AI for 3D

Géraldine Morin

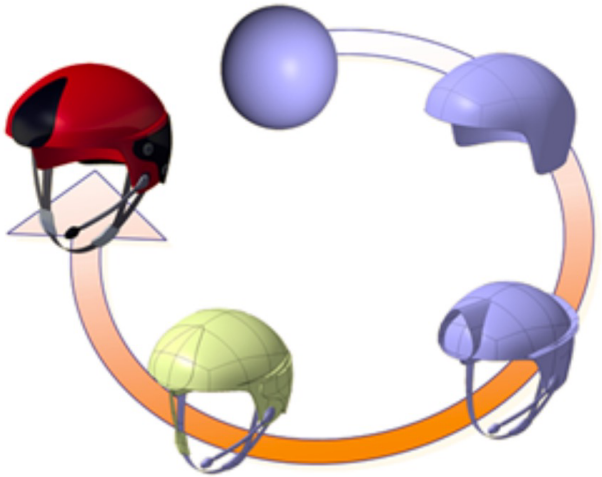
Are deep learning techniques also for 3D ?

- Deep learning for images, video, text ? For what:

What is 3D data ?

Create the model (artists, engineers)

- CAD modeling



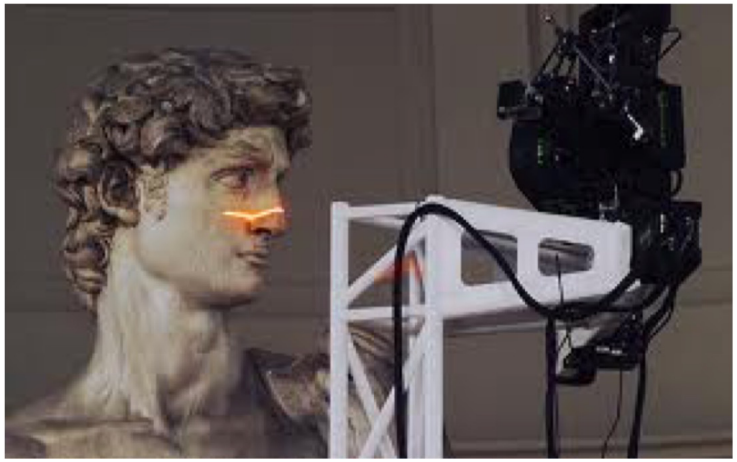
Imagine and Shape: *Dassault Systems* modeler based on subdivision surfaces (and NURBS)

- Sketching

[Kim et al. 3D Agile sketching…2017]

Capture the object (1/2)

- With a scanner



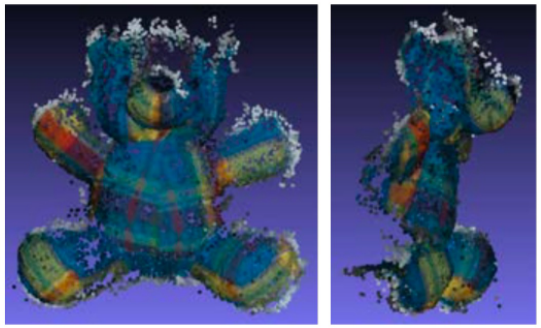
Marc Levoy, Michaelangelo Project

- With a LIDAR

[Urban Capture, Pazos-Perez, 2017]

Capture the object (2/2)

- With images (MVS=Multiview stereo)
- SLAM



Shape from motion: reconstruction à partir de 12 images (AliceVisio)

How do we represent the 3D models ?

- How to represent 3D with a mathematical model ?
- How do we code them ?

Different 3D models

- Modeling 3D objects
  - CAD models (60's) : parametric
  - Meshes
  - Implicit representations

More and more 3D data Applications

- Capture

  Creating existing model

  - Reconstruction from images

  Mapping environnement in 3D offline :

  - architecture
  - cultural heritage preservation
  - 3D maps

  - LiDAR

More and more 3D data Applications

- Capture

  Creating existing model

  - Reconstruction from images

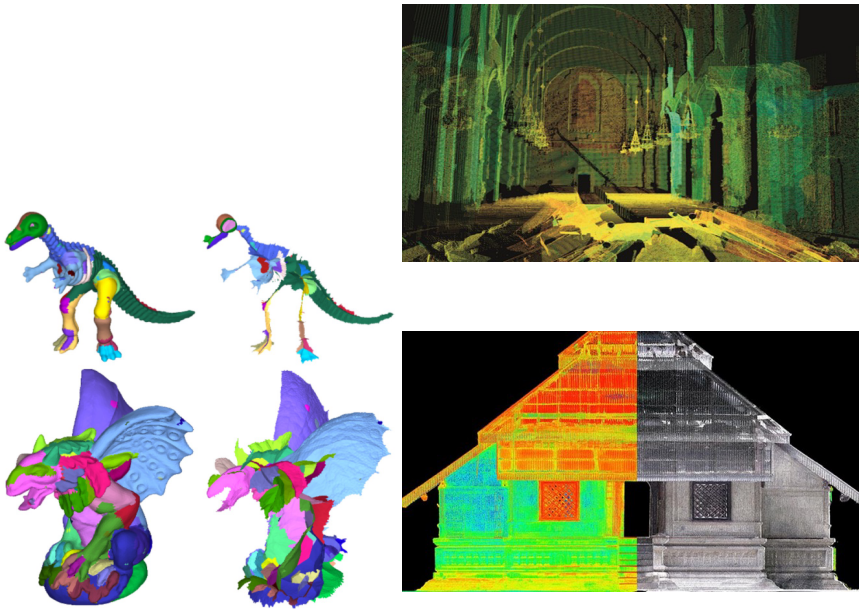
  Mapping environnement in 3D **offline**

  - LiDAR

  Online navigation :

  - navigating with a drone
  - autonomous driving

More and more 3D data Applications

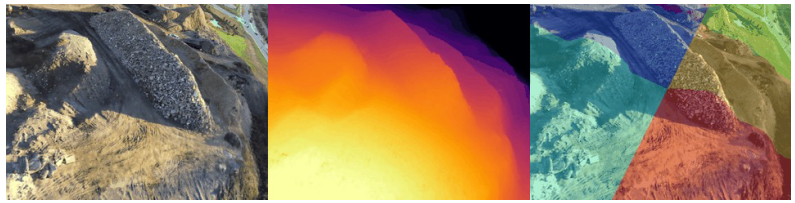


Need to understand,
to analyze,
to interpret

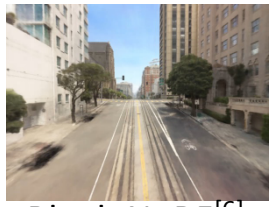
[Images source: Nuage De Points, Lida scan, url: https://www.pinterest.fr/pin/389913280214566434/. Digitally preserving Hukuru Miskiy, url: https://leica-geosystems.com/fr-fr/case-studies/reality-capture/digitally-preserving-hukuru-miskiy

More and more 3D data Applications

Google streetmap



Mega-NeRF[5]



Block-NeRF[6]

Need to interact with
Navigate into
a 3D environment

[5] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. Srinivasan, J. T. Barron, and H. Kretzschmar. ***Block-NeRF: Scalable large scene neural view synthesis***. CVPR, 2022.
[6] H. Turki, D. Ramanan, and M. Satyanarayanan. ***Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs***. CVPR, June 2022.

Part 1 : analyzing 3D content

More and more 3D data Datatype

- Capture

  Creating existing model

  - Reconstruction from images

  Mapping environnement in 3D **offline**

  - LiDAR

  Online navigation :

  - navigating with a drone
  - autonomous driving

3D point clouds

3D + t

Need to understand the content interpret

LARGE DATASETS

Classical analysis methods for 3D content

- Local features = around a point
  - Variance (3 directions)
  - Order 1 = planarity ? Normal ?
  - Order 2 = curvature ? Mean curvature ? K_min K_max ?

  $\Rightarrow$ Flat point, crease, angle

- Global features ? Recognize a car, a pedestrian ?

  $\Rightarrow$ Get the semantic information out of the data !

Why not « also » use deep learning ?

- Deep learning is well developed for images
  - Generalize from 2D to 3D = 3D Voxels
  - Use 2D + depth

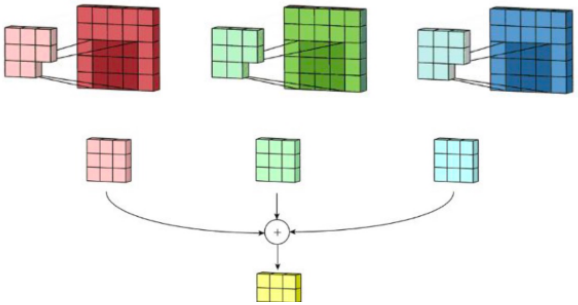
- « Real 3D » : point clouds

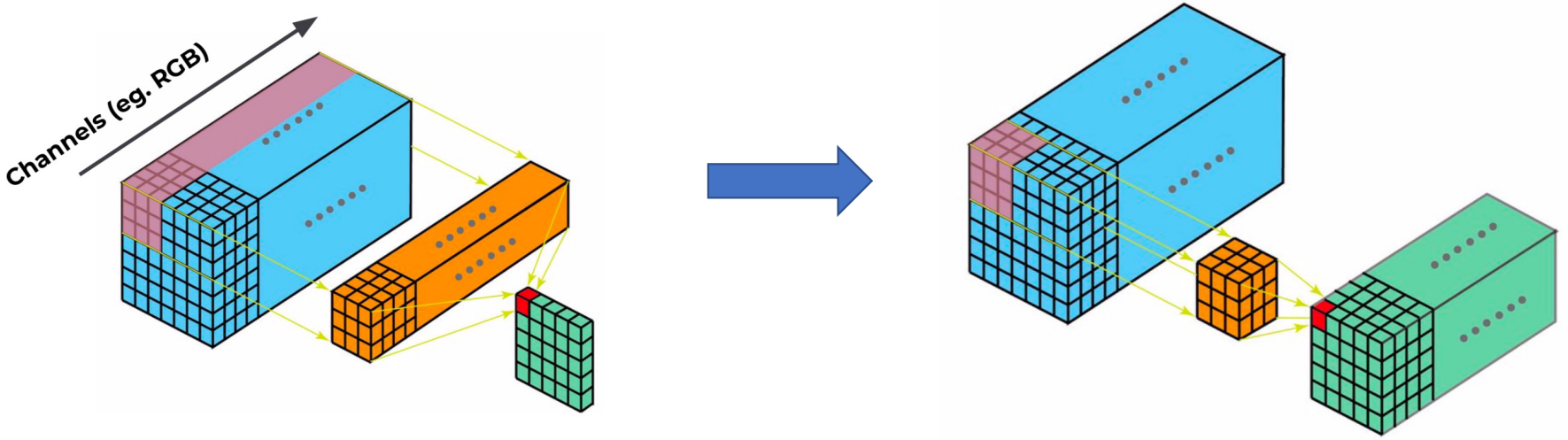
First Approach = 3D Voxels

- Generalize what

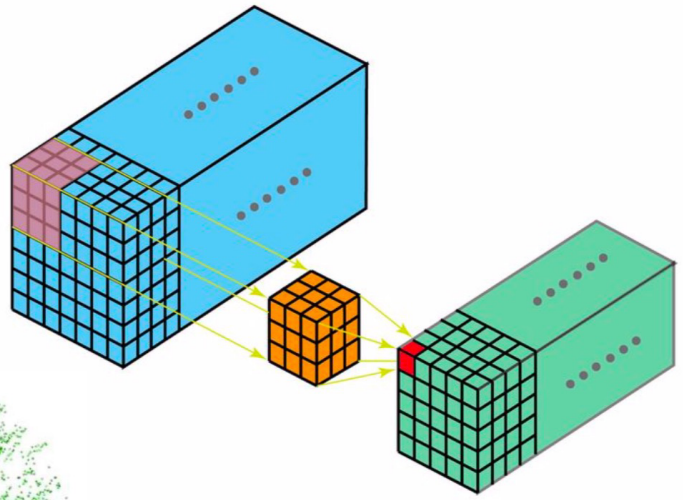
is done for 2D data (images)

- RGB-D (+multiview) 🤢

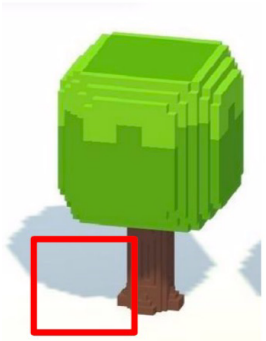




First Approach = 3D Voxels



+ benefits from experience for CNN for images



- Very sparse representation
- Very complex

Image O(n^2)

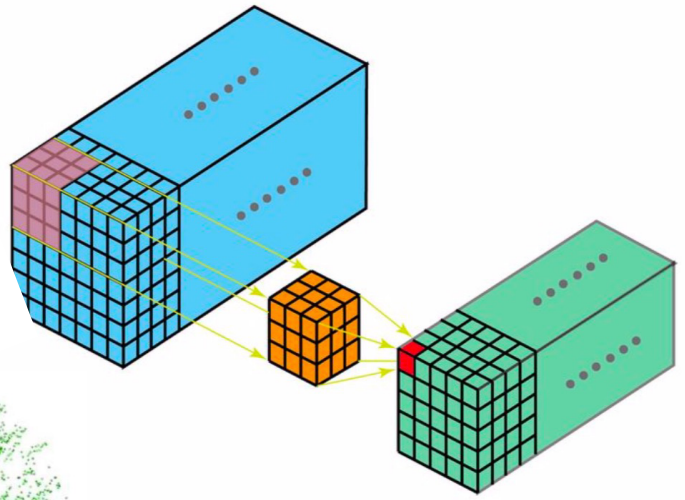
Voxel O(n^3) !!!

- Not the natural structure of 3D content !

ShapeNet [3] (2015), VoxNet [4] (2015), ORION [5] (2016) ou encore OctNet [6] (2016)

[3] Zhirong Wu et al., « 3D ShapeNets: A deep representation for volumetric shapes », 2015.
[4] D. Maturana et S. Scherer, « VoxNet: A 3D Convolutional Neural Network for real-time object recognition », 2015.
[5] N. Sedaghat, M. Zolfaghari, E. Amiri, et T. Brox, « Orientation-boosted Voxel Nets for 3D Object Recognition », 2016
[6] G. Riegler, A. O. Ulusoy, et A. Geiger, « OctNet: Learning Deep 3D Representations at High Resolutions », 2016.

First Approach = 3D Voxels



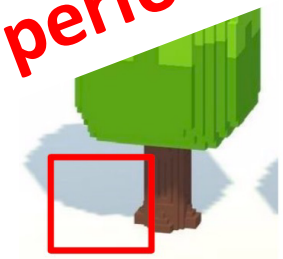
+ benefits from experience for C

- Very sparse

- Ve

\ (n^3) !!!

TROP CHER : Compromis précision / performance !!!



- Not the natural structure of 3D content !

ShapeNet [3] (2015), VoxNet [4] (2015), ORION [5] (2016) ou encore OctNet [6] (2016)

[3] Zhirong Wu et al., « 3D ShapeNets: A deep representation for volumetric shapes », 2015.

[4] D. Maturana et S. Scherer, « VoxNet: A 3D Convolutional Neural Network for real-time object recognition », 2015.

[5] N. Sedaghat, M. Zolfaghari, E. Amiri, et T. Brox, « Orientation-boosted Voxel Nets for 3D Object Recognition », 2016

[6] G. Riegler, A. O. Ulusoy, et A. Geiger, « OctNet: Learning Deep 3D Representations at High Resolutions », 2016.

Second approach : 2D + depth

- For LIDAR (light/laser detection and ranging)

LiDAR enregistre

- x nappes (ex: x=64) –ou #lignes
- une résolution angulaire y (ex: y=0.175 degrés); soit 2048 pixels par lignes

Ainsi, il y a théoriquement x × y points reconstruits en 3D

certains ne rencontrent pas d'obstacle, ce qui fait des *miss* (environ 8%)

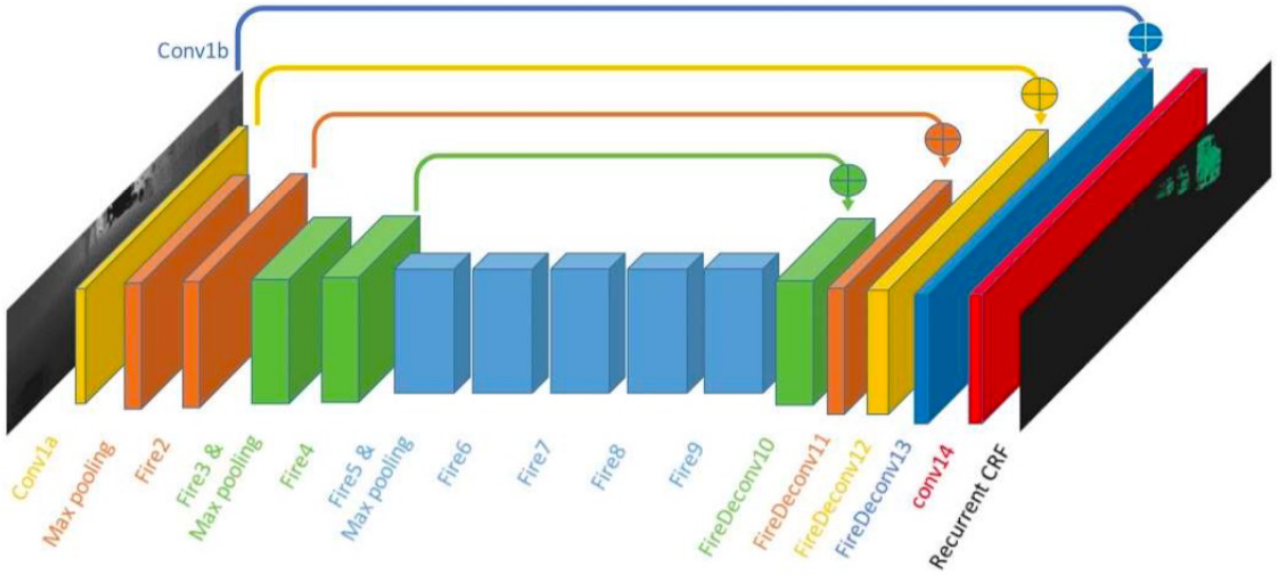
On peut garder ces données sur une image de x × y pixels et travailler sur ces images avec des réseaux de neurones

- Problème de données manquantes
- Résolutions spatiale image != resolution 3D ET inhomogène

Second approach : 2D + depth

réseaux de neurones Encoder / decoder

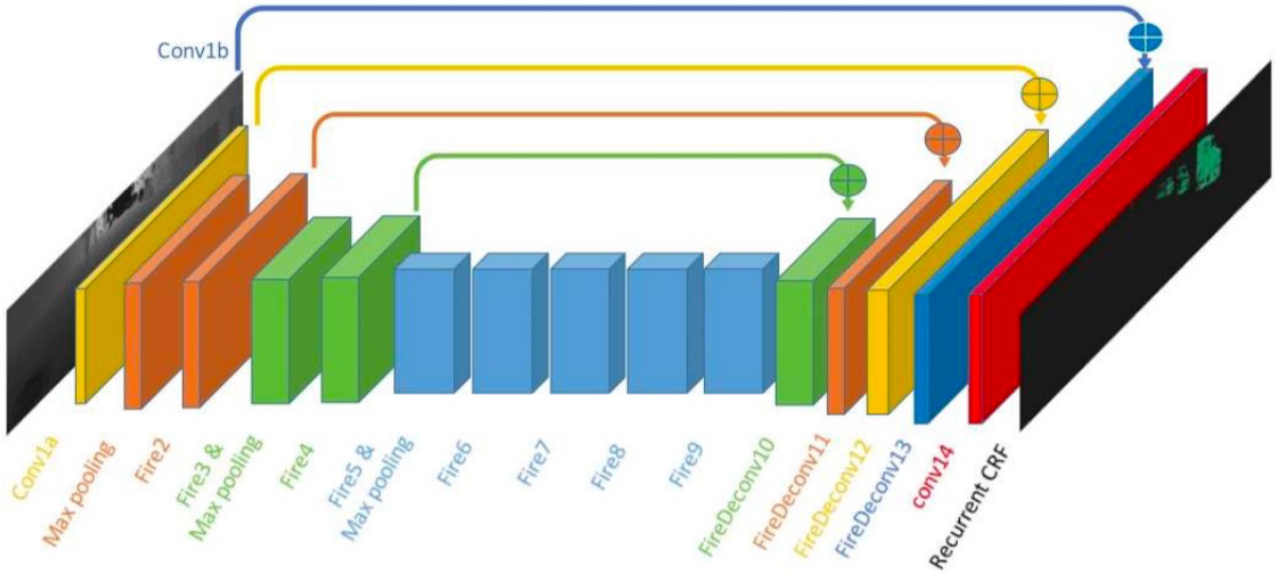
- SqueezeNet [9] (2016) : classification
- SqueezeSeg [10] (2017) : segmentation



Second approach : 2D + depth

réseaux de neurones Encoder / decoder

- SqueezeNet [9] (2016) : classification
- SqueezeSeg [10] (2017) : segmentation



Second approach : 2D + depth

- SqueezeSegV2 [11] (2018) : robuste aux points manquants
- SqueezeSegV3 (Avril 2020) : + adaptation des variabilité de résolution / Spatially-Adaptative Convolution (SAC).

Alternative : also encoder – decoder, also robust to misses and heterogeneous resolution

- SalsaNet [15] (2019)
- SalsaNext [16] (juillet 2020)

[10] Bichen Wu, Alvin Wan, Xiangyu Yue et Kurt Keutzer, “SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud”, octobre 2017.

[11] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue et Kurt Keutzer, “SqueezeSegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud”, septembre 2018.

[12] Chenfeng Xu, Bichen Wu, Zining Wang, Wei Zhan, Peter Vajda, Kurt Keutzer et Masayoshi Tomizuka, “SqueezeSegV3: Spatially-Adaptive Convolution for Efficient Point-Cloud Segmentation”, avril 2020.

[15] Eren Erdal Aksoy, Saimir Baci et Selcuk Cavdar, “SalsaNet: Fast Road and Vehicle Segmentation in LiDAR Point Clouds for Autonomous Driving”, Septembre 2019.

[16] Tiago Cortinhal, George Tzelepis et Eren Erdal Aksoy, “SalsaNext: Fast, Uncertainty-aware Semantic of LiDAR Point Clouds for Autonomous Driving”, juillet 2020.

Second approach : 2D + depth

- SqueezeSegV2 [11] (2018) : robuste aux [illegible] uants
- SqueezeSegV3 (Avril 2020) : + ad[illegible]é de résolution / Spatially-Adaptative Convolution (SA[illegible]

Alternative [illegible] robust to misses and heterogeneous resolution

- Salsa[illegible]
- SalsaN[illegible]2020)

2D + depth : Dépendance au point de vue

[10] Bichen Wu, Alvin Wan, Xiangyu Yue et Kurt Keutzer, “SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud”, octobre 2017.

[11] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue et Kurt Keutzer, “SqueezeSegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud”, septembre 2018.

[12] Chenfeng Xu, Bichen Wu, Zining Wang, Wei Zhan, Peter Vajda, Kurt Keutzer et Masayoshi Tomizuka, “SqueezeSegV3: Spatially-Adaptive Convolution for Efficient Point-Cloud Segmentation”, avril 2020.

[15] Eren Erdal Aksoy, Saimir Baci et Selcuk Cavdar, “SalsaNet: Fast Road and Vehicle Segmentation in LiDAR Point Clouds for Autonomous Driving”, Septembre 2019.

[16] Tiago Cortinhal, George Tzelepis et Eren Erdal Aksoy, “SalsaNext: Fast, Uncertainty-aware Semantic of LiDAR Point Clouds for Autonomous Driving”, juillet 2020.

Proposition : work on point sets (clouds)

- Original paper proposing point sets : DeepSets [1]
- PointNet [2]
- PointNet++ [3]

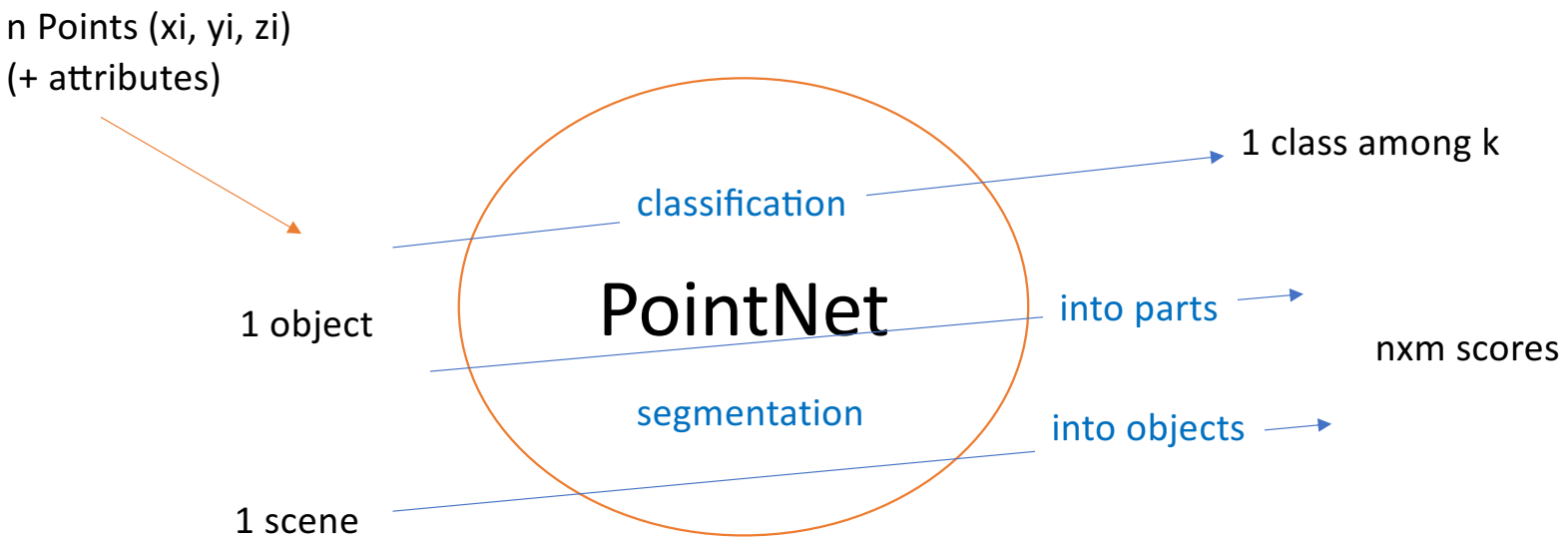
[1] Deep Sets, Zaheer et al. 2018

[2] PointNet : Deep Learning on Point Sets Calssifications and Segmentation, QI et al. 2017

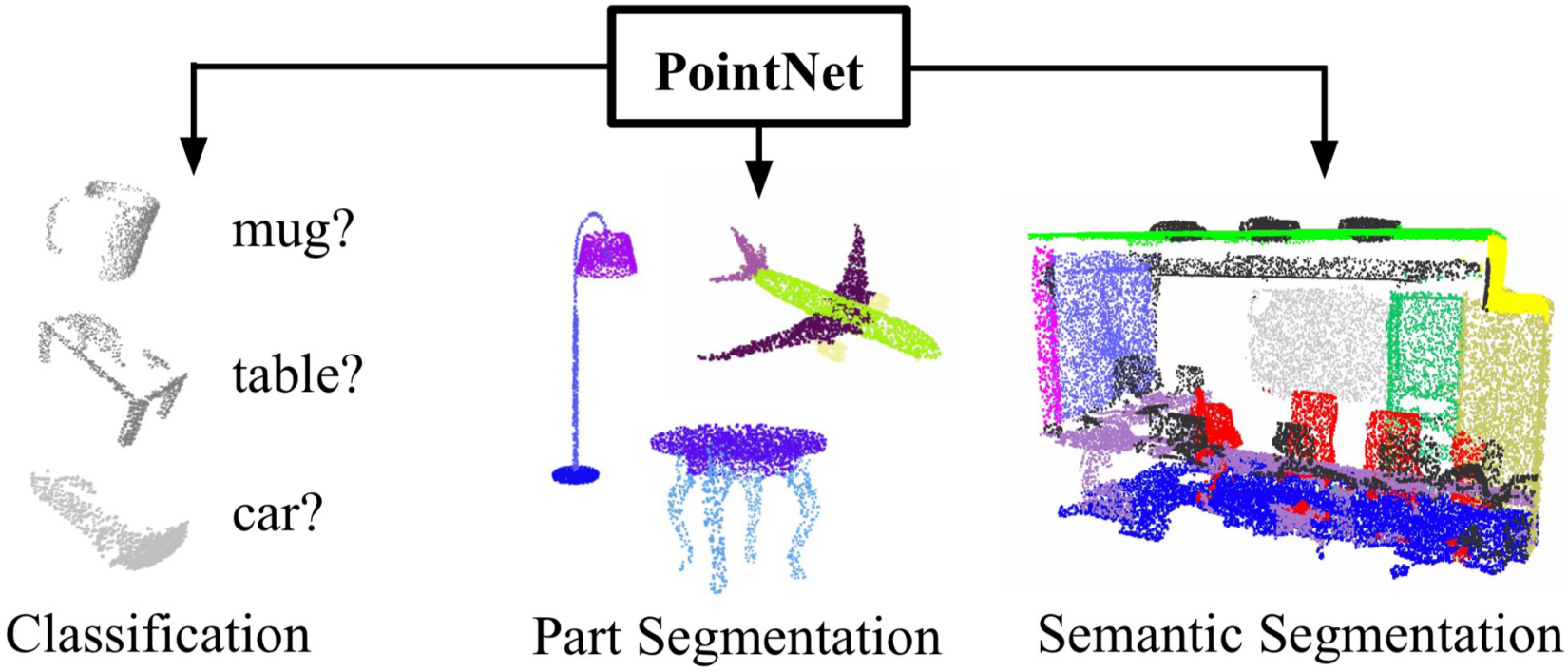
[3] PointNet++ : Deep Hierarchical Feature Learning on Point Sets in a Metric Space, Qi et al. 2017

PointNet

- Propose a NN capable of



PointNet


PointNet
mug?
table?
car?
Classification
Part Segmentation
Semantic Segmentation

Point Sets

To have a symmetric f

$$f(\{x1,x2, \dots xn\}) \approx g(h(x1), \dots , h(xn))$$

h : MLP (multilayer perceptron network) : $\mathbb{R}^N \rightarrow \mathbb{R}^K$

It is sufficient to have a symmetric g : $\mathbb{R}^K$ x …x $\mathbb{R}^K \rightarrow \mathbb{R}$

Point Sets

$$f(\{x1,x2, \dots xn\}) \approx g(h(x1), \dots , h(xn))$$

h : MLP (multilayer perceptron network) : $\mathbb{R}^N \rightarrow \mathbb{R}^K$

g symmetric : $\mathbb{R}^K$ x …x $\mathbb{R}^K \rightarrow \mathbb{R}$

g composition of a max pool and a symmetric function

Ex : g (x1,x2, …, xn) = x1 + x2 + … + xn

g (x1,x2, …, xn) = max(x1,x2, … ,xn)

Point Sets : Approximation Theorem (Cours A. Carlier)

Théorème d'Approximation Universelle

Théorème d'approximation universelle (Cybenko 1989)

Toute fonction f, continue, de $[0,1]^m$ dans $\mathbb{R}$, peut être approximée par un perceptron multi-couche à une couche cachée comportant suffisamment de neurones (avec une fonction d'activation sigmoide).

Note : le théorème a été également prouvé avec la fonction reLU.

Le théorème **ne dit pas comment** déterminer ce réseau de neurones !

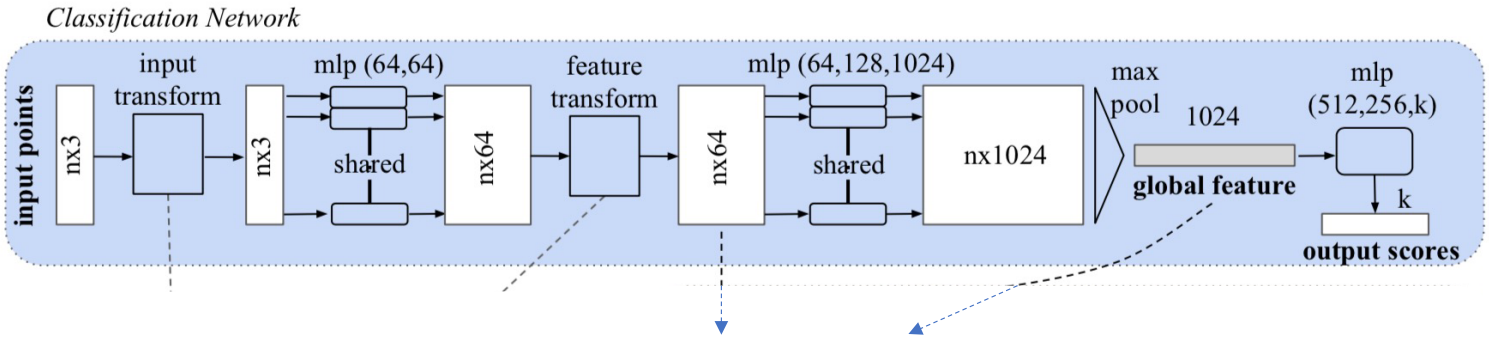
Point Sets : Approximation Theorem for sets

Theorem 1. *Suppose* $f : \mathcal{X} \to \mathbb{R}$ *is a continuous set function w.r.t Hausdorff distance* $d_H(\cdot, \cdot)$. $\forall \epsilon > 0$, $\exists$ *a continuous function* h *and a symmetric function* $g(x_1, \ldots, x_n) = \gamma \circ MAX$, *such that for any* $S \in \mathcal{X}$,

$$\left| f(S) - \gamma \left(\underset{x_i \in S}{MAX} \{h(x_i)\} \right) \right| < \epsilon$$

where $x_1, \ldots, x_n$ *is the full list of elements in* S *ordered arbitrarily,* γ *is a continuous function, and MAX is a vector max operator that takes* n *vectors as input and returns a new vector of the element-wise maximum.*

PointNet

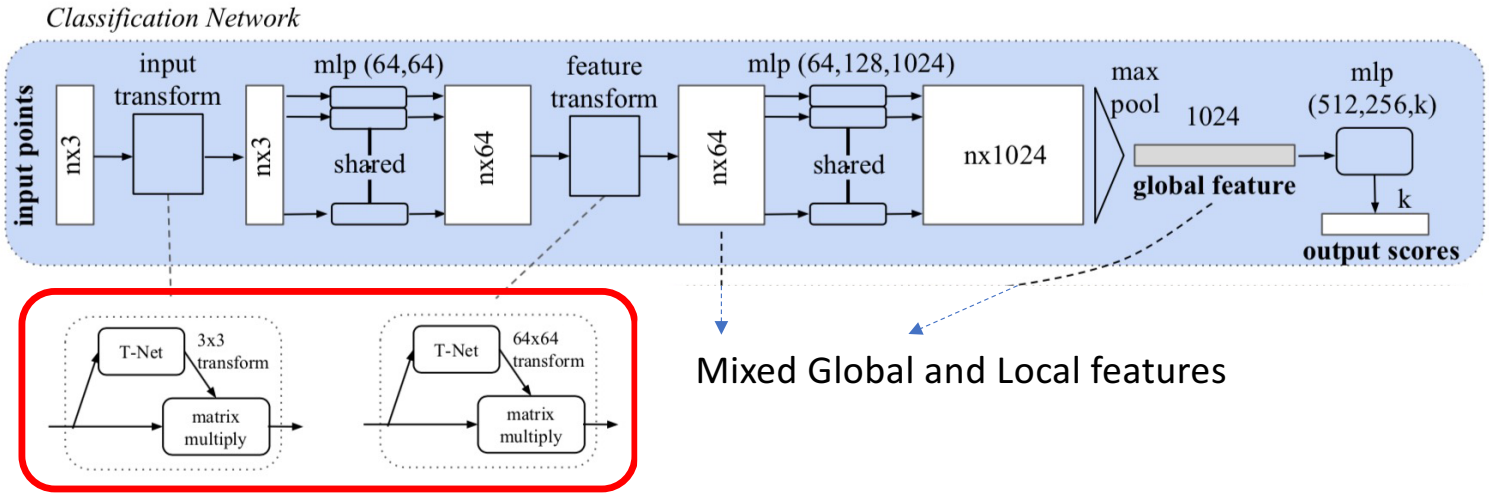


Mixed Global and Local features

Input points :

- a fixed # of pts (1024 uniformly sampled on a model, 4096 random per 1mx1m tile on scenes)

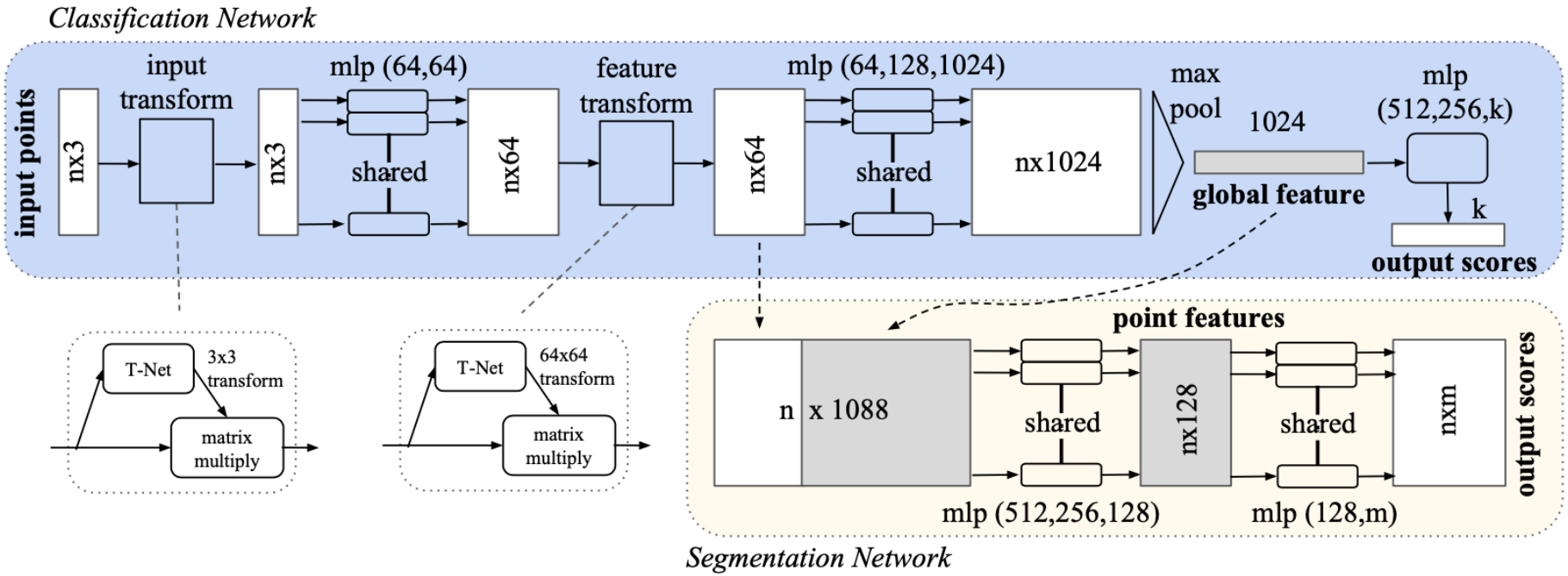
PointNet



Characteristics :

- Invariance to rigid motion : transformation

PointNet


Classification Network
input points
nx3
input transform
nx3
mlp (64,64)
shared
nx64
feature transform
nx64
mlp (64,128,1024)
shared
nx1024
max pool
1024
global feature
mlp (512,256,k)
k
output scores
T-Net
3x3 transform
matrix multiply
T-Net
64x64 transform
matrix multiply
point features
n x 1088
shared
nx128
shared
nxm
output scores
mlp (512,256,128)
mlp (128,m)
Segmentation Network

PointNet : Complexity

Image O(n^2)

Voxel O(n^3)

Complexity is linear in the number of points ! 😀

PointNet : part segmentation results

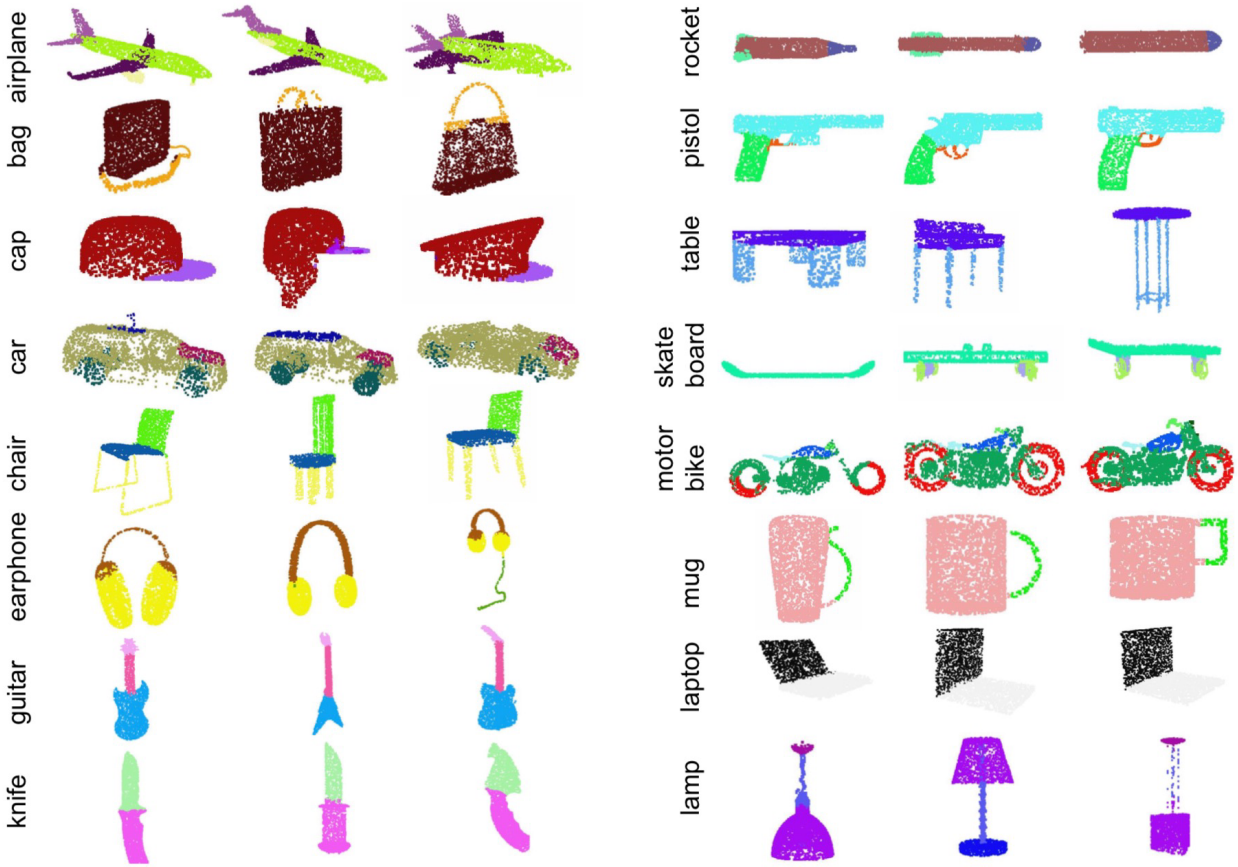


Figure 21. **PointNet segmentation results on complete CAD models.**

PointNet : part segmentation results

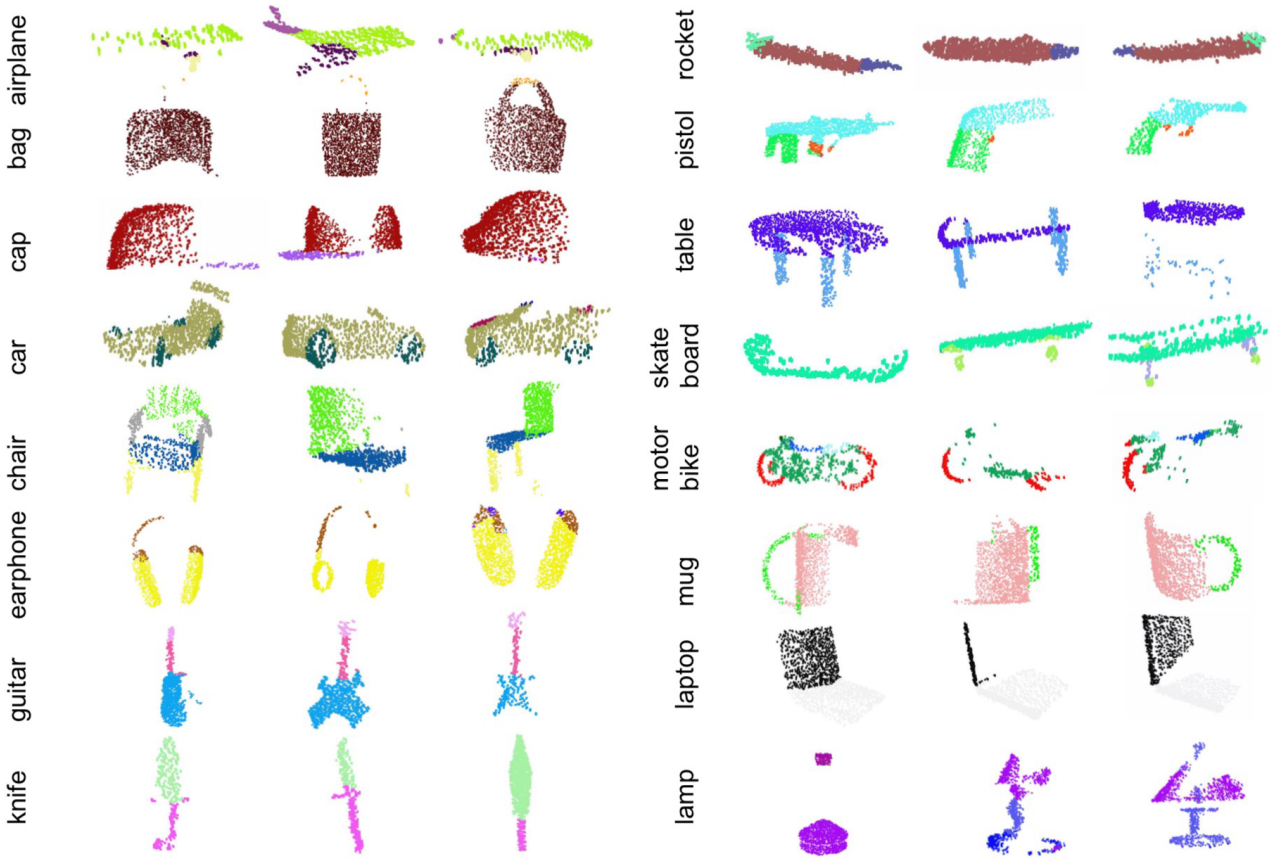
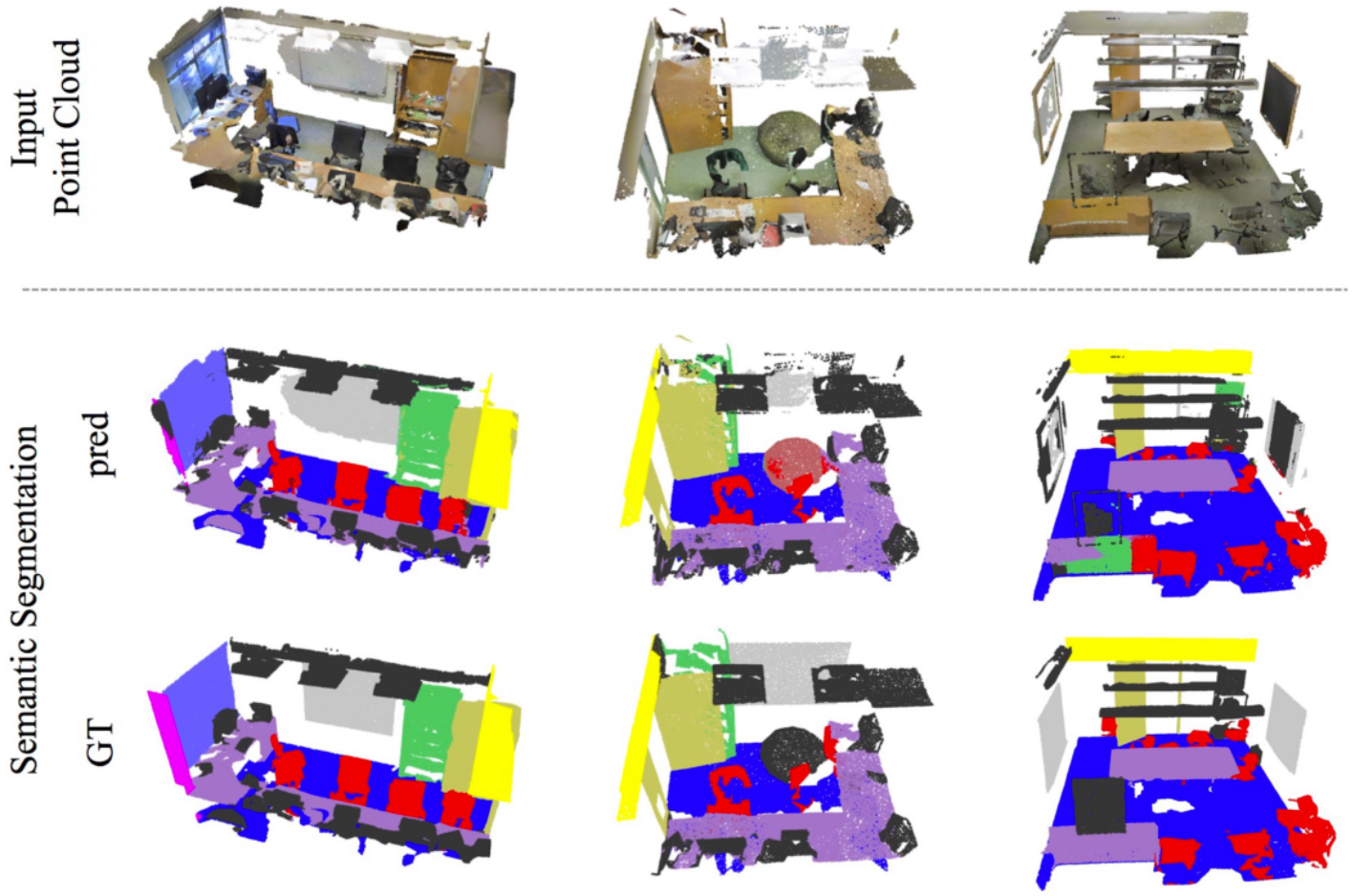


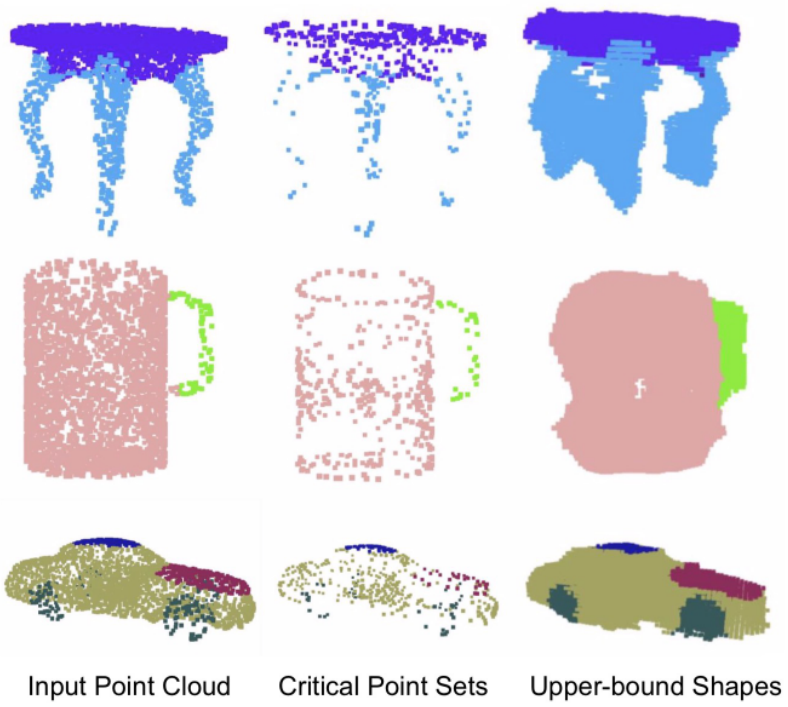
Figure 22. **PointNet segmentation results on simulated Kinect scans.**

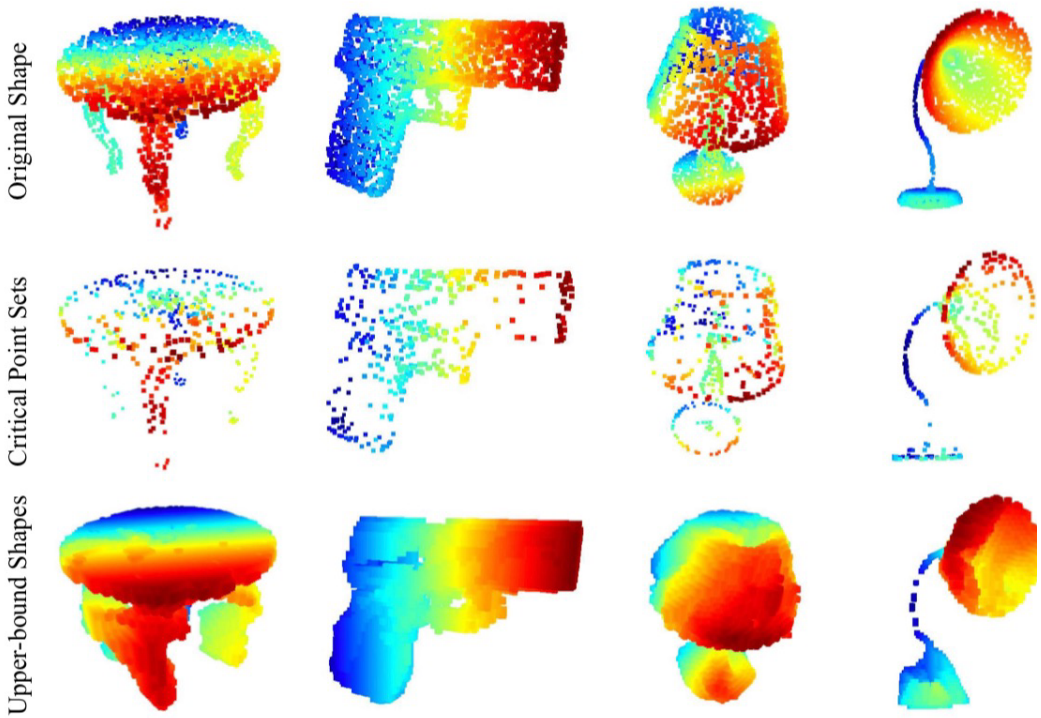
PointNet : semantic segmentation results



PointNet : Critical point Set

- Another result from PointNet is the identification of relevant features learnt by the Network, called critical point set Cs





PointNet : Critical point Set

- Another result from PointNet is the identification of relevant features learnt by the Network, called critical point set Cs

Theorem 2. *Suppose* $\mathbf{u} : \mathcal{X} \rightarrow \mathbb{R}^K$ *such that* $\mathbf{u} = \underset{x_i \in S}{MAX}\{h(x_i)\}$ *and* $f = \gamma \circ \mathbf{u}$. *Then,*

(a) $\forall S, \exists\, \mathcal{C}_S, \mathcal{N}_S \subseteq \mathcal{X},\ f(T) = f(S)\ \text{if}\ \mathcal{C}_S \subseteq T \subseteq \mathcal{N}_S;$

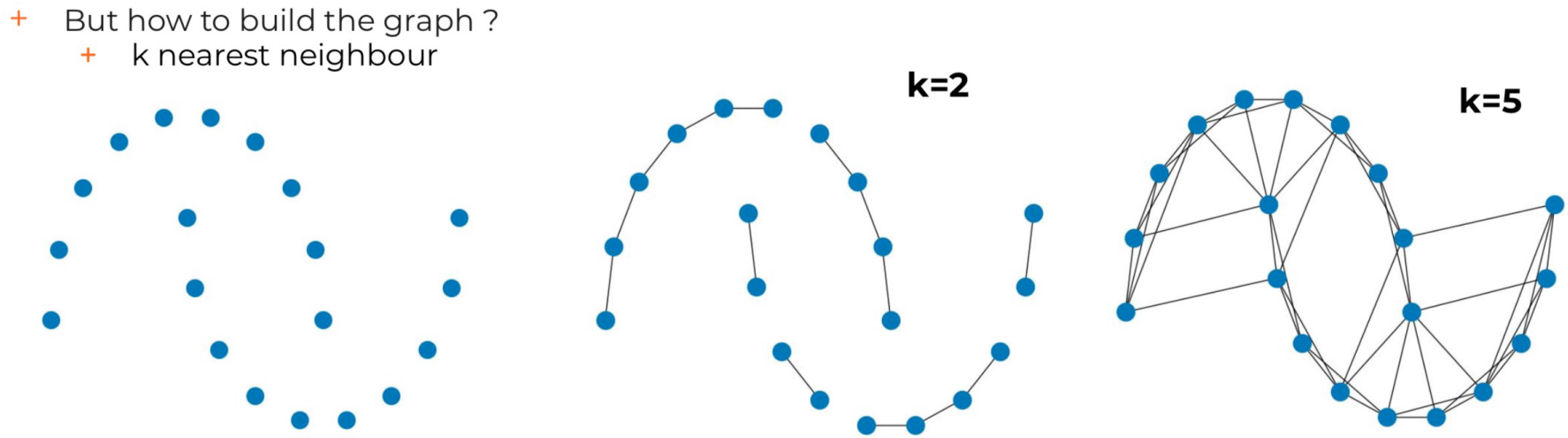
(b) $|\mathcal{C}_S| \leq K$

Further work : to gain locality

- PointNet++ (same authors)-> work at several resolutions / Hierarchical
- PointNet++ is a hierarchical neural network that will recursively apply PointNet to hierarchical partitions of the set of points in order to combine the features highlighted by PointNet at several levels of scale.
- Expensive..;

Other work : Point clouds as a graph

- From irregular point repartition in 3D, find local topology

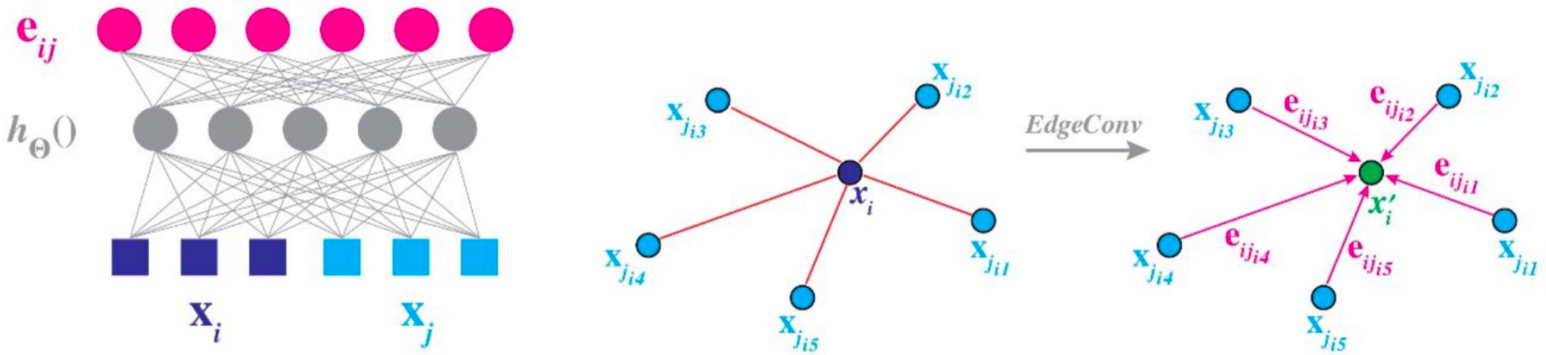


- In practice, find a set of neighbors (distance, orientation)

Point clouds as a graph

Edge convolution :

- m convolutional filters
- Graph structure is dynamic, it changes during the training



$$e_{ijm} = ReLU(\theta_m \cdot (x_j - x_i) + \phi_m \cdot x_i)$$

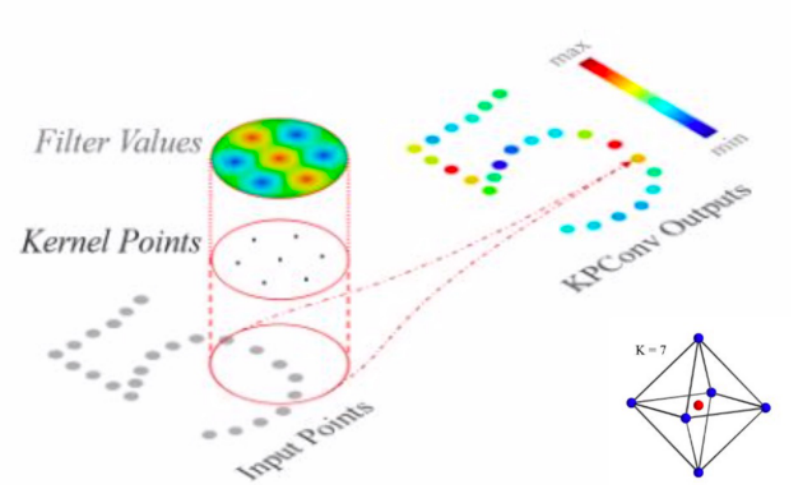
$$x_{im}{}' = max_j(e_{ijm}) \qquad \Theta = [\theta, \phi]$$

Point clouds as a graph : Kernel Point Conv.

Recent work propose convolutions defined for a varying # of points

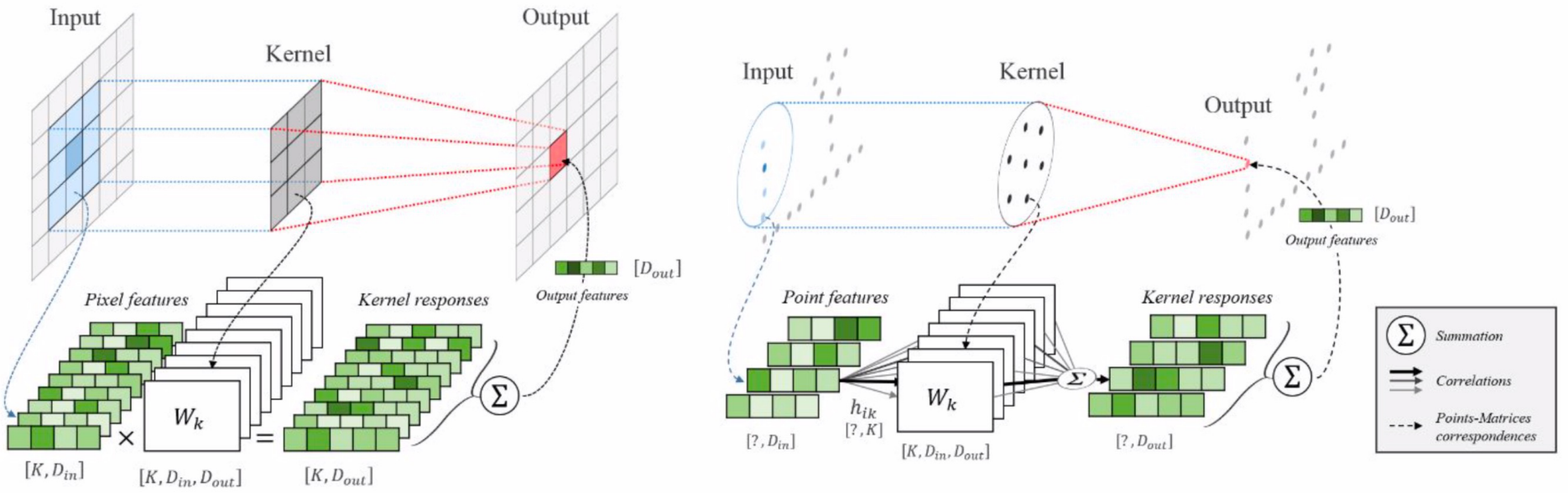
+ no preprocessing on input data
+ topology is differentiable so can be learnt

(french research : cocorico)



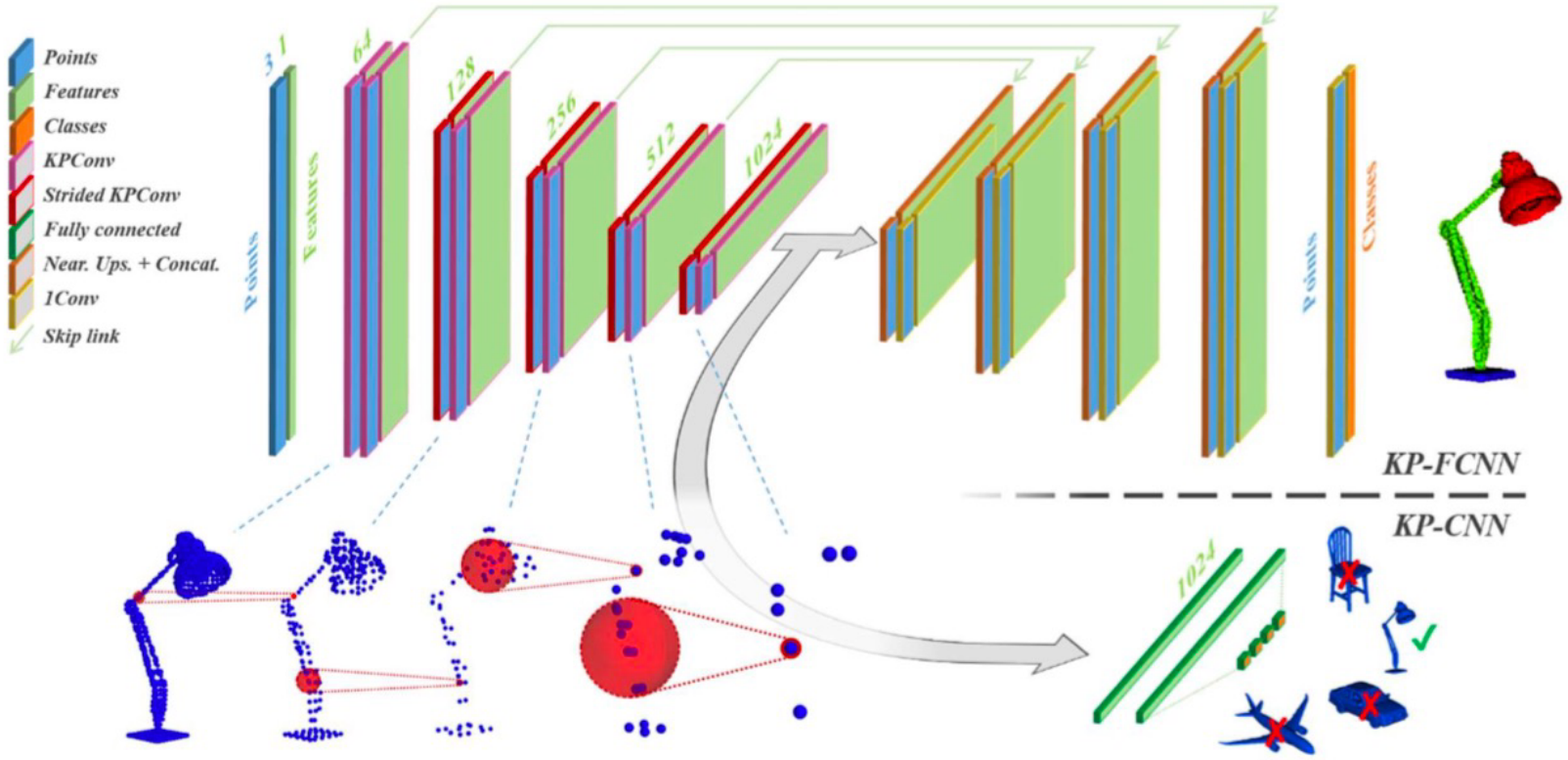
[17] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette et Léonidas J. Guibas, « KPConv: Flexible and Deformable Convolution for Point Clouds ».

Point clouds as a graph : Kernel Point Conv.



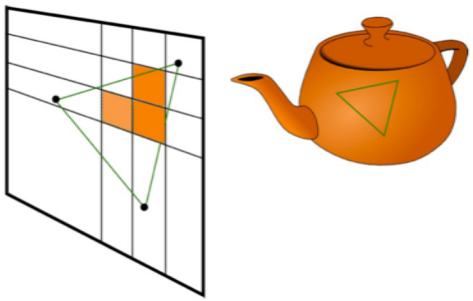
+ outperforms previous 3D segmentation

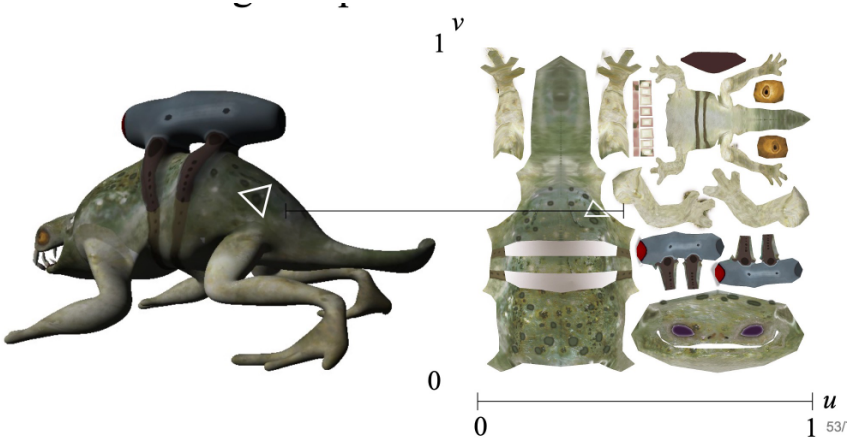
Point clouds as a graph : Kernel Point Conv.



Part 2 : navigating through 3D content

- Explicit representation + classical rendering





Part 2 : navigating through 3D content

Implicit representation + ray tracing

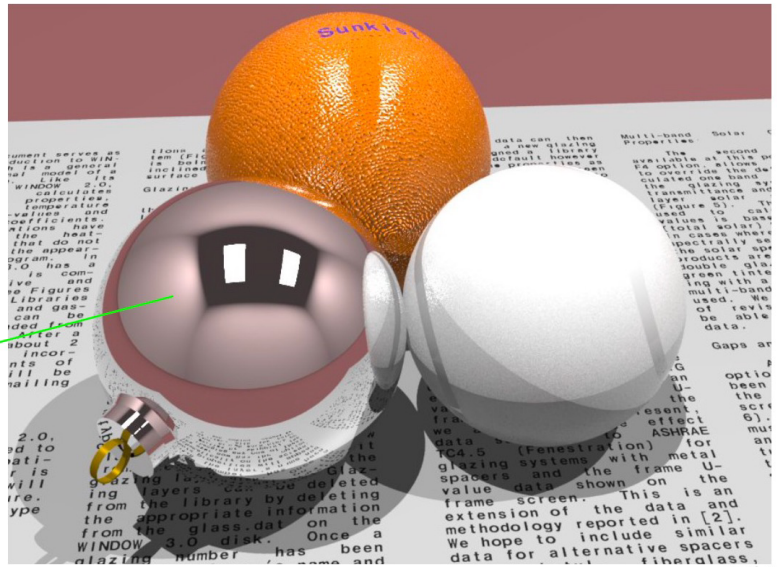


Image taken from http://radsite.lbl.gov/radiance/book/img/plate10.jpg

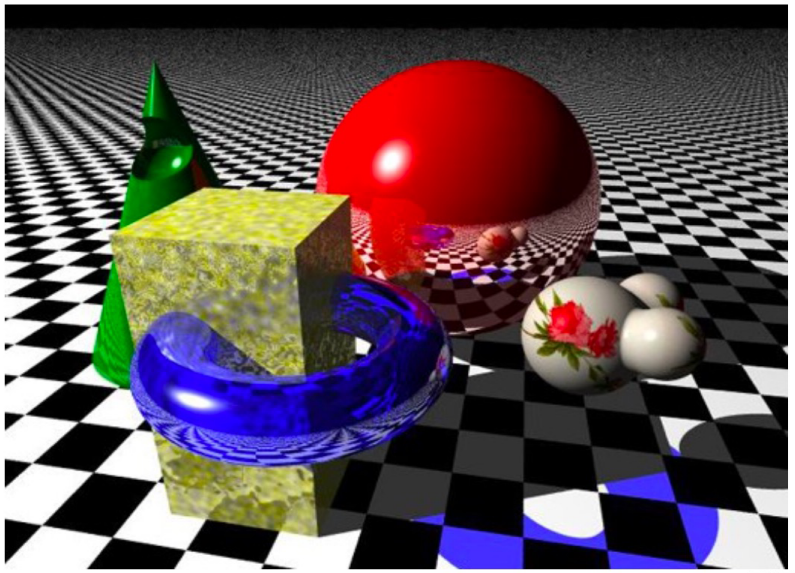


Image taken from http://www.tjhsst.edu/~dhyatt/superap/samplex.jpg

Using implicit models in DL

- Modeling from images

Local Deep Implicit Functions for 3D Shape, Genova & al. 2020

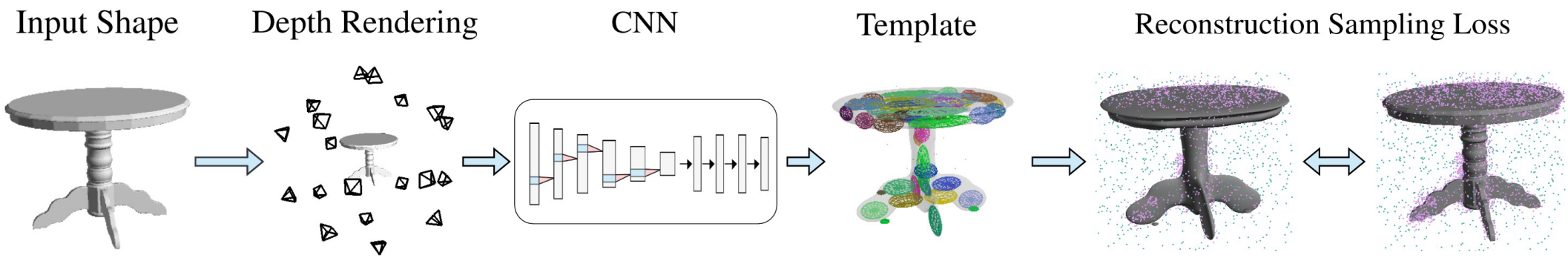
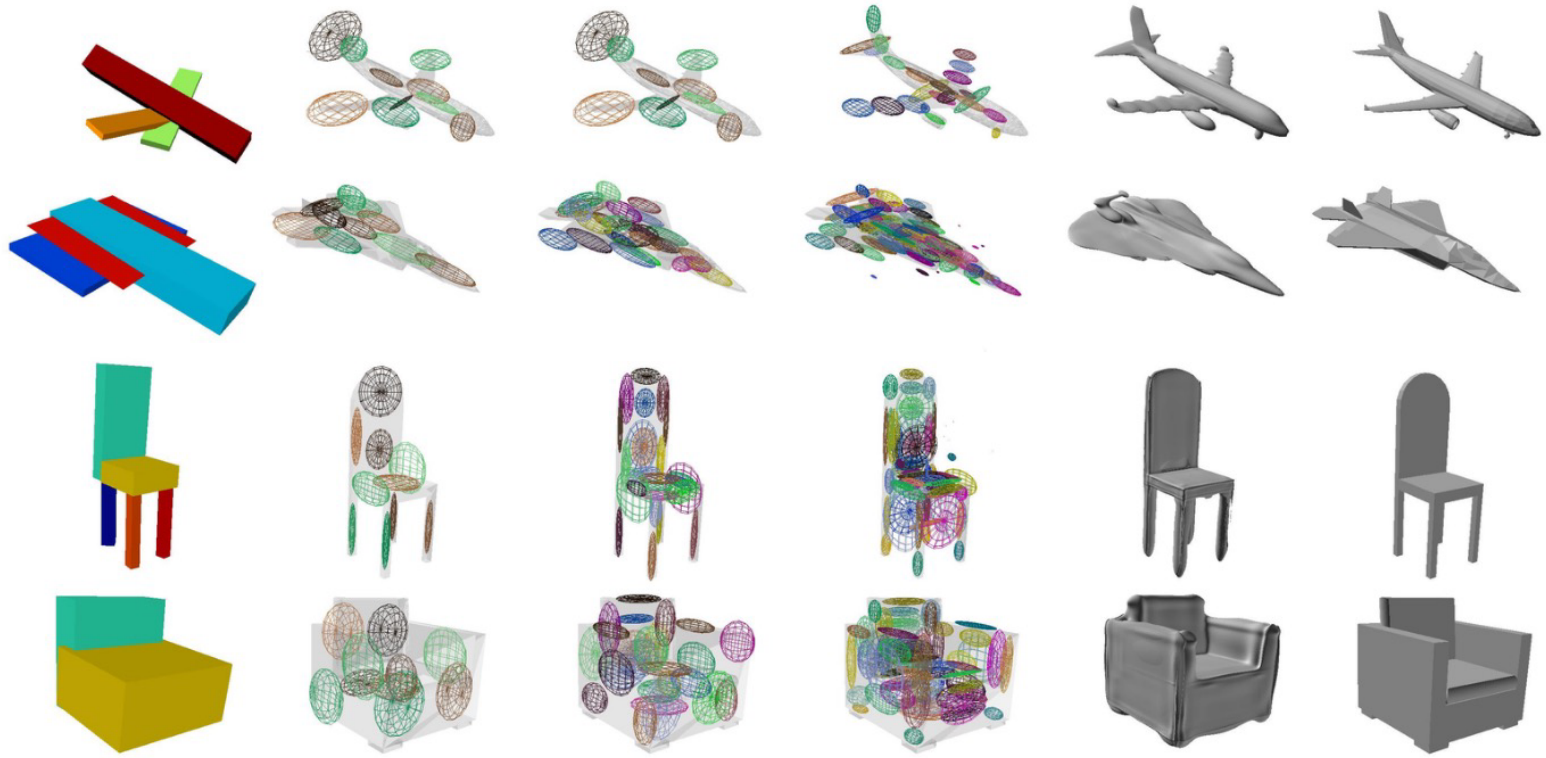


Figure 3. An overview of our method. The input to our system is a mesh. We render a stack of depth images around the mesh, and provide these as input to an early-fusion CNN. The output of the CNN is a vector with fixed dimensionality. This vector is interpreted as a shape template with parameters that define an implicit surface. Next, we sample points near the ground truth surface and also uniformly in space. A classification loss enforces that each sample point is correctly labeled as inside/outside by the surface reconstruction.

- Modeling from images : application to single image reconstruction (learning by categories)

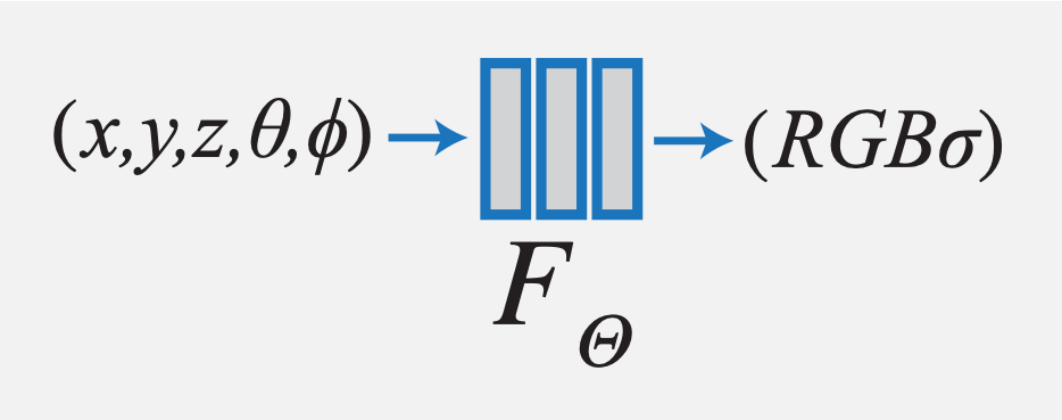
Local Deep Implicit Functions for 3D Shape, Genova & al. 2020



a) VP [51] b) N=10 c) N=25 d) N=100 e) Recon f) GT

NeRF

- Neural Radiance Fields, Mildenhal & al. 2020



F is a implicit representation

See https://www.matthewtancik.com/nerf. (first minute)

NeRF

Drawbacks :

- It is *slow*, both for training and rendering.
- A trained NeRF representation does not generalize to other scenes/objects
- It can only represent static scenes
- It “bakes in” lighting

Lot of follow up work…

NeRF : overview

Neural Radiance Field: NeRF[1]

- Synthesizing views from unseen positions
- Important explosive paper: ≈ 4000 citations[2,3]

Input

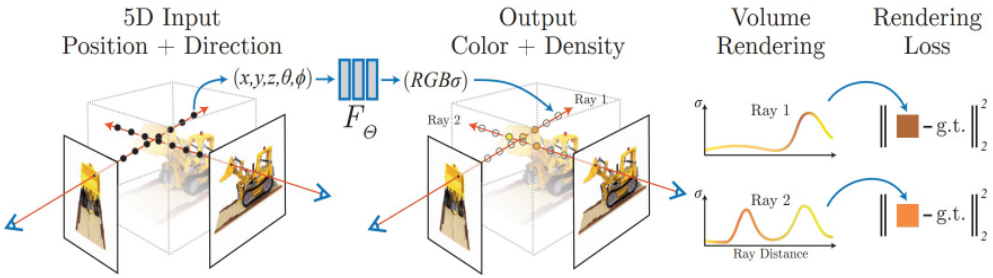
- N color images of a static model
- Perspective cameras: known positions/direction

Methods

- Differentiable Volume rendering
- Neural network : Multi Layer Perceptron (MLP)
- (xyz, θΦ) position and direction (RGB, σ) color and density

Training

- Generation of pixel colors
- Calculating least squares L2 loss
- Backpropagation



+ Quality generated images

- Long training and image inference times (offline)
- Low complexity models / poorly defined input images
- Perspective camera

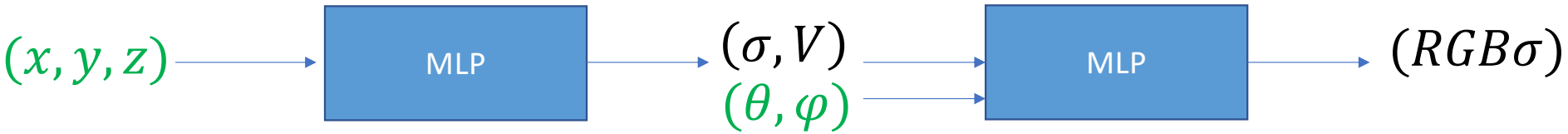
[1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. ***Nerf: Representing scenes as neural radiance fields for view synthesis***. In European Conference on Computer Vision (ECCV), 2020.

[2] K. Gao, Y. Gao, H. He, D. Lu, L. Xu and J. Li. ***NeRF: Neural Radiance Field in 3D Vision, A Comprehensive Review.*** Preprint, 2023

[3] According to Google Scholar.

NeRF : 3D model

More precisely, first estimate density σ given (x,y,z) position in space (space occupancy)



V 256-vector (latent space)

MLP : 8 layers

Input

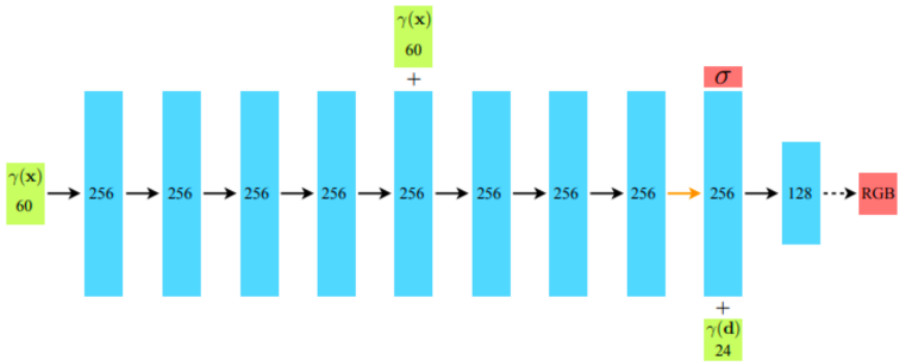
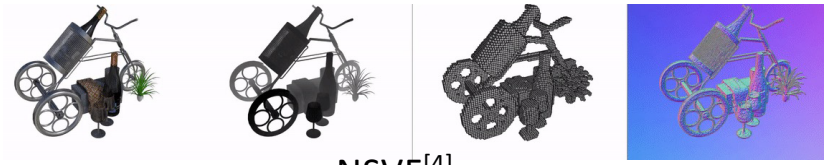


Figure 1The NeRF's fully connected network architecture.

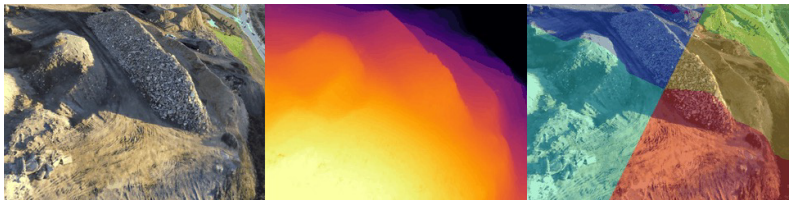
Multi-resolution in NeRF

Related work: NeRF variations

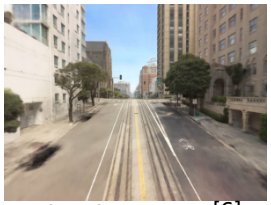
- Acceleration: *Neural sparse voxel fields (NSVF)*[4]
  - Low detail over large areas.
- Large-scale and composition: *Mega-NeRF*[5], *Block-NeRF*[6]
  - Not considering the objects of interest
- Geometry Editing: *Object-NeRF*[7]
  - Require 2D objects mask
- Dynamic and Decomposition: *Neural Scene Graphs*[8]
  - Not provide detailed object rendering quality



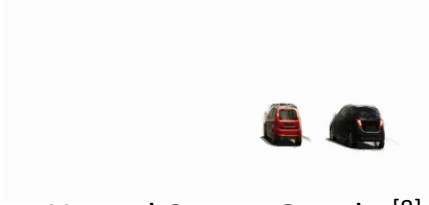
NSVF[4]



Mega-NeRF[5]



Block-NeRF[6]



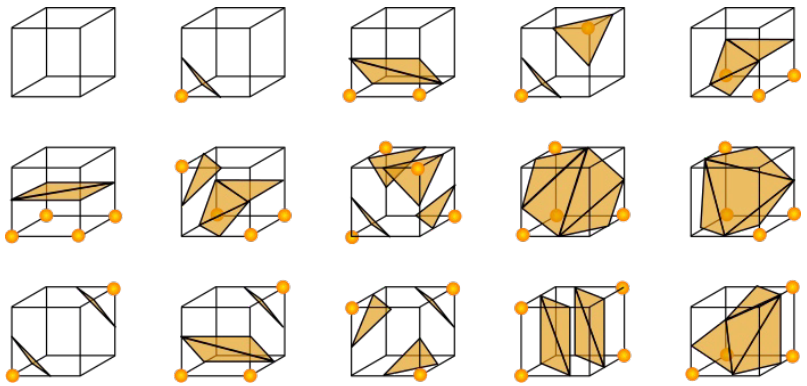
Neural Scene Graphs[8]

Object-NeRF[7]

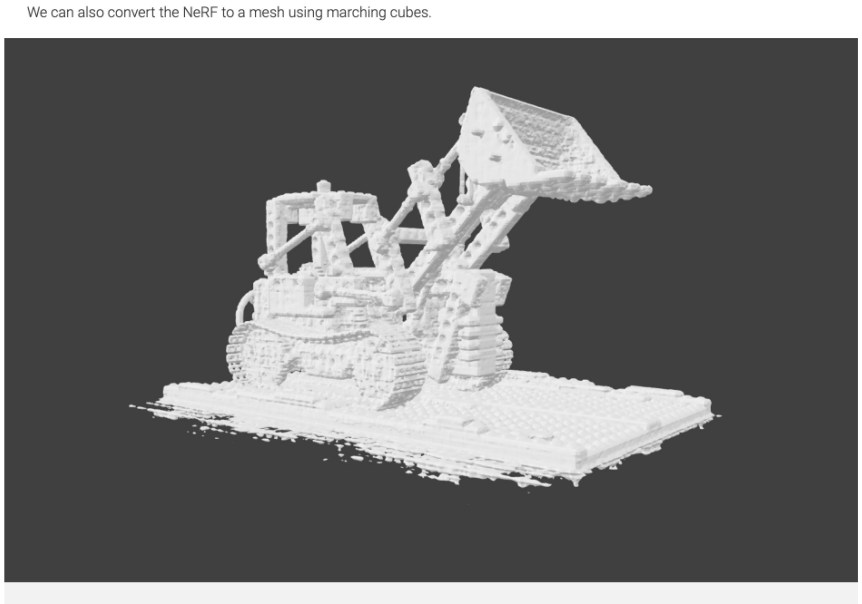
[4] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt. ***Neural sparse voxel fields***. NeurIPS, 2020.
[5] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. Srinivasan, J. T. Barron, and H. Kretzschmar. ***Block-NeRF: Scalable large scene neural view synthesis***. CVPR, 2022.
[6] H. Turki, D. Ramanan, and M. Satyanarayanan. ***Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs***. CVPR, June 2022.
[7] B. Yang, Y. Zhang, Y. Xu, Y. Li, H., Zhou, H., Bao, G. Zhang and Z. Cui. ***Learning Object-Compositional Neural Radiance Field for Editable Scene Rendering***. ICCV, 2021.
[8] J. Ost, F. Mannan, N. Thuerey, J. Knodt and F. Heide. ***Neural Scene Graphs for Dynamic Scenes***. CVPR, 2021.

NeRF : explicit model and appearance

Explicit model through marching cubes



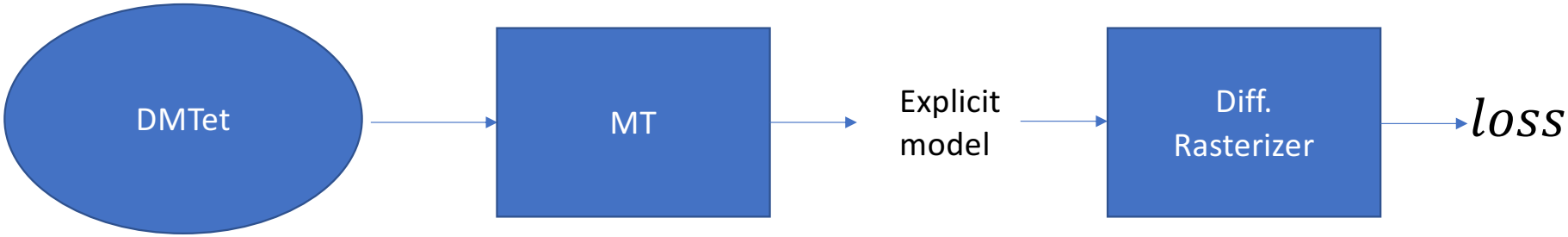
[wikipedia]



(see last video of view dependant appearance @https://www.matthewtancik.com/nerf)

Extracting Triangular 3D Model, Material and Lighting from Images, Munkberg et al, 2022

Estimating an explicit model (triangular mesh)



DeepMarchingTet : a discrete SDF function

Conclusion

- DL for 3D
  - Voxels ? -> too $$$ bad resolution … but in medical applications
  - Point Sets : first ‘break-through’
  - Implicit representations
    - Needs ray tracing
    - Or … Explicit representation !