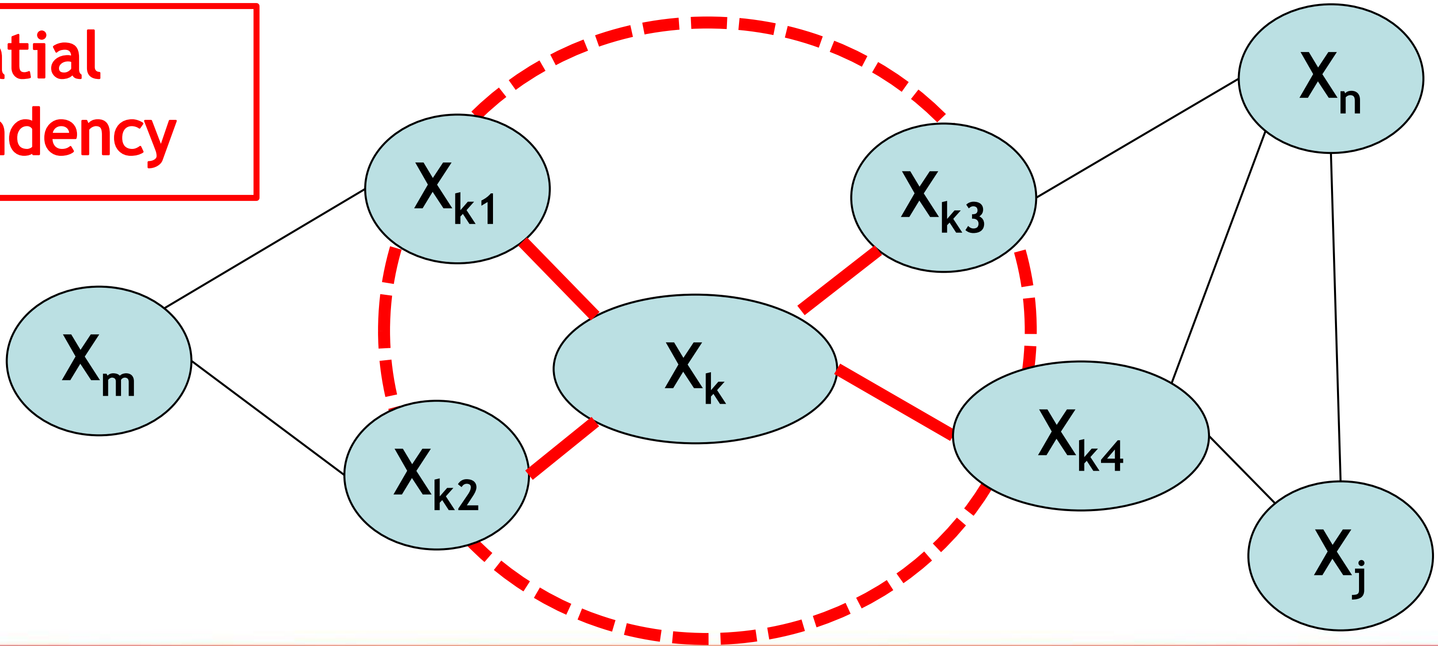


Markov property

- in 2D or on a manifold in 3D (Markov field)

$$P[\, X_k \mid X - \{Xk\} \,] = P[\, X_k \mid (X_{n(k)}) \,] \quad \text{with } n(k) \text{ neighbors of } k$$

Spatial dependency



Notion of neighborhood

$\mathcal{N} = \{n(i) \; / i \in V\}$ is a neighborhood system if

- (a) $i \notin n(i)$
- (b) $i \in n(j) \Leftrightarrow j \in n(i)$

- a MRF is always associated to a neighborhood system defining the dependency between graph nodes

MRF as an energy

- Gibbs energy (Hammersley-Clifford theorem)

- Let X be a MRF so that for all x $\in \Omega$, P(X=x)>0,

Then P(X) is a Gibbs distribution of the form

$$P(X=x) = \frac{\exp -U(x)}{Z}$$

- U is called a Gibbs energy

- $Z = \sum_{X \in \Omega} \exp -U(X)$

Markov property

- **Why is the markovian property important ?**

  - graph with 1M nodes
  - if each node is adjacent to every other nodes:
    1M*(999,999)/2 edges ~ 500 G edges

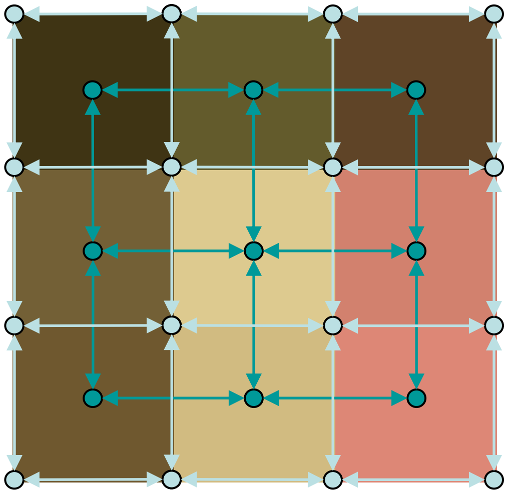
  - each random variable cannot be dependent to all the other ones
    $\Rightarrow$complexity needs to be reduced by spatial considerations

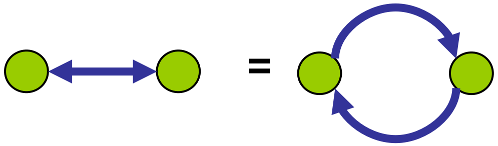
Markov Random Fields for images

- two common graphs

nodes = pixel centers
edges = adjacent pixels (4-connexity)

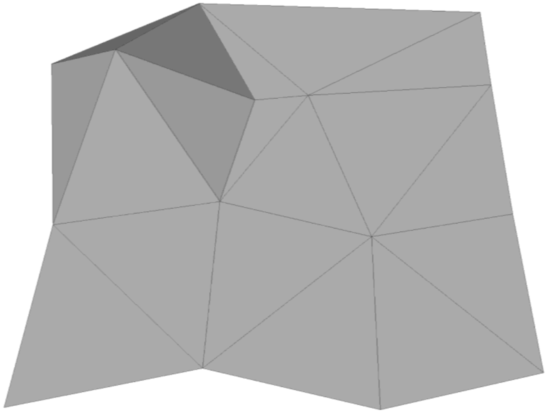
nodes = pixel corners
edges = pixel borders





Markov Random Fields for meshes

- Graph nodes = vertices & graph edges = edges
- Graph nodes = facets & graph edges = edges
- Graph nodes = edges & graph edges = facets



Bayesian formulation

Let y, the data (attributes)
x, the label

we want to model the probability of having x knowing y

$$\Pr(X = x \;/\; Y = y) = \frac{\Pr(Y = y \;/\; X = x) \;.\; \Pr(X = x)}{\Pr(Y = y)}$$

Bayes law

$$\underbrace{\Pr(X = x \;/\; Y = y)}_{\downarrow} \propto \underbrace{\Pr(Y = y \;/\; X = x)}_{\downarrow} \;.\; \underbrace{\Pr(X = x)}_{\downarrow}$$

Posterior probability **Likelihood** **Prior probability**

Standard assumptions

- conditional independence of the observation

$$P(Y=y|X=x) = \prod_{i \in V} P(y_i|x_i)$$

- X is an MRF

From probability to energy

- data term : local dependency hypothesis (l=x)
- regularization : soft constraints

$$U(l) = \underbrace{\sum_{i \in V} D_i(l_i)}_{\text{Data term}} + \beta \underbrace{\sum_{\{i,j\} \in E} V_{ij}(l_i, l_j)}_{\text{Regularisation term}}$$

Data term

= -log (likelihhood) when Bayesian

Regularisation term

= - log (pairwise interaction prior) when Bayesian

Optimal configuration

We search for the label configuration x that maximizes $P(X=x \mid Y=y)$

$$\longrightarrow \quad x^{*} = \arg\max_{x} \Pr(X=x|Y=y)$$

$$= \arg\min_{x} U(x)$$

exercise: binary segmentation

Graph structure

Graph nodes = facets
Graph edges = common edges

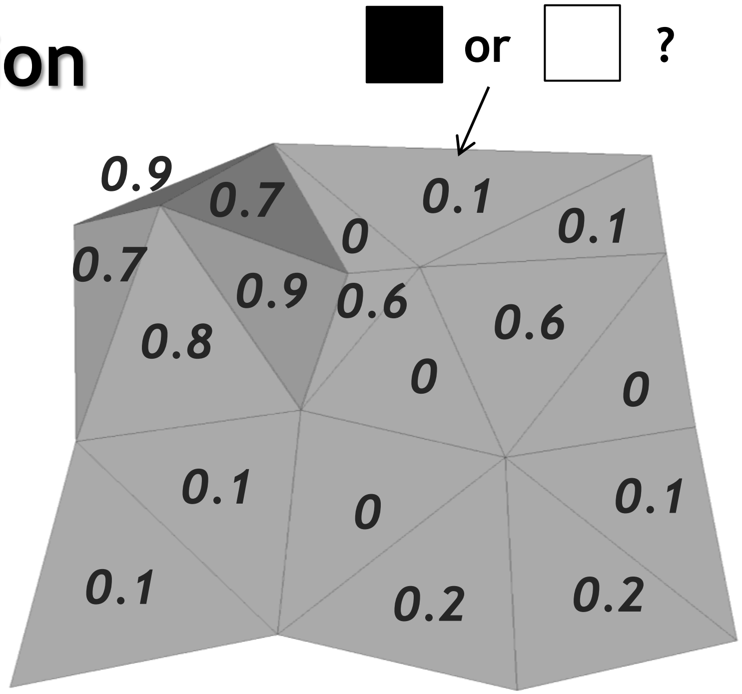
Attributes on facet: *[0,1]* (y)

labels: {*white*, *black*} (l)

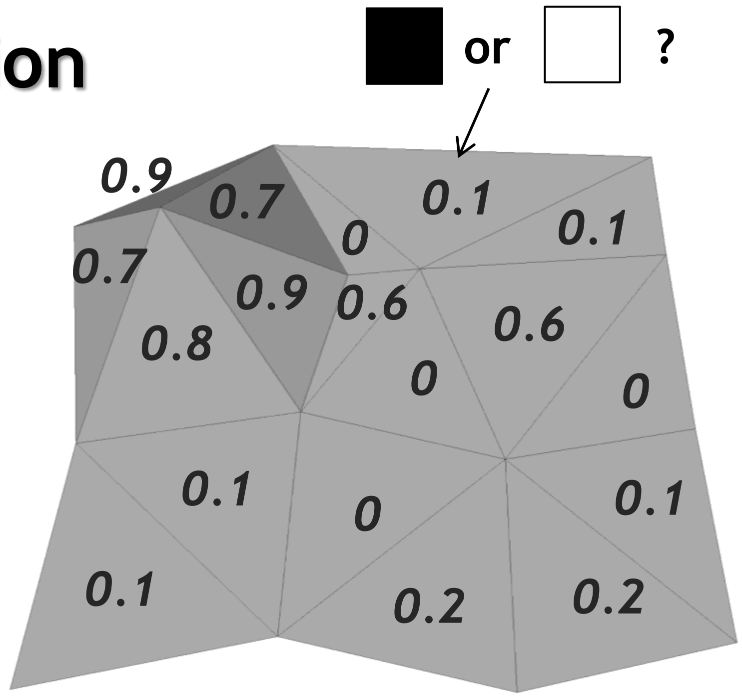
Energy: $U(l) = \sum_{i \in V} D_i(l_i) + \beta \sum_{\{i,j\} \in E} V_{ij}(l_i, l_j)$

with $D_i(l_i) = \begin{cases} y_i & \text{if } l_i = \text{'white'} \\ 1 - y_i & \text{otherwise} \end{cases}$

$$V_{i,j}(l_i, l_j) = \begin{cases} 0 & \text{if } \; l_i = l_j \\ 1 & \text{otherwise} \end{cases}$$



exercise: binary segmentation



Graph structure

Graph nodes = facets

Graph edges = common edges

Attributes on facet: *[0,1]* (y)

labels: *{white, black}* (l)

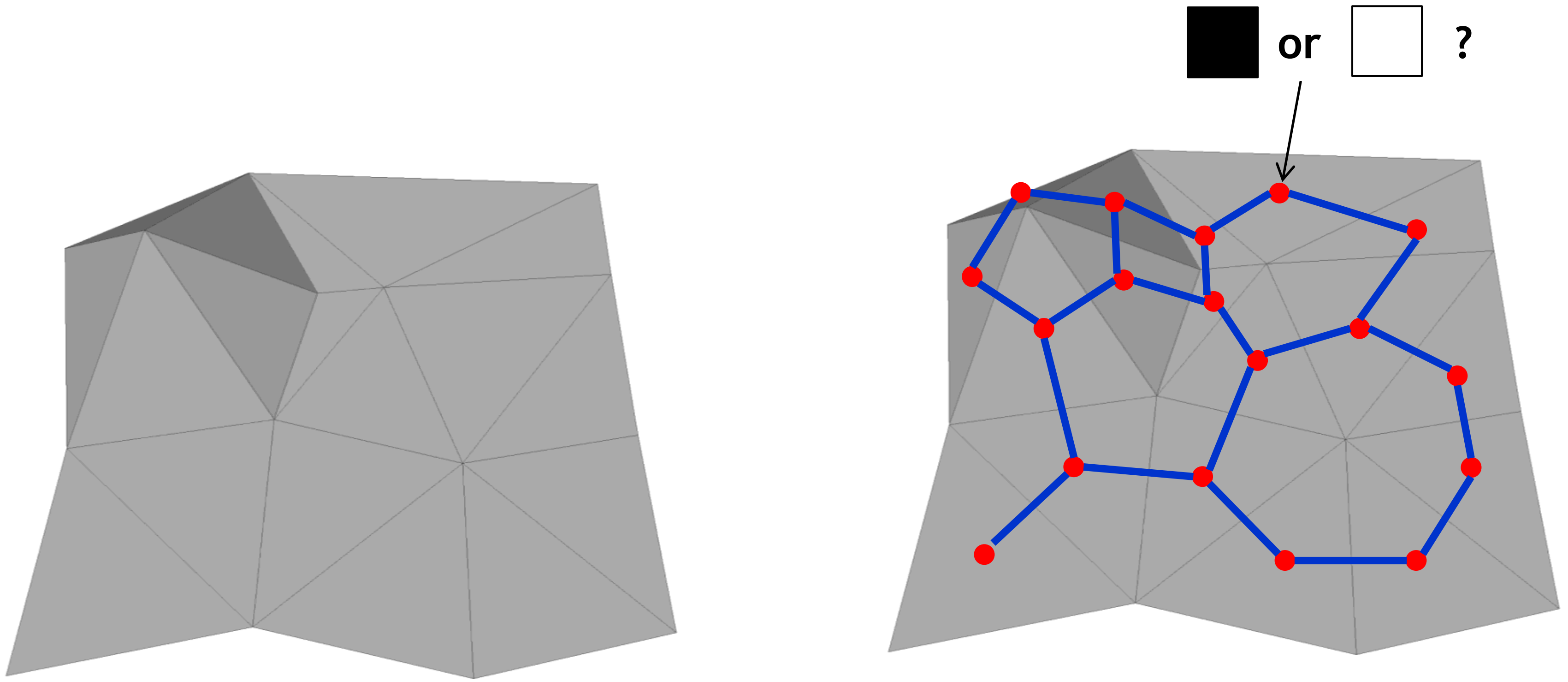
Energy: $U(l) = \sum_{i \in V} D_i(l_i) + \beta \sum_{\{i,j\} \in E} V_{ij}(l_i, l_j)$

with $D_i(l_i) = \begin{cases} y_i & \text{if } l_i = \text{'white'} \\ 1 - y_i & \text{otherwise} \end{cases}$

$$V_{i,j}(l_i, l_j) = \begin{cases} 0 & \text{if } l_i = l_j \\ 1 & \text{otherwise} \end{cases}$$

Q1: what is the optimal configuration l if $\beta = 0$? What is its energy ?
Q2: what is the optimal configuration if $\beta \to \inf$?
Q3: what are the other possible optimal configurations in function of β ?

exercise: binary segmentation



Finding the optimal configuration of labels

Graph-cut based approches

fast but restrictions on energy formulation

Monte Carlo sampling

slow but no restriction

Example: mesh segmentation with principal curvature attributes & soft geometric constraints

Multi-label energy model of the form

$$U(l) = \sum_{i \in V} D_i(l_i) + \beta \sum_{\{i,j\} \in E} V_{ij}(l_i, l_j)$$

with V, set of vertices of the input mesh

E, set of edges in the mesh

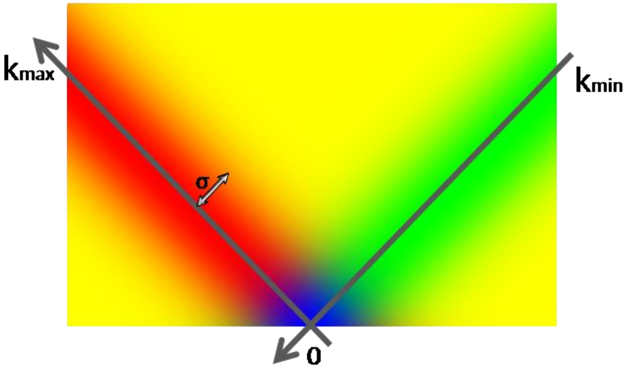
l_i, the label of the vertex i among : *planar* (1), *developable convex* (2), *developable concave* (3) and *non developable* (4)

Data term

$$D_i(l_i) = 1 - Pr(l_i | k_{min}^{(i)}, k_{max}^{(i)})$$

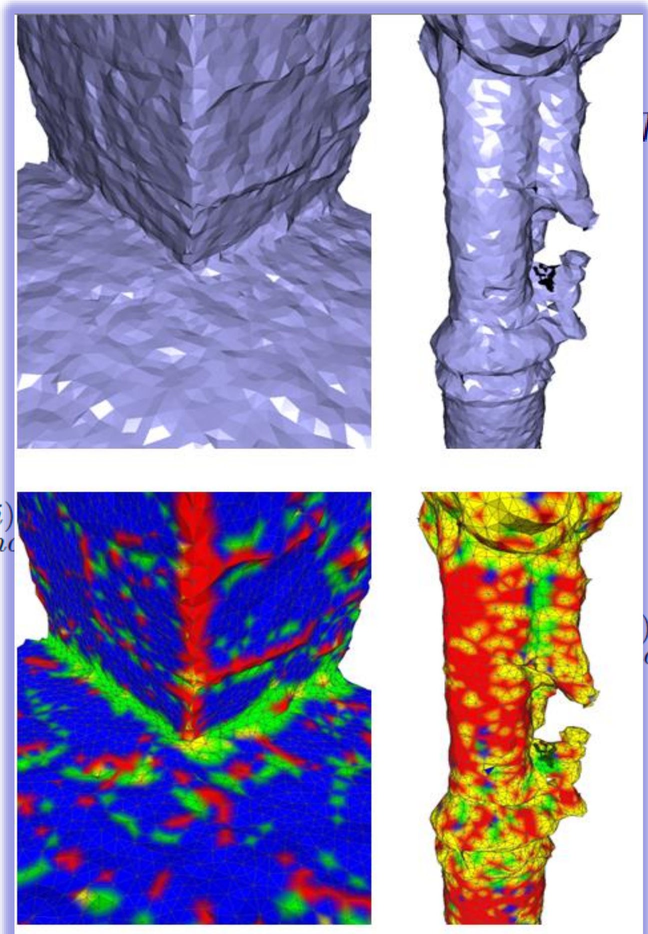
with

$$Pr(l_i | k_{min}^{(i)}, k_{max}^{(i)}) = \begin{cases} G_\sigma(k_{min}^{(i)}) G_\sigma(k_{max}^{(i)}) & \text{if } l_i = 1 \\ G_\sigma(k_{min}^{(i)})(1 - G_\sigma(k_{max}^{(i)})) & \text{if } l_i = 2 \\ (1 - G_\sigma(k_{min}^{(i)})) G_\sigma(k_{max}^{(i)}) & \text{if } l_i = 3 \\ (1 - G_\sigma(k_{min}^{(i)}))(1 - G_\sigma(k_{max}^{(i)})) & \text{if } l_i = 4 \end{cases}$$



$$G_\sigma(k) = \exp(-k^2 / 2\sigma^2)$$

Data term



$$r(l_i|k_{min}^{(i)}, k_{max}^{(i)})$$

with

$$Pr(l_i|k_{min}^{(i)}, k_{m}^{(i)}$$

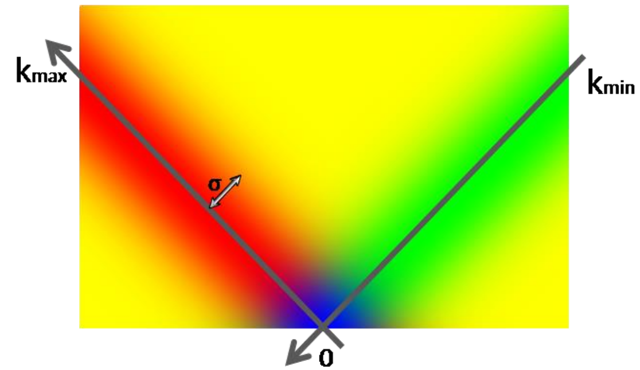
$$_{ax}))$$

if $l_i = 1$

if $l_i = 2$

if $l_i = 3$

if $l_i = 4$

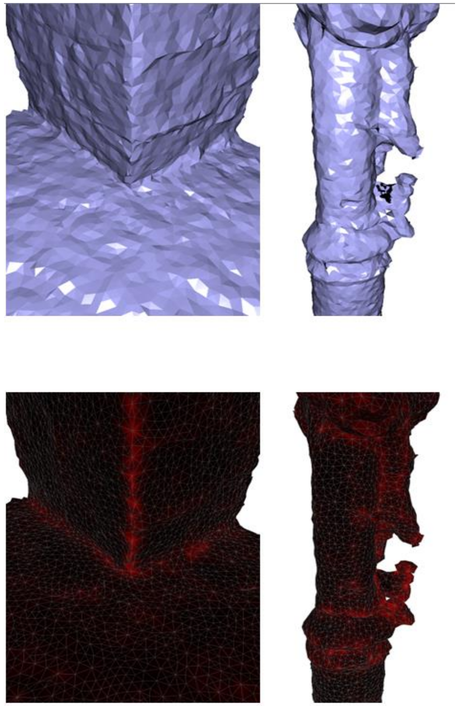


$$G_\sigma(k) =$$

Soft constraints

Label smoothness

Edge preservation



$$V_{ij}(l_i, l_j) = \begin{cases} 1 & \text{if } l_i \neq l_j \\ \min(1, a||\mathbf{W_i} - \mathbf{W_j}||_2) & \text{otherwise} \end{cases}$$

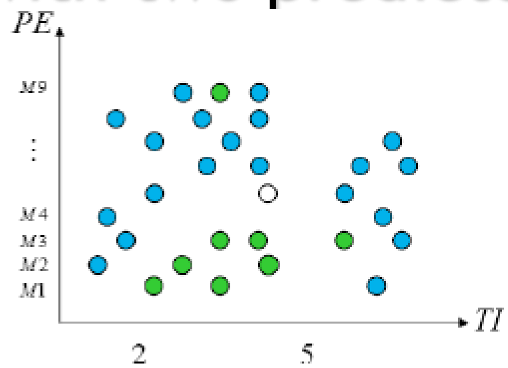
with $\mathbf{W} = \begin{pmatrix} k_{min}.\mathbf{w_{min}} \\ k_{max}.\mathbf{w_{max}} \end{pmatrix}$

Classification by Machine learning (Random Forest)

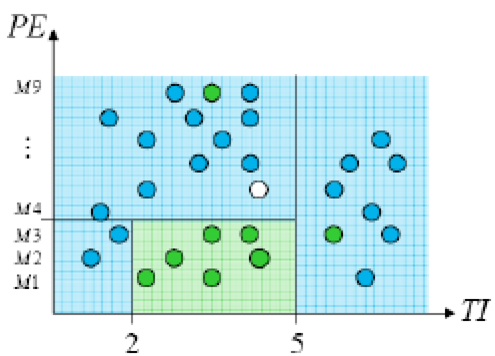
Decision tress involve greedy, recursive partitioning

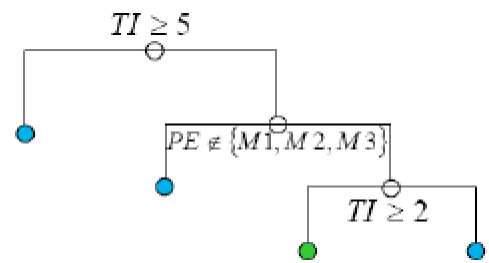
- Simple dataset with two predictors

TI	*PE*	*Response*
1.0	*M2*	good
2.0	*M1*	bad
...	...	...
4.5	*M5*	?

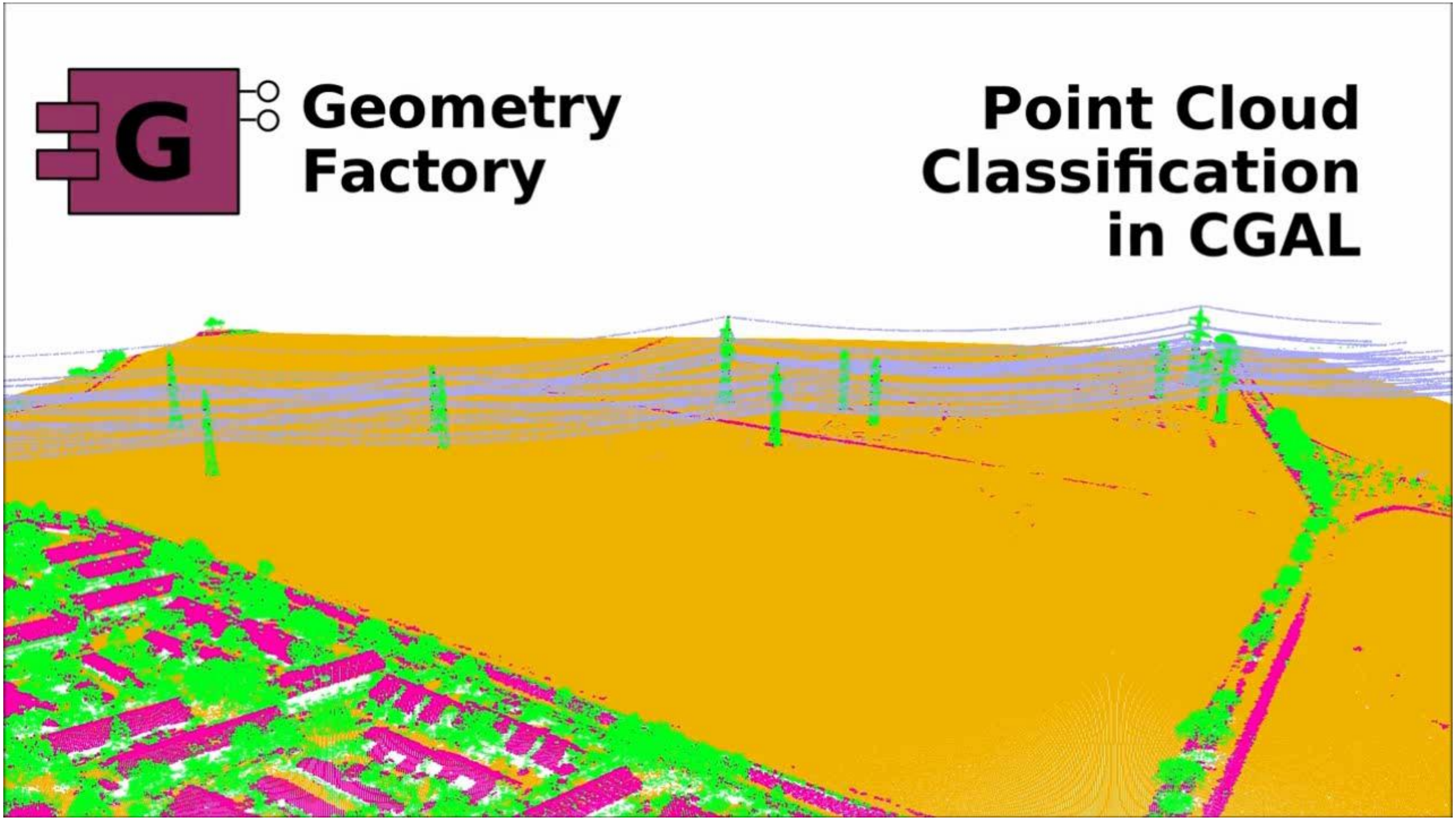


- Greedy, recursive partitioning along TI and PE





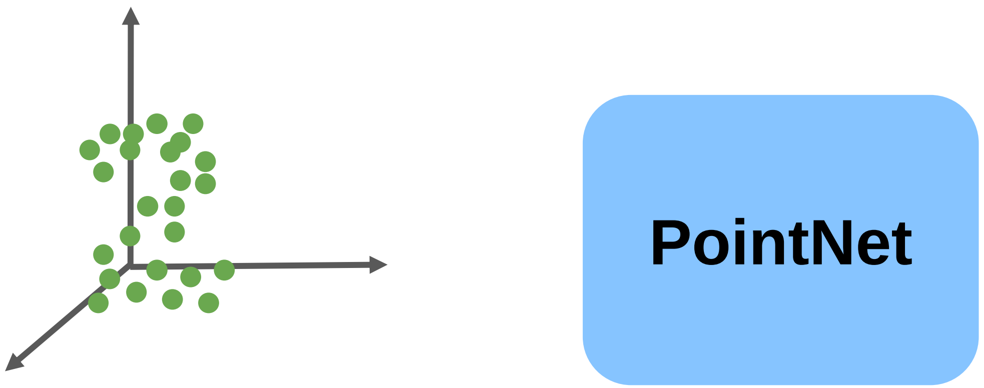
Example with cgal



Classification by Deep learning (PointNet)

PointNet

End-to-end learning for irregular point data

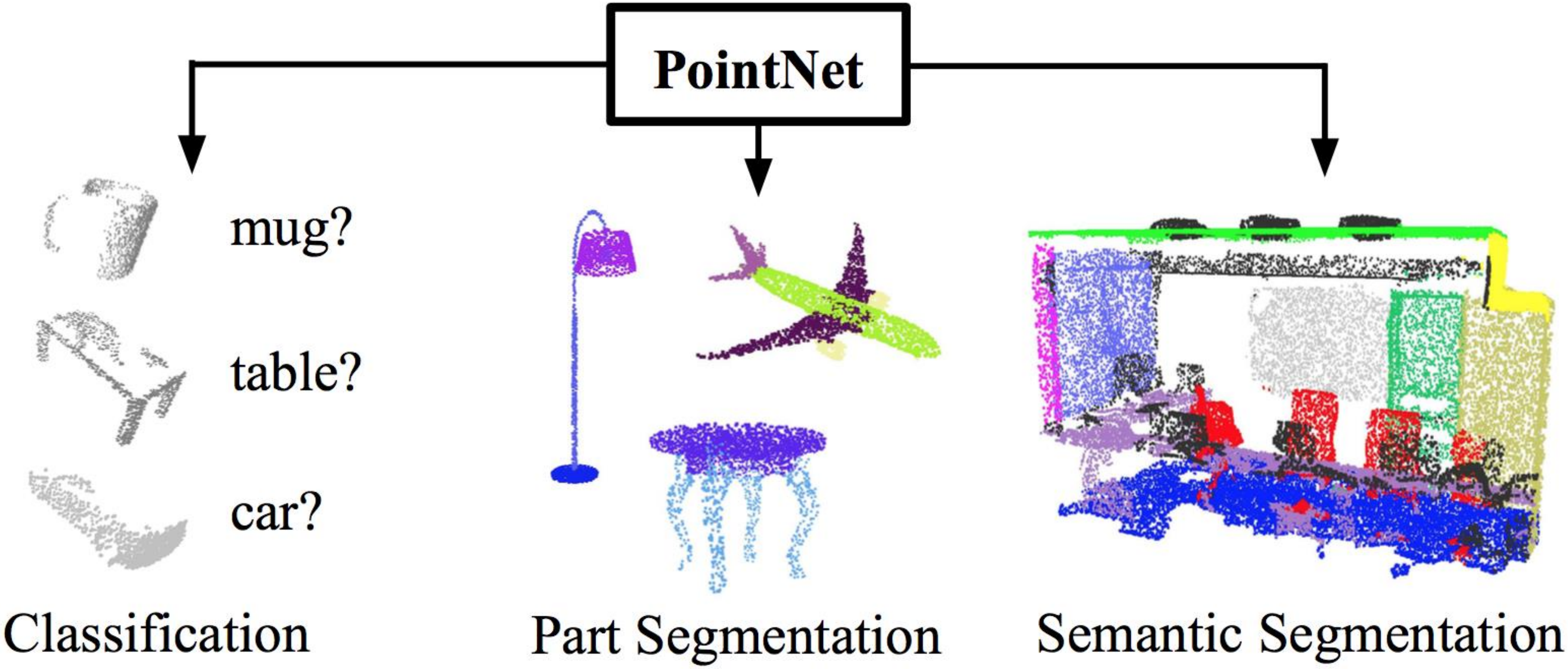


Charles R. Qi, Hao Su, Kaichun Mo, Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. (CVPR'17)

PointNet

End-to-end learning for irregular point data

Unified framework for various tasks



PointNet: challenges

The model has to respect key properties of point clouds:

Point Permutation Invariance

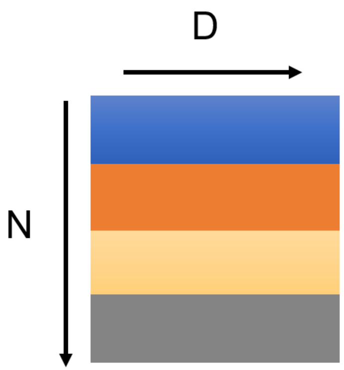
Point cloud is a set of unordered points

Spatial Transformation Invariance

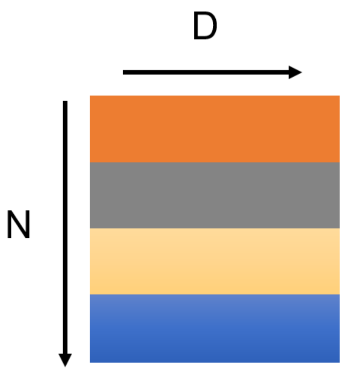
Point cloud rigid motions should not alter classification results

First property: point permutation invariance

Point cloud: set of N **unordered** points, each represented by a D dim vector



represents the same **set** as



Model needs to be invariant to N! permutations

First property: point permutation invariance

$$f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D$$

Examples:

$$f(x_1, x_2, \ldots, x_n) = \max\{x_1, x_2, \ldots, x_n\}$$

$$f(x_1, x_2, \ldots, x_n) = x_1 + x_2 + \ldots + x_n$$

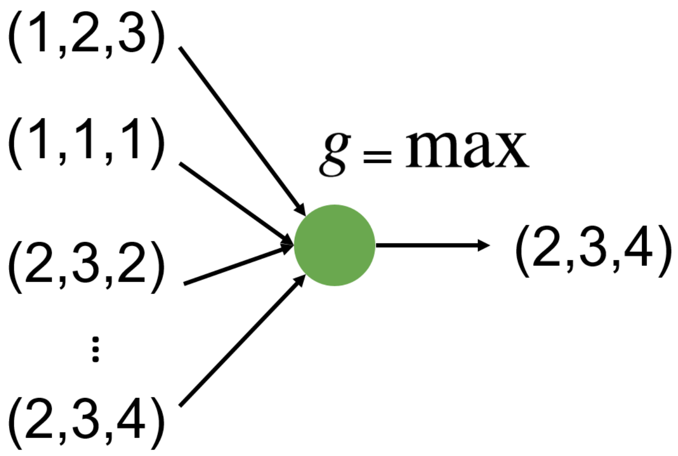
…

How can we construct a universal family of symmetric functions by neural networks?

First property: point permutation invariance

Simplest form: directly aggregate all points with a symmetric operator g

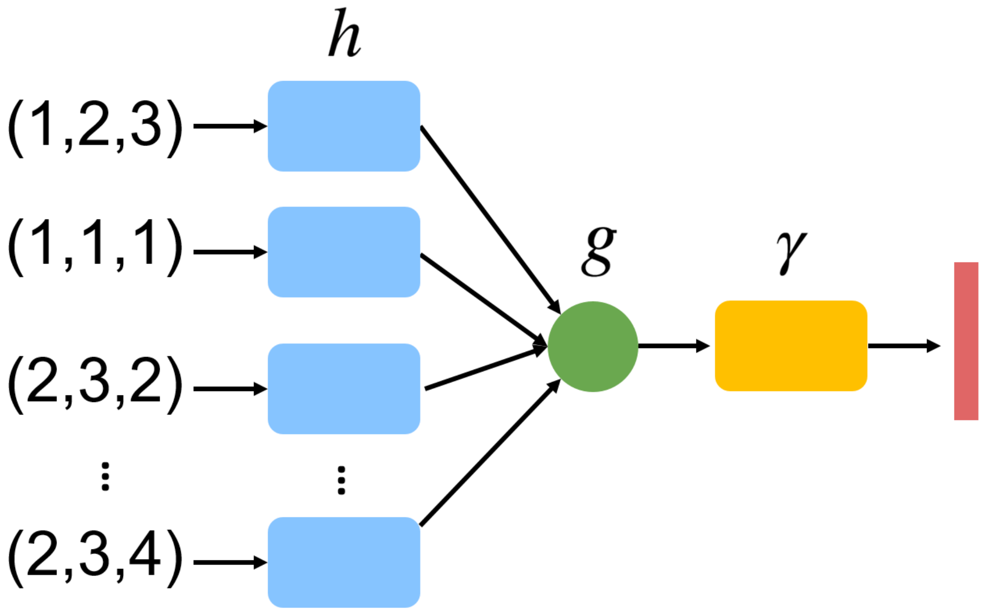
Just discovers simple extreme/aggregate properties of the geometry



First property: point permutation invariance

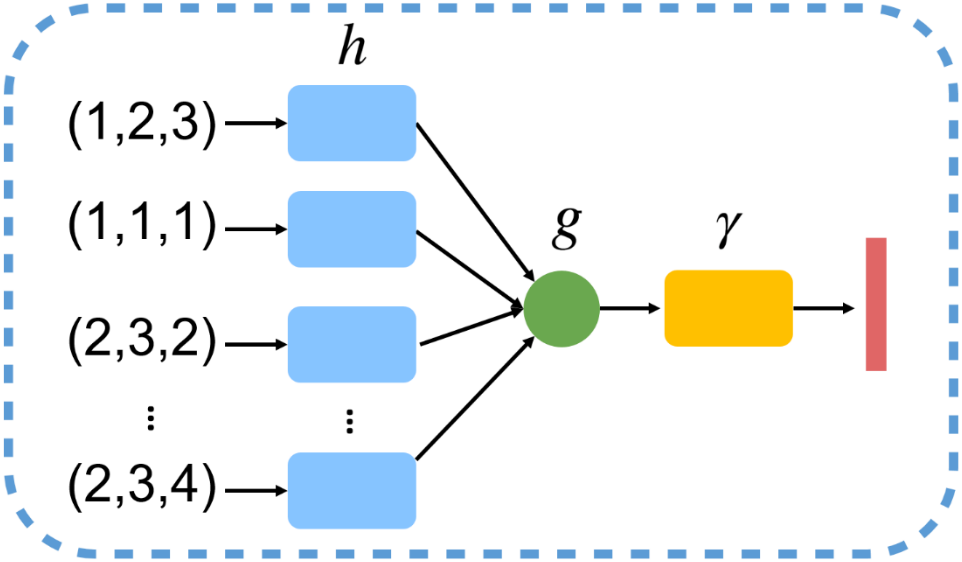
Embed points to a high-dim space before aggregation.

Aggregation in the (redundant) high-dim space encodes more interesting properties of the geometry.



First property: point permutation invariance

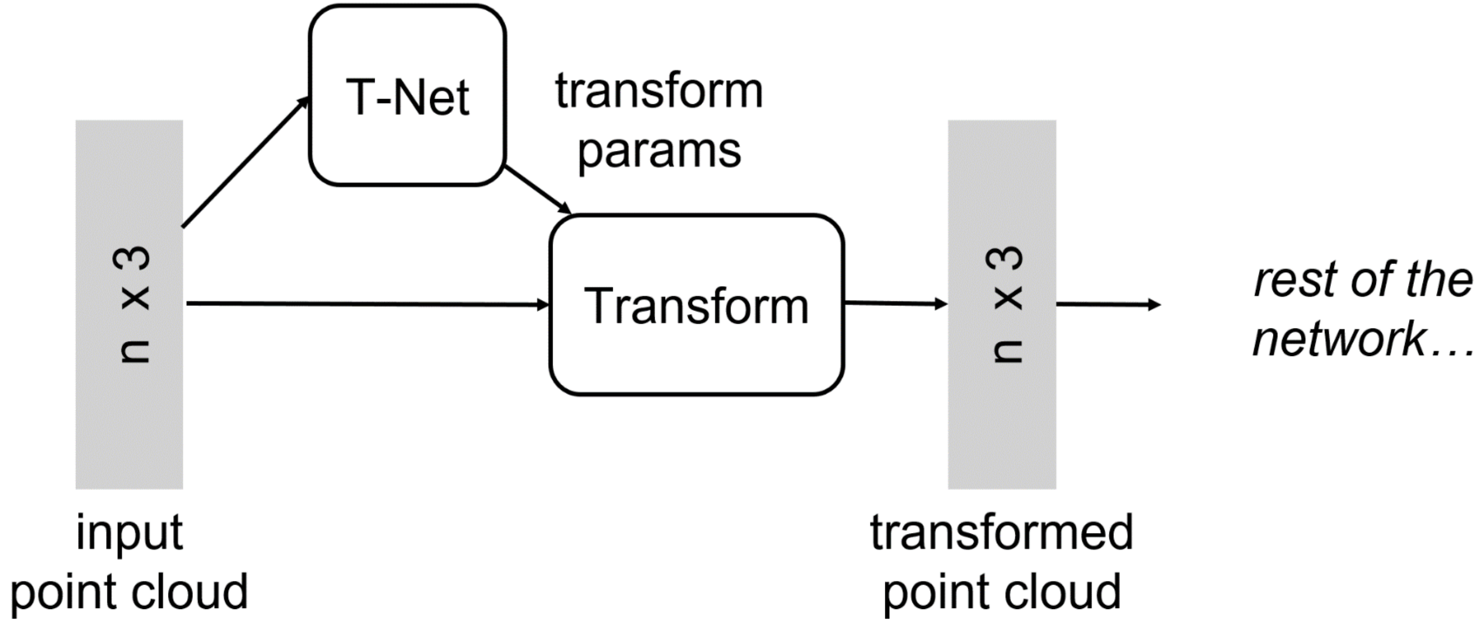
$f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n))$ is symmetric if g is symmetric



PointNet (vanilla)

Second property: spatial transformation invariance

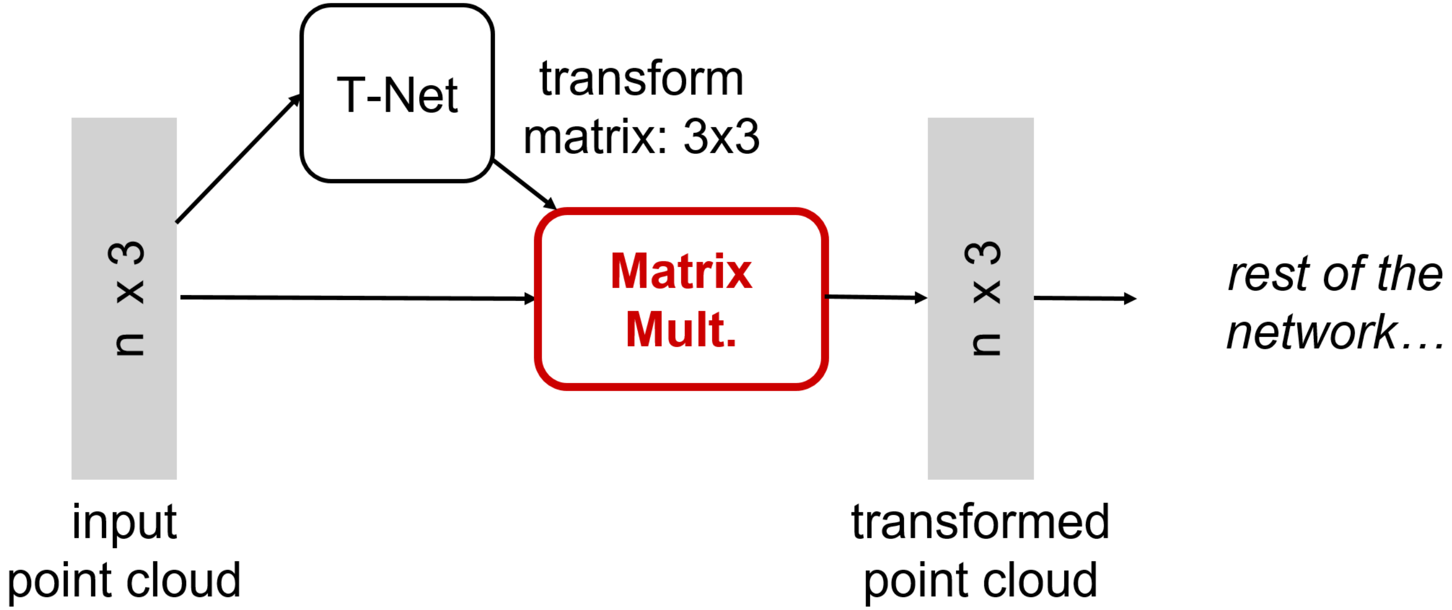
Idea: Data dependent transformation for automatic alignment



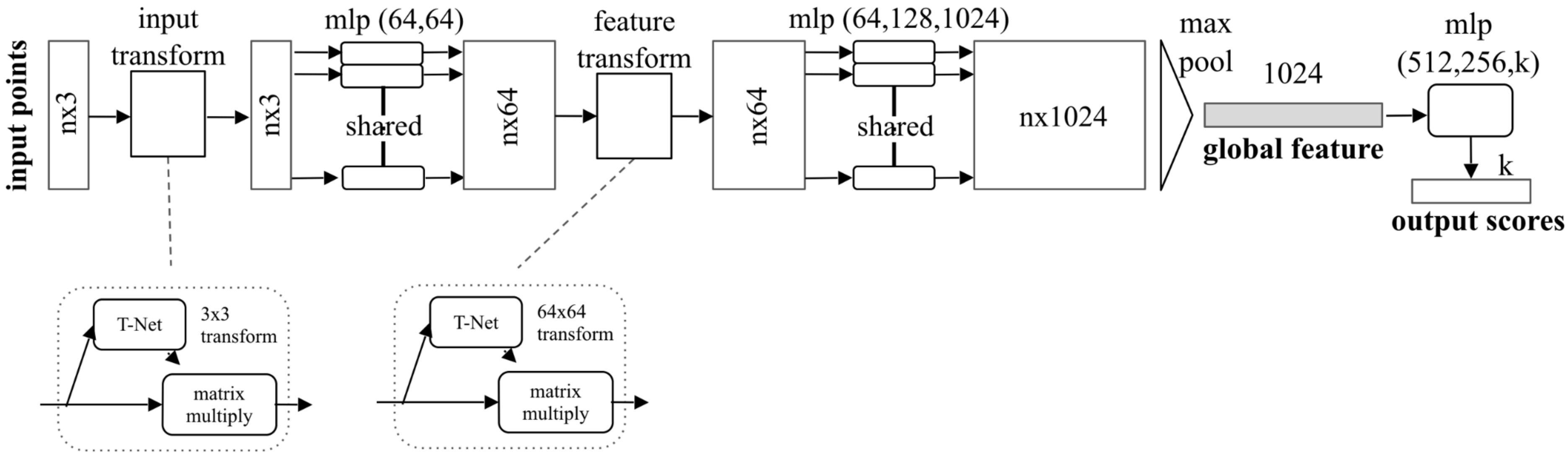
Second property: spatial transformation invariance

Idea: Data dependent transformation for automatic alignment

The transformation is just matrix multiplication!



PointNet architecture for classification tasks

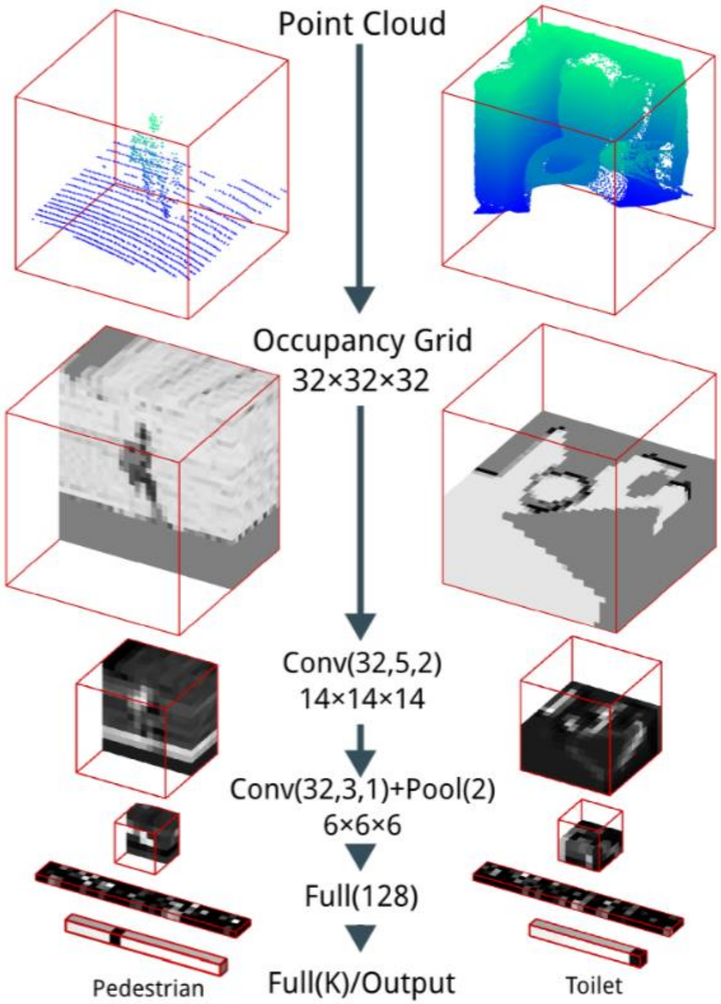


Results on indoor scene classification

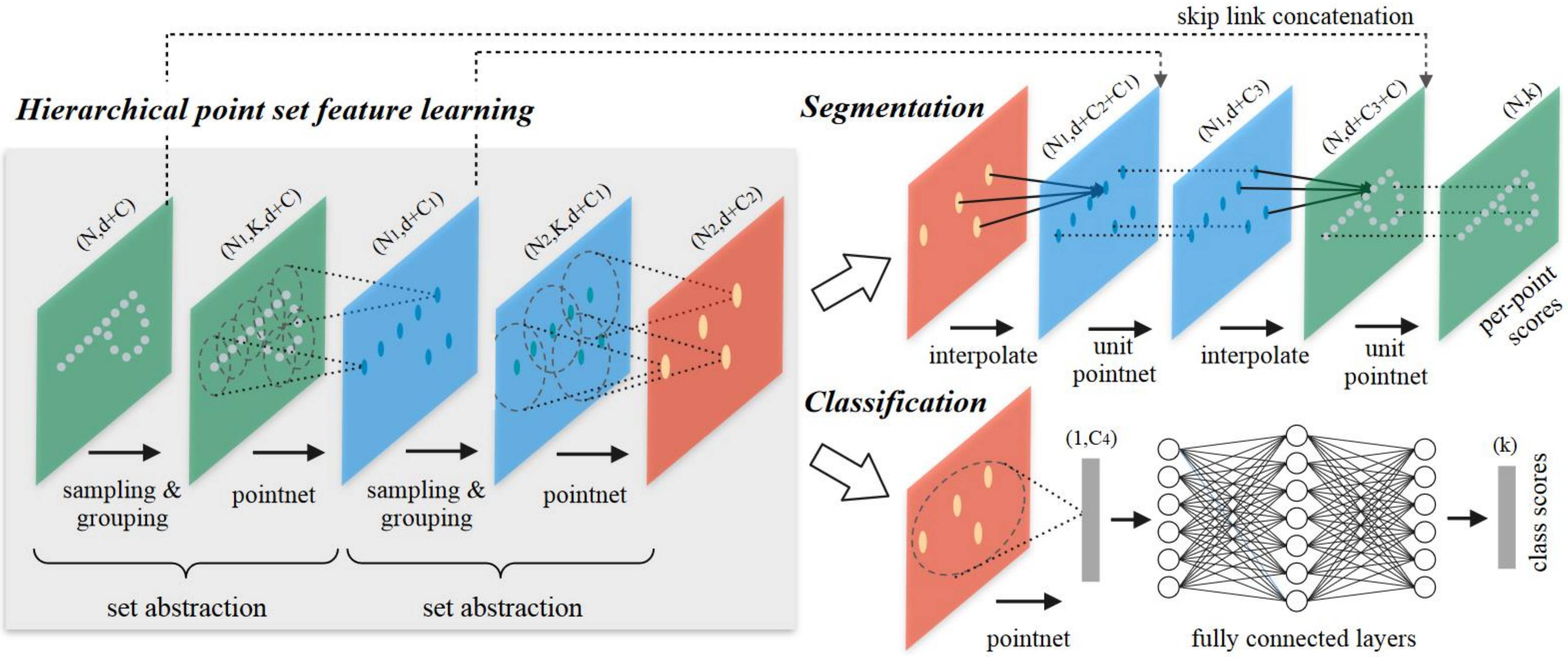
Other deep learning architectures for point cloud classification

- VoxNet (2015)
- PointNet++ (2017)
- Superpoint graph (2018)
- DG-CNN (2019)
- Point Cloud Transformer (2021)
- …

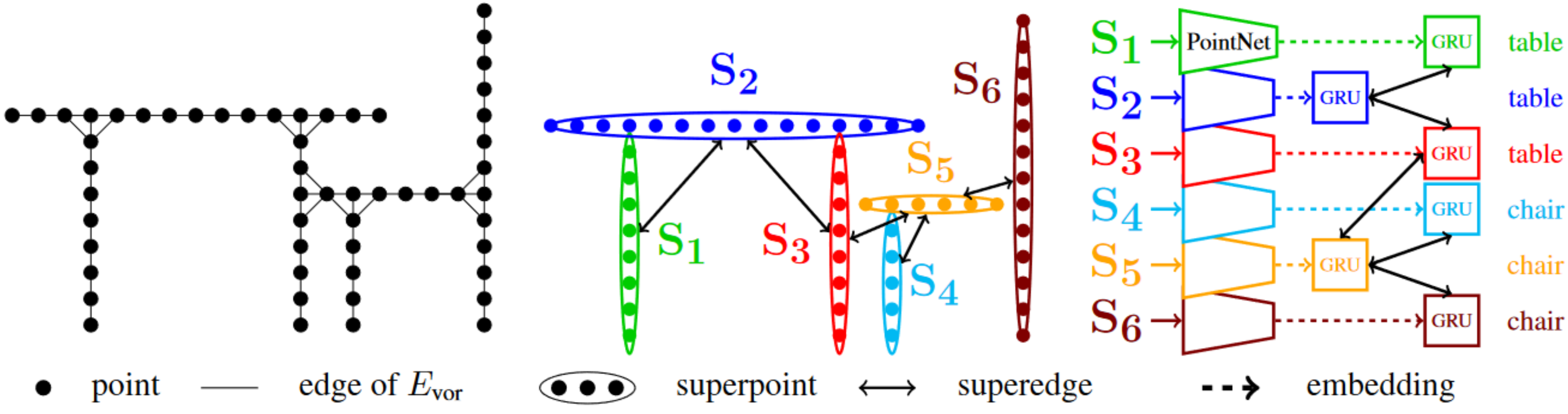
VoxNet (Maturana et al, VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition. In IROS, 2015)



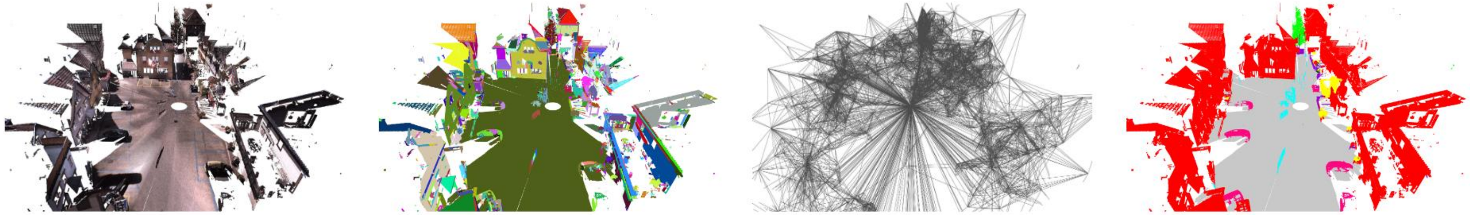
PointNet++ (Qi et al, PointNet++: Deep hierarchical feature learning on point sets in a metric space. In NIPS, 2017)



Superpoint graph (Landrieu et al, Large-scale point cloud semantic segmentation with superpoint graphs. In CVPR, 2018)



(a) Input point cloud (b) Superpoint graph (c) Network architecture


(a) RGB point cloud (b) Geometric partition (c) Superpoint graph (d) Semantic segmentation

DG-CNN (Wang et al., Dynamic graph cnn for learning on point clouds. TOG, 2019)