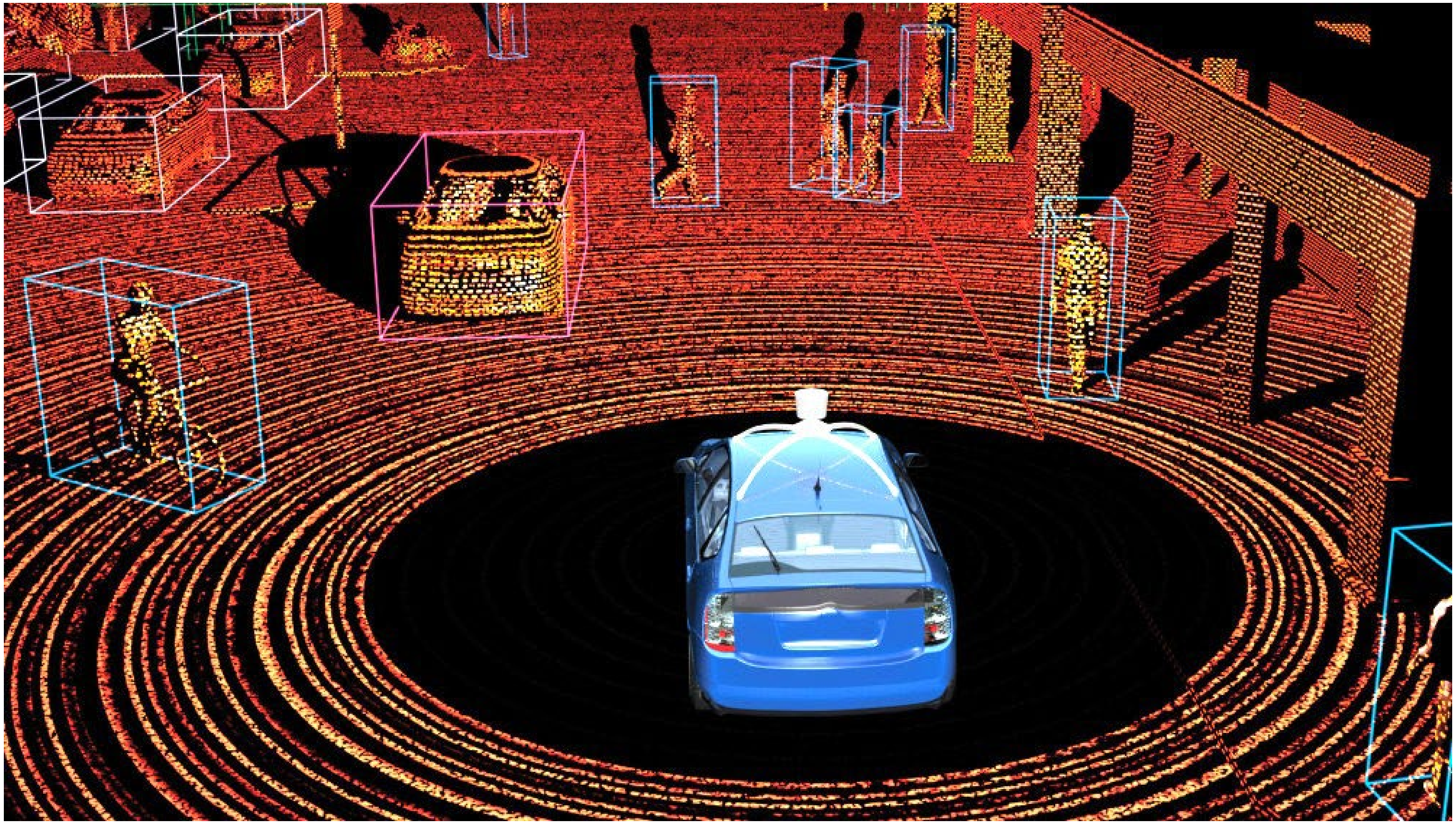
3D Deep Learning approaches
Point-based Networks



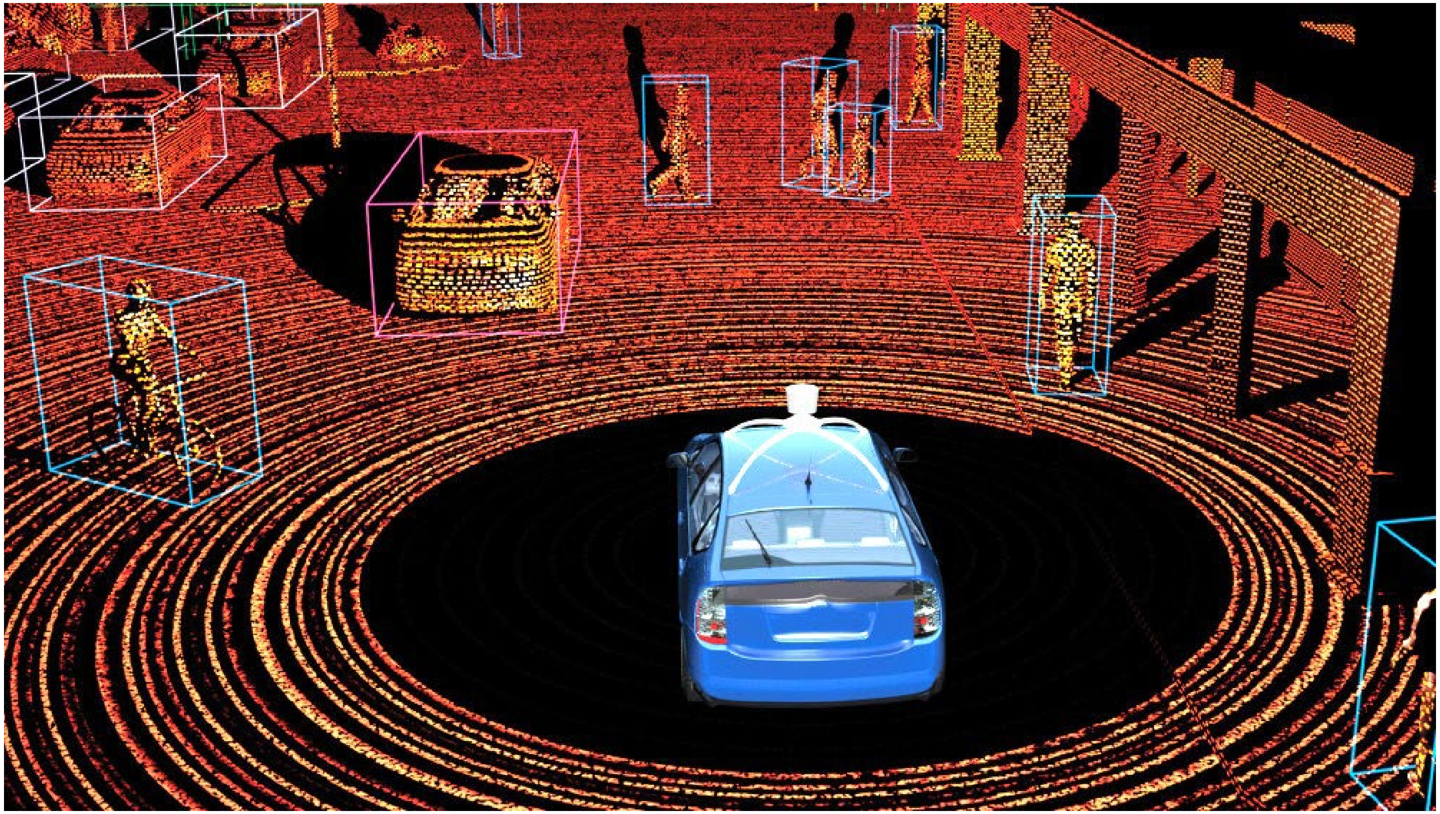
Evangelos Kalogerakis
574/674

3D Deep Learning approaches

- The Multi-View approach
- The Voxel approach
- The Point approach
  - PointNet
  - PointNet++
  - KPConv
- The Graph approach

Motivation

Lots of **scanned data** are raw 3D point clouds



https://3dprint.com/116569/self-driving-cars-privacy/

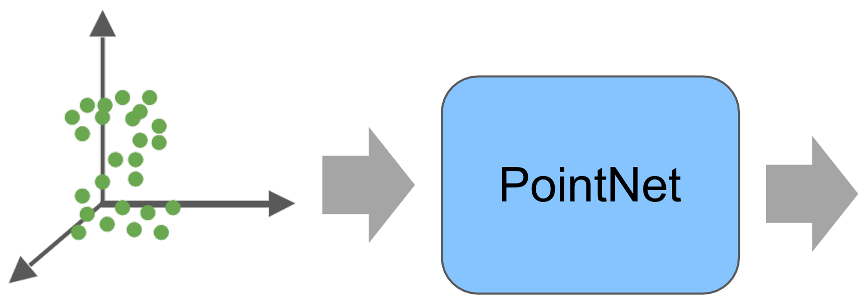
What and where are the objects in a LiDAR scanned scene?

3D Deep Learning approaches

- The Multi-View approach
- The Voxel approach
- The Point approach
  - **PointNet**
  - PointNet++
  - KPConv
- The Graph approach

PointNet

PointNet processes input point clouds for various tasks



Object Classification

Segmentation

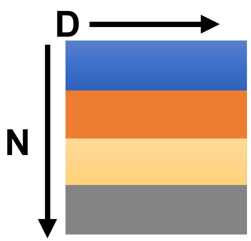
Correspondences

…

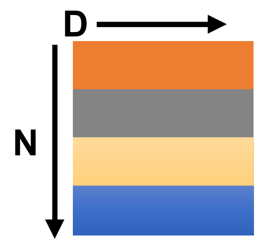
Qi et al. PointNet

Desired Properties of PointNets

Point cloud: N **orderless** points, each represented by a D dim coordinate



represents the same **set** as

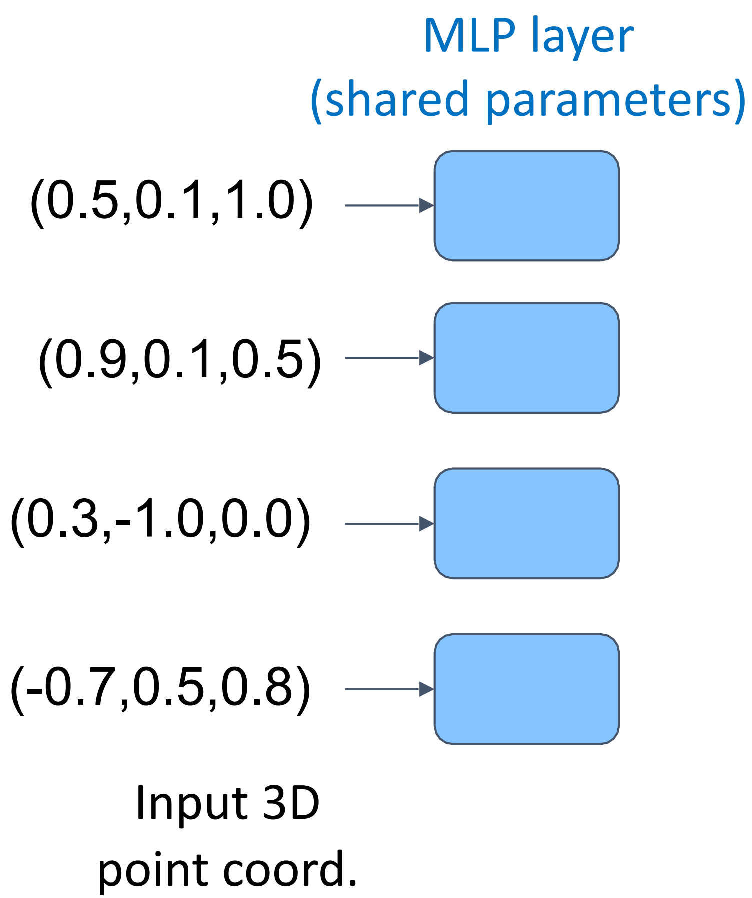


2D array representation

Need permutation invariance!

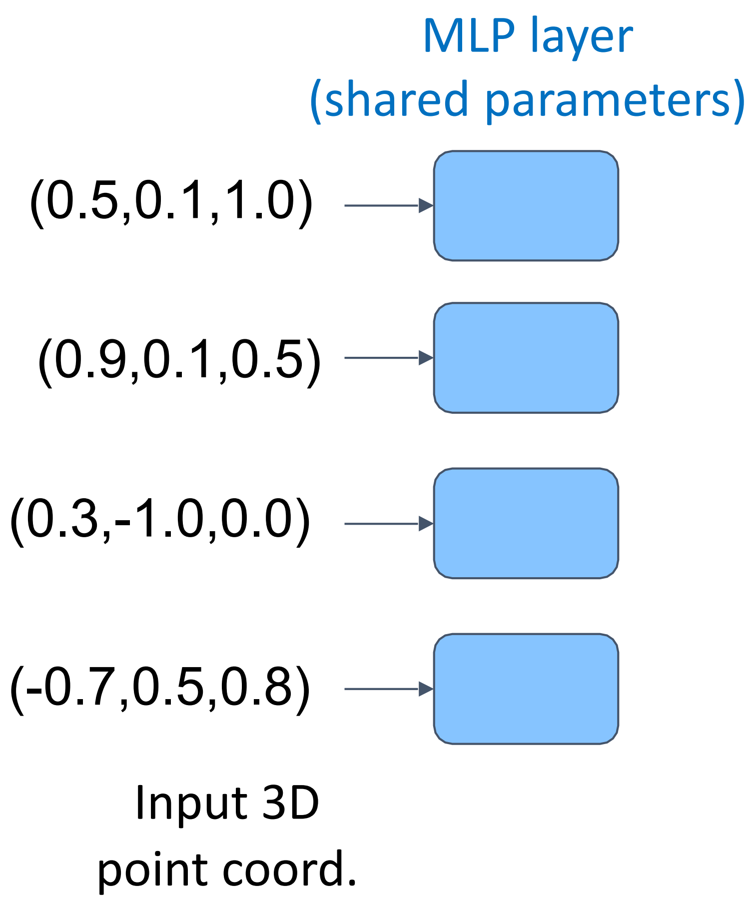
Qi et al. PointNet

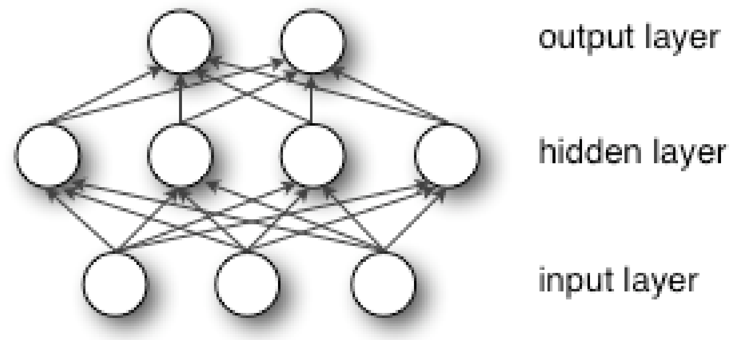
PointNet vanilla architecture



Qi et al. PointNet

PointNet vanilla architecture

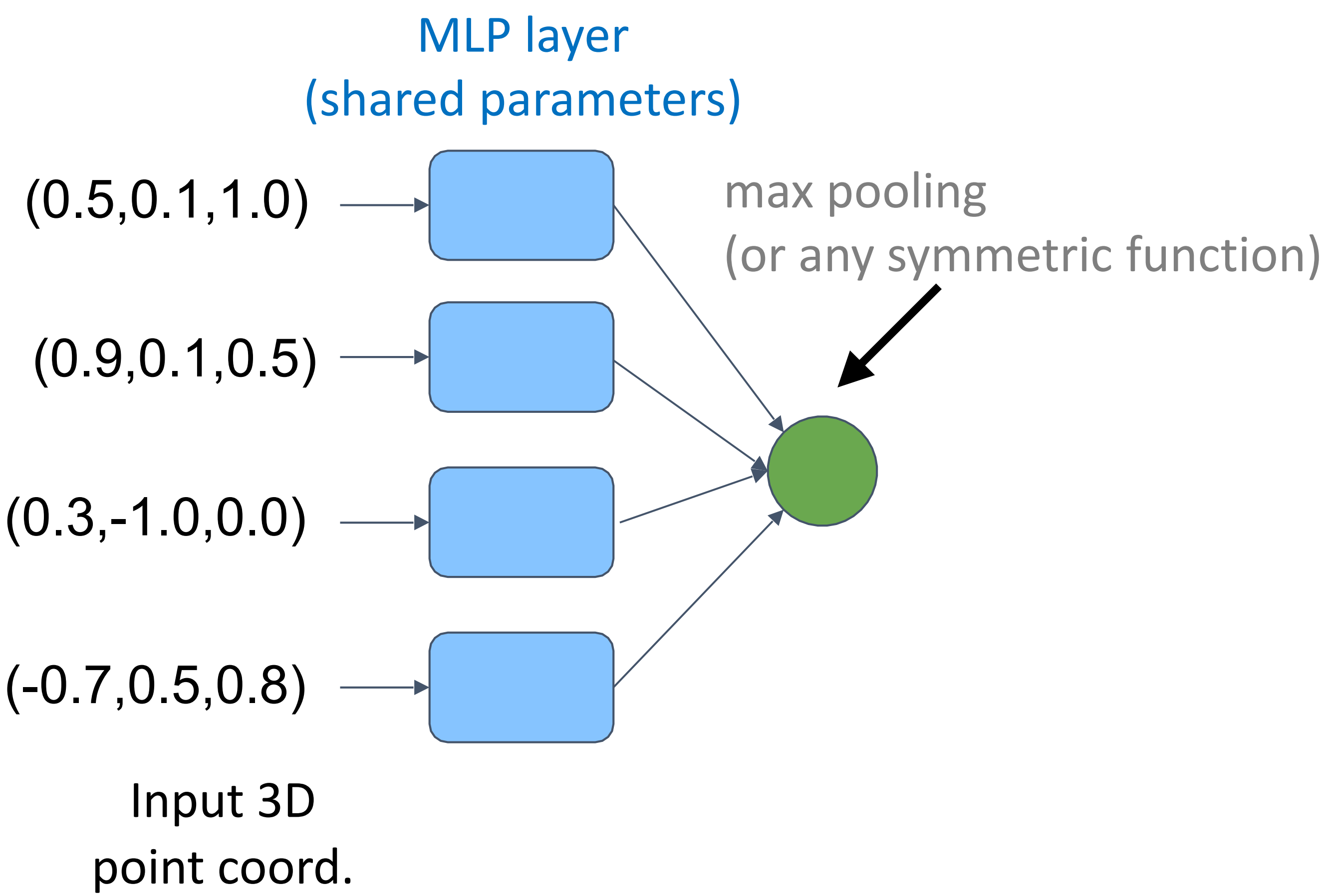




Simply a fully connected NN with one hidden layer, 3 inputs for 3D points, and T outputs (T is layer parameter)

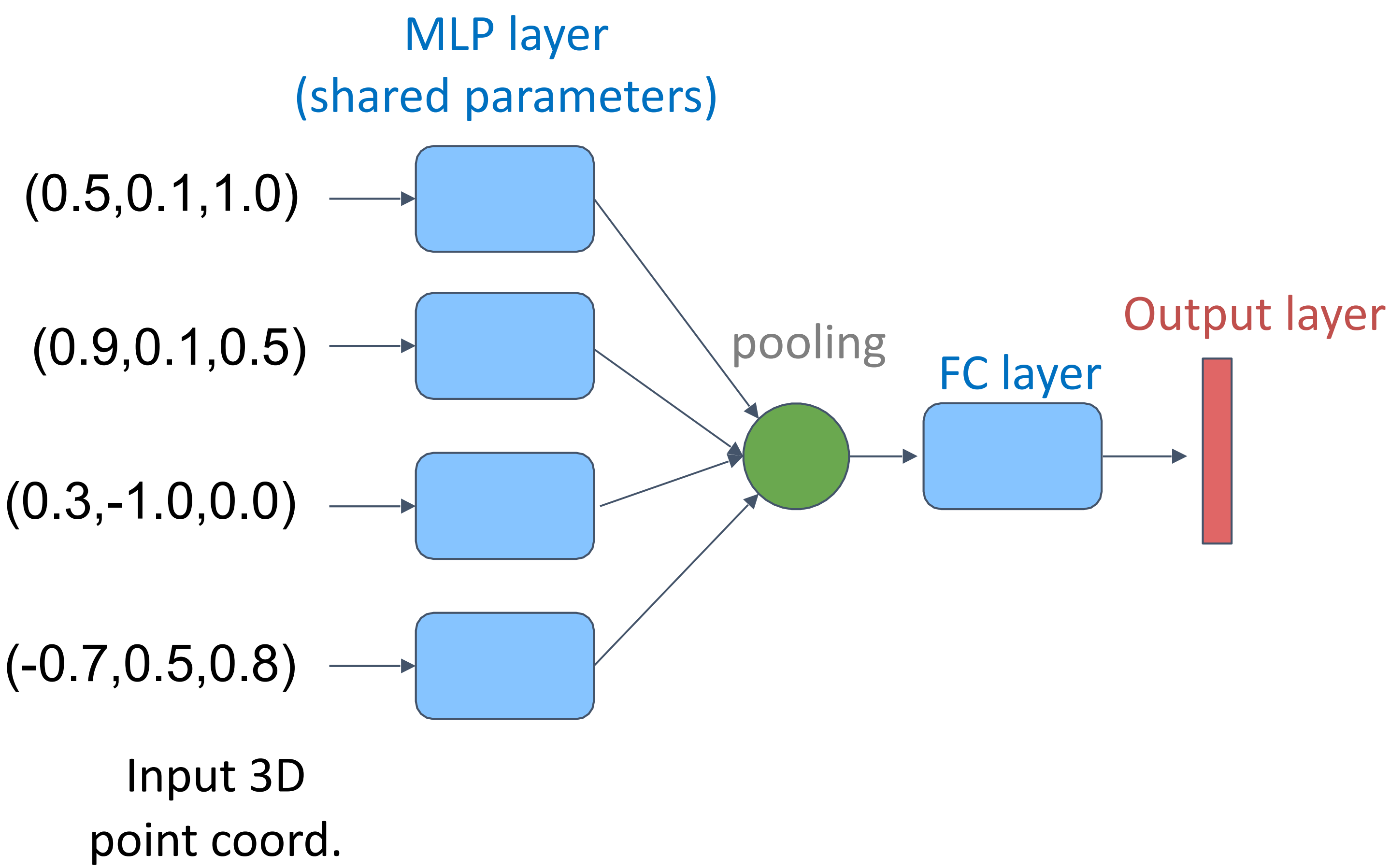
Qi et al. PointNet

PointNet vanilla architecture



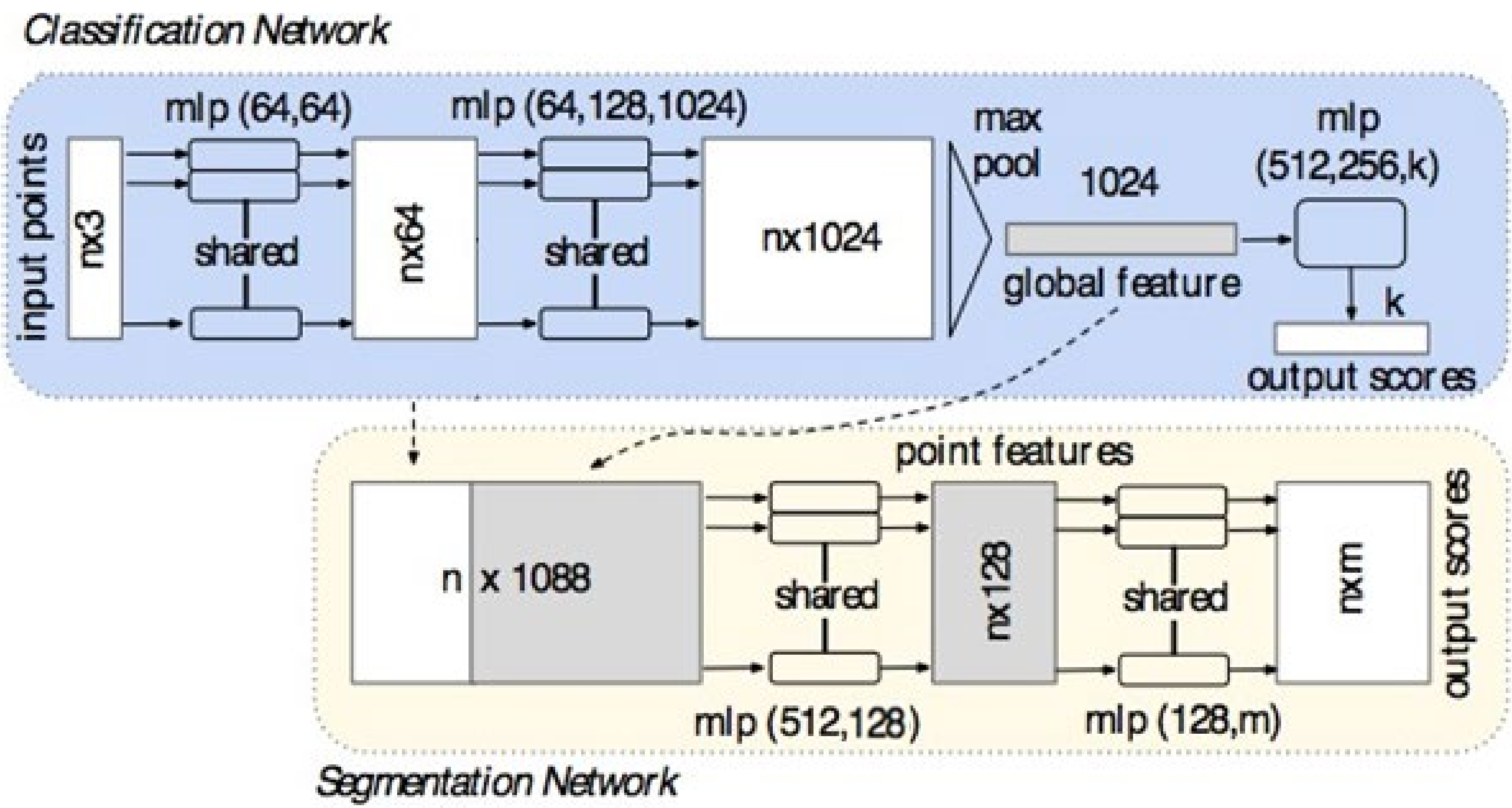
Qi et al. PointNet

PointNet vanilla architecture



Qi et al. PointNet

Basic PointNet architecture



Qi et al. PointNet

Experiments

Object Classification Accuracy on ModelNet40

	input	accuracy avg. class
3DShapeNets [29]	volume	77.3
MVCNN [24]	image	**90.1**
Ours PointNet	point	86.2

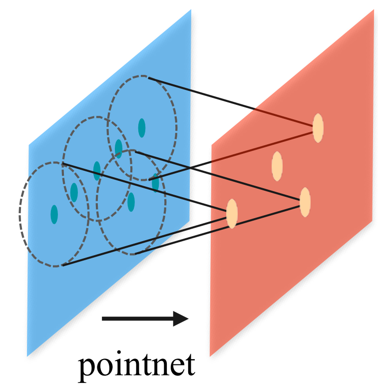
Qi et al. PointNet

3D Deep Learning approaches

- The Multi-View approach
- The Voxel approach
- The Point approach
  - PointNet
  - **PointNet++**
  - KPConv
  - *Application: Point Cloud Registration*
- The Graph approach

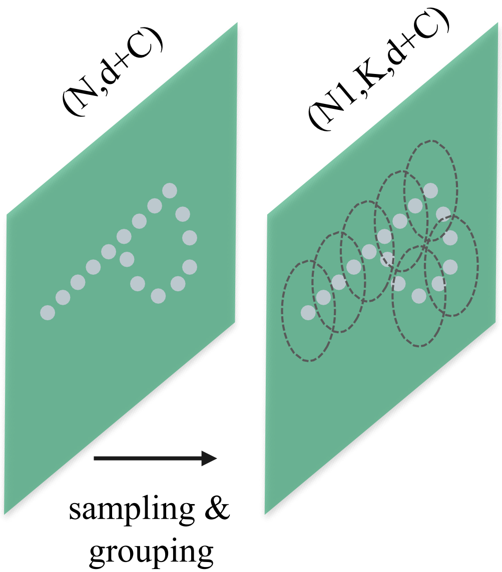
PointNet++

Use pointnet in local regions, aggregate local features by pointnet again
=> **hierarchical feature learning**



Qi et al. PointNet++

PointNet++

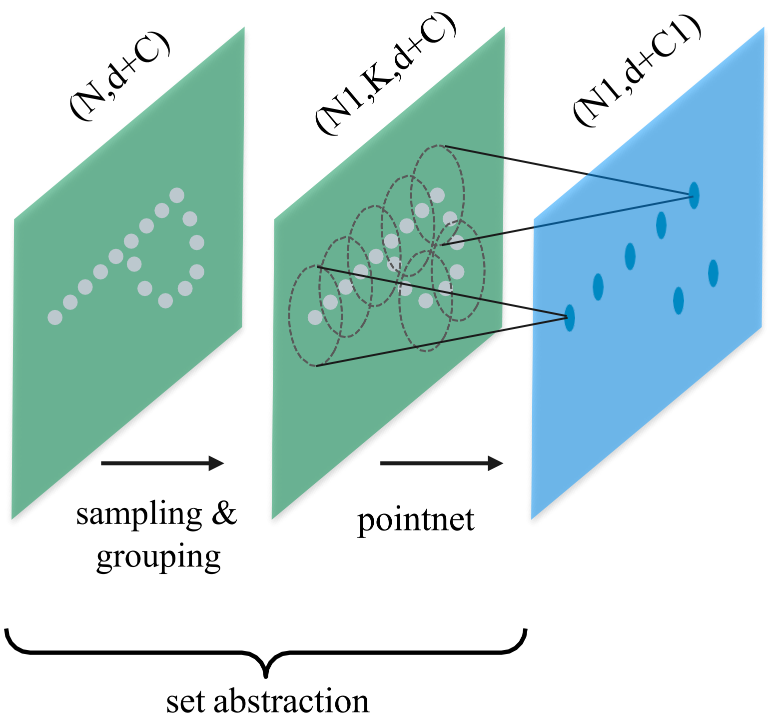


Sampling: Farthest Point Sampling (FPS)

Grouping: radius based ball query

Qi et al. PointNet++

PointNet++

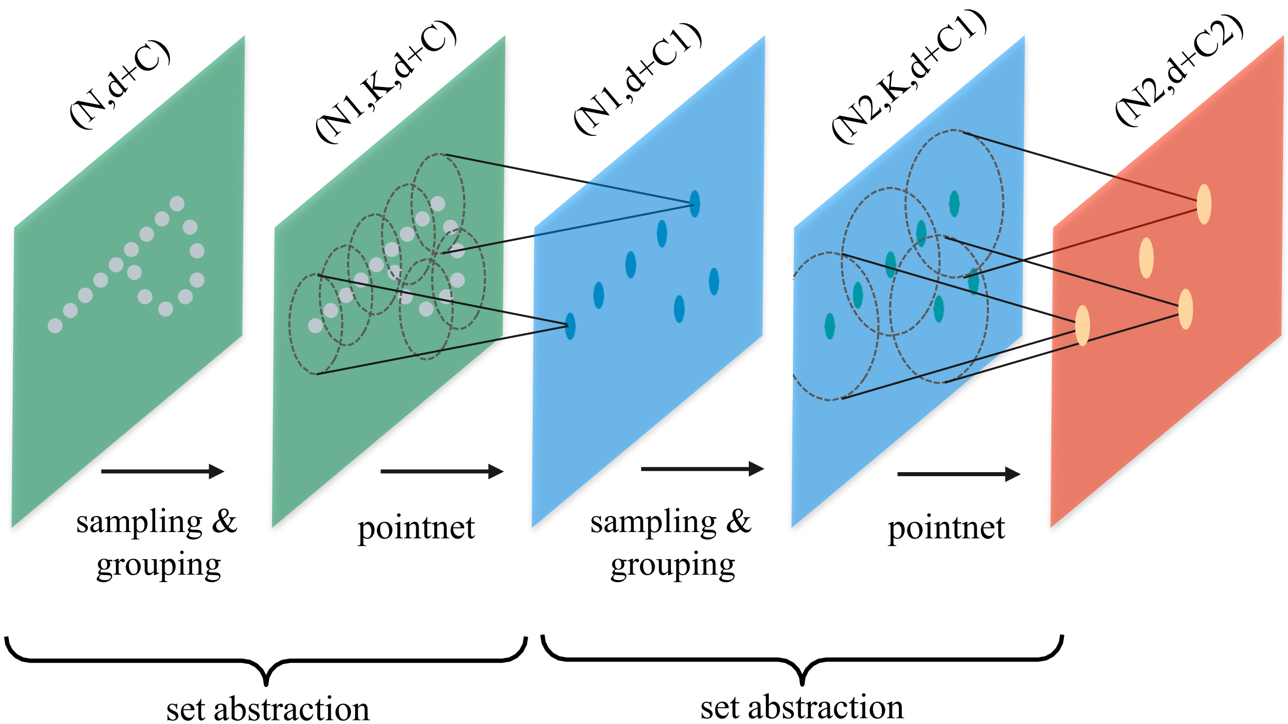


Shared pointnet applied in each local region using local coord.

Qi et al. PointNet++

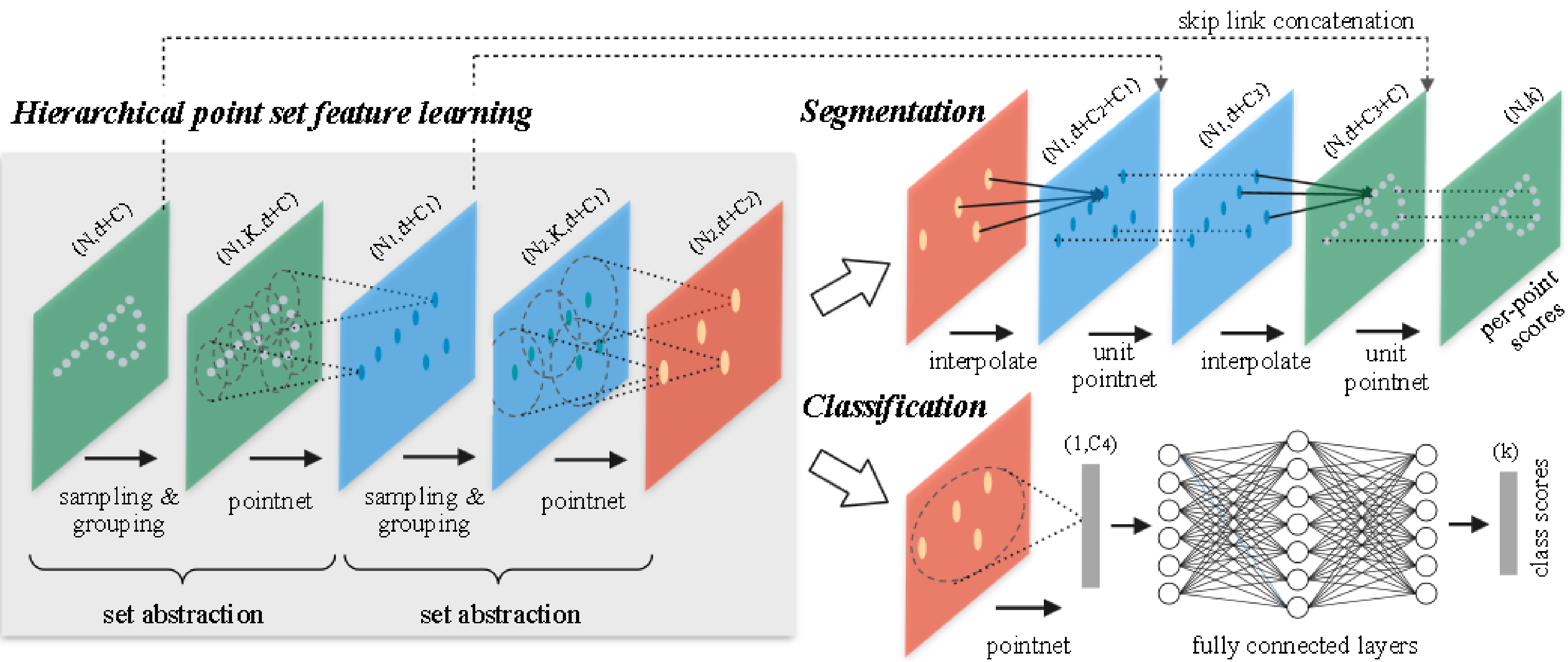
PointNet++

Apply pointnet multiple times:



Qi et al. PointNet++

PointNet++ architecture



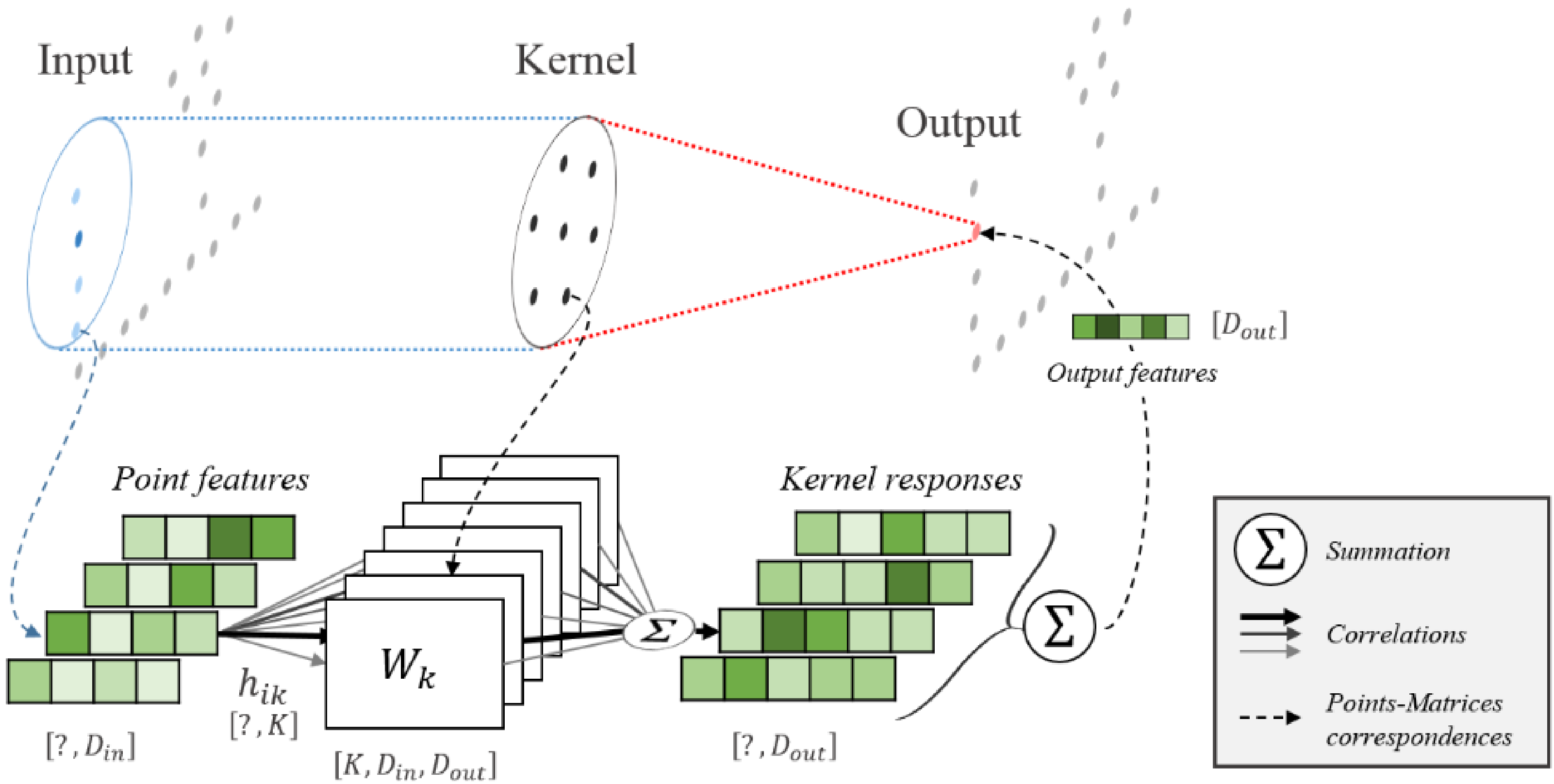
Qi et al. PointNet++

3D Deep Learning approaches

- The Multi-View approach
- The Voxel approach
- **The Point approach**
  - PointNet
  - PointNet++
  - **KPConv**
  - *Application: Point Cloud Registration*
- The Graph approach

KPConv

A filter is a collection of points (kernel). Convolving such kernel at an input point requires a new convolution operator.



Thomas et al. ICCV 19

“KPConv” convolution with rigid kernel

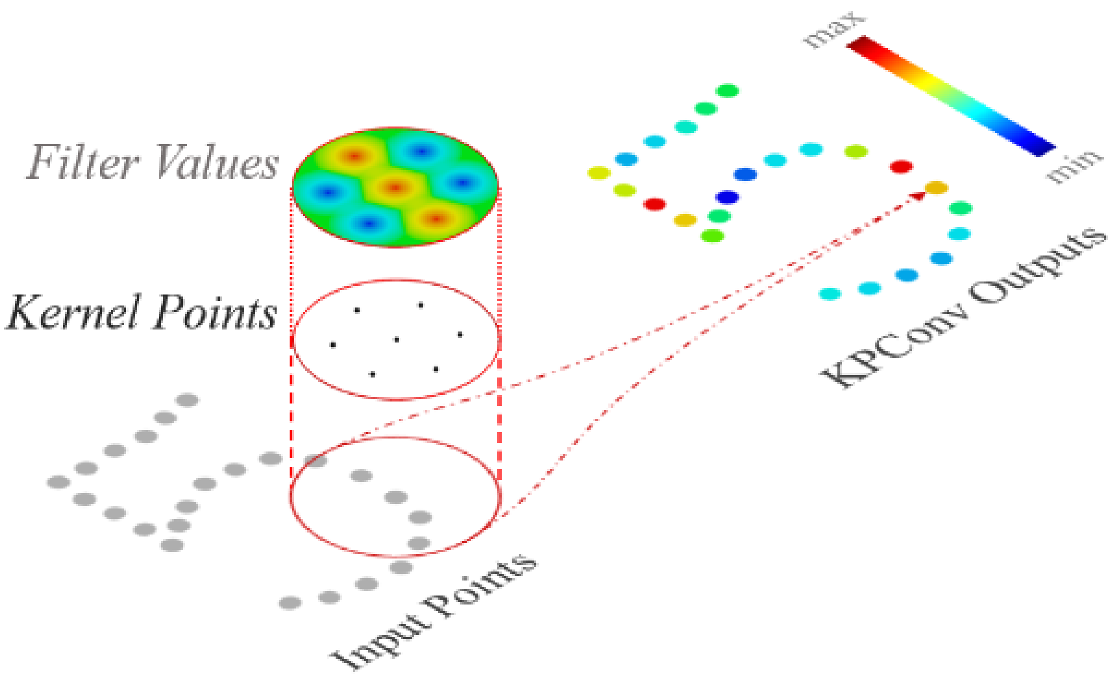


Figure 1. KPConv illustrated on 2D points. Input points with a constant scalar feature (in grey) are convolved through a KPConv that is defined by a set of kernel points (in black) with filter weights on each point.

Thomas et al. ICCV 19

Convolution for points

$$O(x,y,q) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} \sum_{channel\ c} w_q(k,l,c) I(x+k, y+l, c)$$

$$O(\mathbf{p},q) = \sum_{\mathbf{p_i} \in Nb(\mathbf{p})} \sum_{channel\ c} w_q(\mathbf{p_i} - \mathbf{p}, c) \mathbf{f_i}(c)$$

$\mathbf{f_i}$ stores feature vector at $\mathbf{p_i}$ (incl 1 entry for bias)

Retrieve neighbors inside a ball centered at $\mathbf{p}$

Thomas et al. ICCV 19

Convolution for points

$$O(x,y,q) = \sum_{k=-n}^{k=n} \sum_{l=-n}^{l=n} \sum_{channel\ c} w_q(k,l,c) I(x+k, y+l, c)$$

$$O(\mathbf{p},q) = \sum_{\mathbf{p_i} \in Nb(\mathbf{p})} \sum_{channel\ c} w_q(\mathbf{p_i} - \mathbf{p}, c) \mathbf{f_i}(c)$$

$\mathbf{f_i}$ stores feature vector at $\mathbf{p_i}$ (incl 1 entry for bias)

$$O(\mathbf{p},q) = \sum_{\mathbf{p_i} \in Nb(\mathbf{p})} \mathbf{w}_q(\mathbf{p_i} - \mathbf{p}) \bullet \mathbf{f_i}$$

a bit more compactly:
use a dot product instead
sum over channels

Thomas et al. ICCV 19

“KPConv” convolution with rigid kernel

$$O(\mathbf{p}, q) = \sum_{\mathbf{p_i} \in Nb(\mathbf{p})} \mathbf{w}_q(\mathbf{p_i} - \mathbf{p}) \bullet \mathbf{f_i}$$

Thomas et al. ICCV 19

“KPConv” convolution with rigid kernel

$$O(\mathbf{p}, q) = \sum_{\mathbf{p_i} \in Nb(\mathbf{p})} \mathbf{w}_q(\mathbf{p_i} - \mathbf{p}) \bullet \mathbf{f_i}$$

$$\boxed{\mathbf{w}_q(\mathbf{p_i} - \mathbf{p}) = \sum_{\substack{\text{kernel} \\ \text{point k}}} h(\mathbf{p_i} - \mathbf{p},\ \mathbf{x_k})\mathbf{w_k}}$$

measure correlation of each kernel point $\mathbf{x_k}$ with neighbor $\mathbf{p_i}$’s relative position,
$\mathbf{w_k}$ is a learnable weight vector for kernel point

Thomas et al. ICCV 19

"KPConv" convolution with rigid kernel

$$O(\mathbf{p}, q) = \sum_{\mathbf{p_i} \in Nb(\mathbf{p})} \mathbf{w}_q(\mathbf{p_i} - \mathbf{p}) \bullet \mathbf{f_i}$$

$$\boxed{\mathbf{w}_q(\mathbf{p_i} - \mathbf{p}) = \sum_{\substack{\text{kernel} \\ \text{point k}}} h(\mathbf{p_i} - \mathbf{p},\ \mathbf{x_k})\mathbf{w_k}}$$

measure correlation of each kernel point $\mathbf{x_k}$ with neighbor $\mathbf{p_i}$'s relative position, $\mathbf{w_k}$ is a learnable weight vector for kernel point

$$\boxed{h(\mathbf{p_i} - \mathbf{p},\ \mathbf{x_k}) = \max\left(0, 1 - \frac{\| \mathbf{p_i} - \mathbf{p} - \mathbf{x_k} \|}{\sigma}\right)}$$

Thomas et al. ICCV 19

Rigid Kernel examples

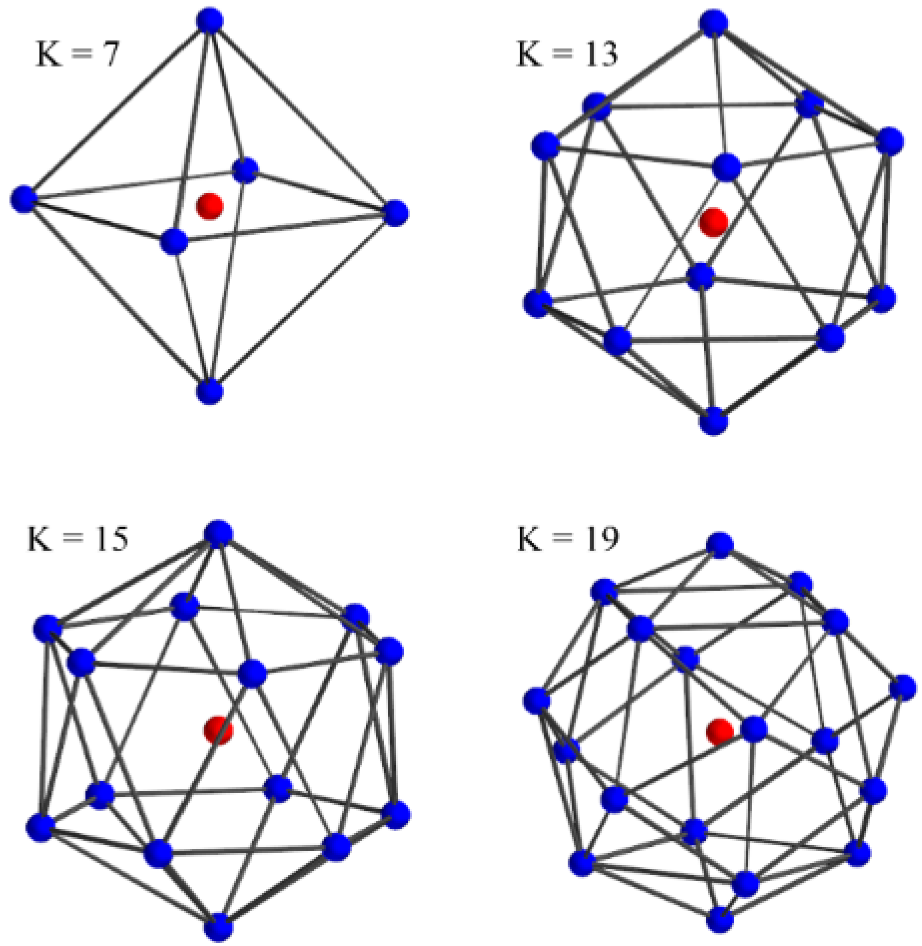
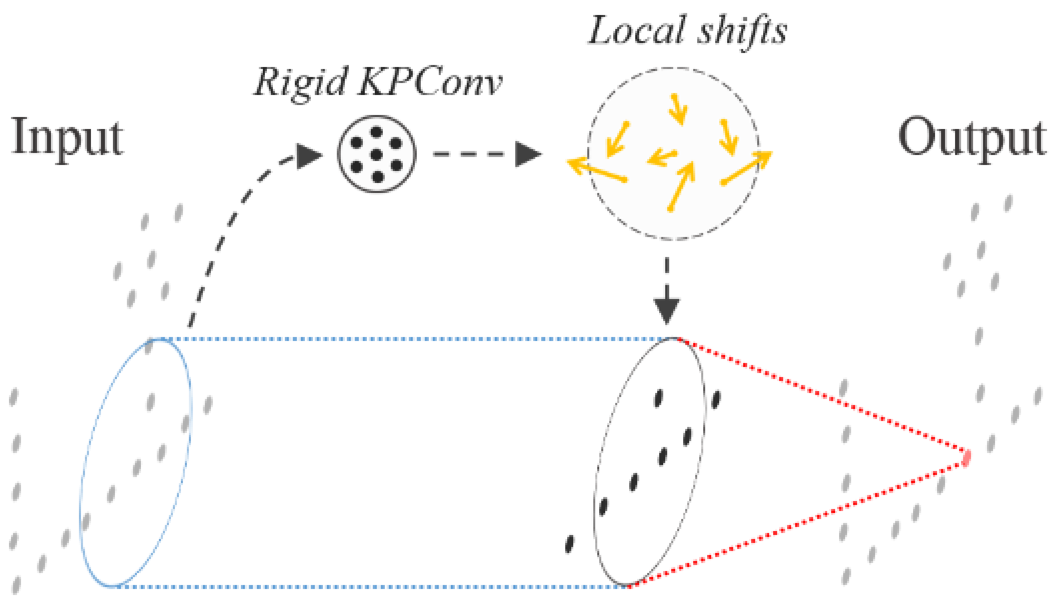


Figure 10. Illustration of the kernel points in stable dispositions.

Thomas et al. ICCV 19

“KPConv” convolution with deformable kernel

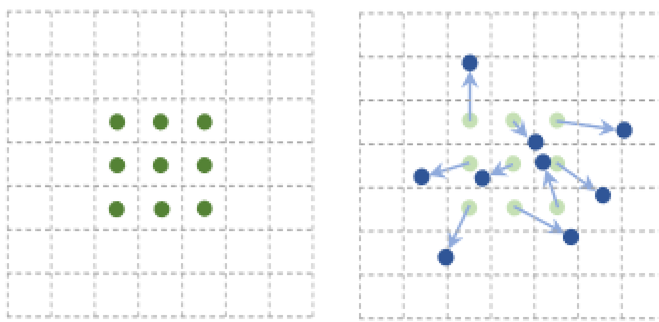


$$\mathbf{w}_q(\mathbf{p}_i - \mathbf{p}) = \sum_{\substack{\text{kernel}\\\text{point k}}} h(\mathbf{p}_i - \mathbf{p},\ \mathbf{x}_k + \Delta\mathbf{x}_k)\mathbf{w}_k$$

learns to perturb kernel points
=> Deformable kernel

Thomas et al. ICCV 19

Deformable convolution in 2D

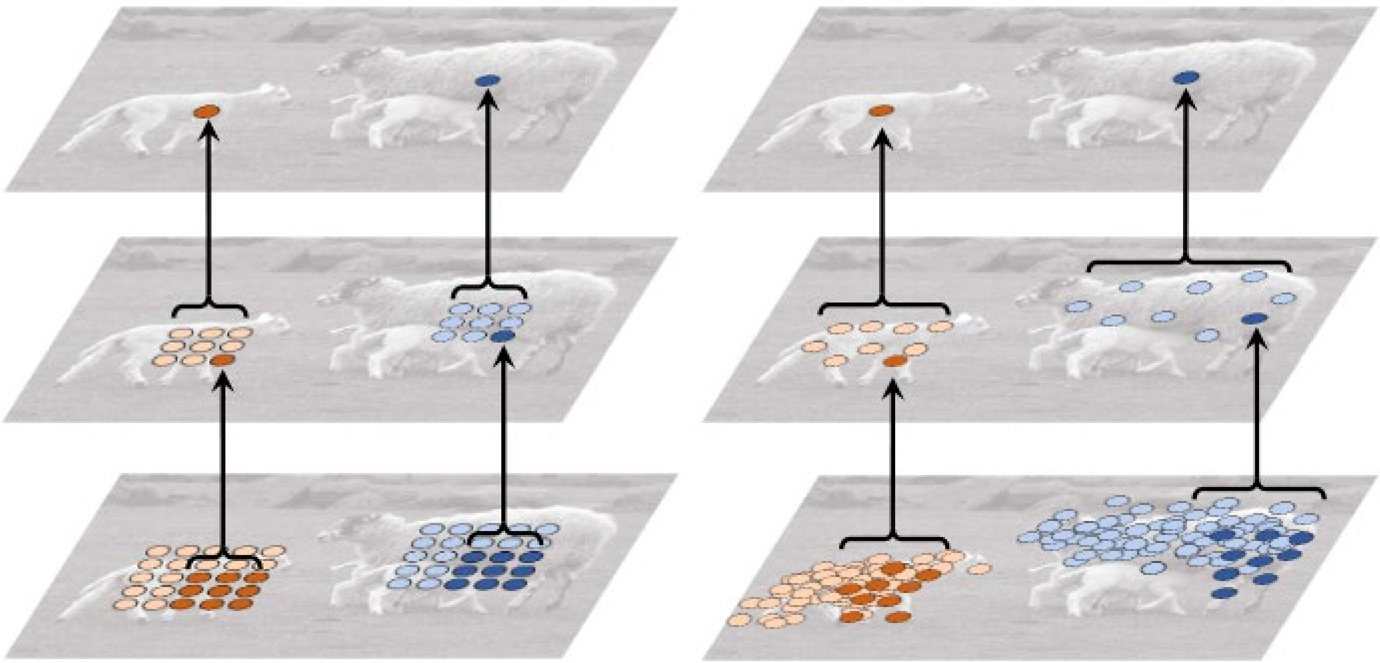


(a) (b)

Figure 1: Illustration of the sampling locations in 3×3 standard and deformable convolutions. (a) regular sampling grid (green points) of standard convolution. (b) deformed sampling locations (dark blue points) with augmented offsets (light blue arrows) in deformable convolution.

Deformable Convolutional Networks, ICCV17
https://arxiv.org/abs/1703.06211

Deformable convolution in 2D



(a) standard convolution (b) deformable convolution

Deformable Convolutional Networks, ICCV17
https://arxiv.org/abs/1703.06211

Performance

Excellent performance on segmentation (esp for scenes)

	ModelNet40	ShapeNetPart	
Methods	OA	mcIoU	mIoU
SPLATNet [34]	-	83.7	85.4
SGPN [42]	-	82.8	85.8
3DmFV-Net [9]	91.6	81.0	84.3
SynSpecCNN [48]	-	82.0	84.7
RSNet [15]	-	81.4	84.9
SpecGCN [40]	91.5	-	85.4
PointNet++ [27]	90.7	81.9	85.1
SO-Net [19]	90.9	81.0	84.9
PCNN by Ext [2]	92.3	81.8	85.1
SpiderCNN [45]	90.5	82.4	85.3
MCConv [13]	90.9	-	85.9
FlexConv [10]	90.2	84.7	85.0
PointCNN [20]	92.2	84.6	86.1
DGCNN [43]	92.2	85.0	84.7
SubSparseCNN [9]	-	83.3	86.0
KPConv *rigid*	**92.9**	85.0	86.2
KPConv *deform*	92.7	**85.1**	**86.4**

Table 1. 3D Shape Classification and Segmentation results.

Methods	Scannet	Sem3D	S3DIS	PL3D
Pointnet [26]	-	-	41.1	-
Pointnet++ [27]	33.9	-	-	-
SnapNet [4]	-	59.1	-	-
SPLATNet [34]	39.3	-	-	-
SegCloud [37]	-	61.3	48.9	-
RF_MSSF [38]	-	62.7	49.8	56.3
Eff3DConv [50]	-	-	51.8	-
TangentConv [36]	43.8	-	52.6	-
MSDVN [30]	-	65.3	54.7	66.9
RSNet [15]	-	-	56.5	-
FCPN [28]	44.7	-	-	-
PointCNN [20]	45.8	-	57.3	-
PCNN [2]	49.8	-	-	-
SPGraph [17]	-	73.2	58.0	-
ParamConv [41]	-	-	58.3	-
SubSparseCNN [9]	**72.5**	-	-	-
KPConv *rigid*	68.6	**74.6**	65.4	72.3
KPConv *deform*	68.4	73.1	**67.1**	**75.9**

Table 2. 3D scene segmentation scores (mIoU). Scannet, Semantic3D and Paris-Lille-3D (PL3D) scores are taken from their respective online benchmarks (reduced-8 challenge for Semantic3D). S3DIS scores are given for Area-5 (see supplementary material for k-fold).

Performance

Excellent performance on segmentation (esp for scenes)

	ModelNet40	ShapeNetPart	
Methods	OA	mcIoU	mIoU
SPLATNet [34]	-	83.7	85.4
SGPN [42]	-	82.8	85.8
3DmFV-Net [9]	91.6	81.0	84.3
SynSpecCNN [48]	-	82.0	84.7
RSNet [15]	-	81.4	84.9
SpecGCN [40]	91.5	-	85.4
PointNet++ [27]	90.7	81.9	85.1
SO-Net [19]	90.9	81.0	84.9
PCNN by Ext [2]	92.3	81.8	85.1
SpiderCNN [45]	90.5	82.4	85.3
MCConv [13]	90.9	-	85.9
FlexConv [10]	90.2	84.7	85.0
PointCNN [20]	92.2	84.6	86.1
DGCNN [43]	92.2	85.0	84.7
SubSparseCNN [9]	-	83.3	86.0
KPConv *rigid*	**92.9**	85.0	86.2
KPConv *deform*	92.7	**85.1**	**86.4**

Table 1. 3D Shape Classification and Segmentation results.

Methods	Scannet	Sem3D	S3DIS	PL3D
Pointnet [26]	-	-	41.1	-
Pointnet++ [27]	33.9	-	-	-
SnapNet [4]	-	59.1	-	-
SPLATNet [34]	39.3	-	-	-
SegCloud [37]	-	61.3	48.9	-
RF_MSSF [38]	-	62.7	49.8	56.3
Eff3DConv [50]	-	-	51.8	-
TangentConv [36]	43.8	-	52.6	-
MSDVN [30]	-	65.3	54.7	66.9
RSNet [15]	-	-	56.5	-
FCPN [28]	44.7	-	-	-
PointCNN [20]	45.8	-	57.3	-
PCNN [2]	49.8	-	-	-
SPGraph [17]	-	73.2	58.0	-
ParamConv [41]	-	-	58.3	-
SubSparseCNN [9]	**72.5**	-	-	-
KPConv *rigid*	68.6	**74.6**	65.4	72.3
KPConv *deform*	68.4	73.1	**67.1**	**75.9**

MinkowskiNet42 (2cm)†	**73.4**

scores (mIoU). Scannet, Semantic3D and Paris-Lille-3D (PL3D) scores are taken from their respective online benchmarks (reduced-8 challenge for Semantic3D). S3DIS scores are given for Area-5 (see supplementary material for k-fold).