PAPER ANALYSIS

Presented by Yannis He

-

Paper: **RPVNet: A Deep and Efficient Range-Point-Voxel Fusion Network for LiDAR Point Cloud Segmentation**

Authors: Jianyun Xu, Ruixiang Zhang, Jian Dou, Yushi Zhu, Jie Sun, Shiliang Pu

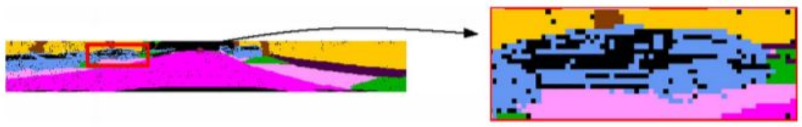
- | Hikvision Research Institute
- | Zhejiang University

Published Date: 24th March, 2021

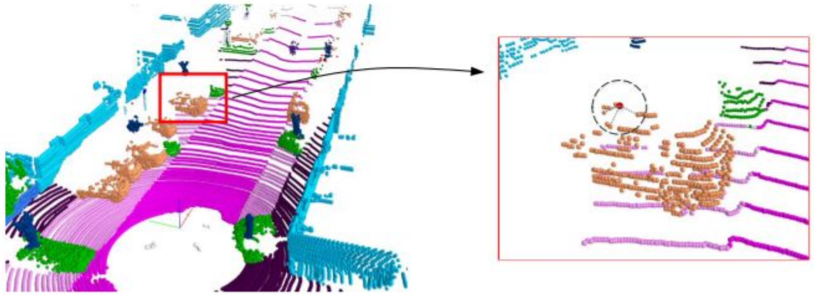
https://arxiv.org/abs/2103.12978

ABSTRACT

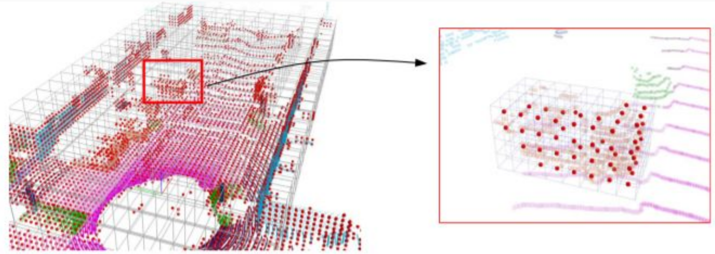
- Background:
  - Point cloud can be represented in many forms (point-based sets, voxel-based cells, range-based images)
    - Point-based view is geometrically accurate by disordered → difficult to find local neighbors efficiently
    - Voxel-based is sparse → computation grows cubicly when voxel resolution increase
    - Range-basd is regular and dense, but spherical projection → distorted physical dimension
- Idea:
  - To utilize different views' advantages:
    - Range-point-voxel fusion network, RPVNet
  - Mutual information interactions among three views
- Contribution:
  - RPVNetwork: current (2021/8/21) state of the art (1st place) for semantic segmentation challenge on *Semantic KITTI*
  - Propose a gated fusion module (GFM), which can adaptively merge 3 features based on concurrent inputs
  - Propose RPV interaction mechanism: highly efficient using a general formulation



(c) Range-based: physical dimensions distorted



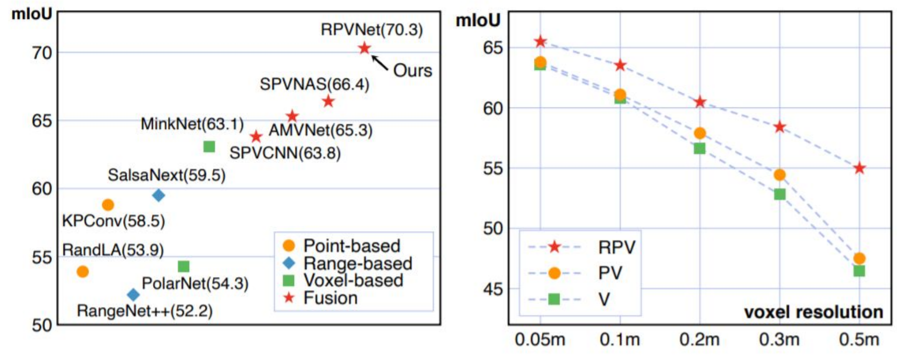
(a) Point-based: disordered



(b) Voxel-based: sparse, quantization loss

INTRO

- Discovery:
  - Voxel-based method are higher in performance than point- and range-base (these 2 have similar performances).
  - Range-based is more efficient since we can use 2D convolution.
    - Point based are far from real-time when involving local neighbor searching
  - Voxel-based are hard to have a high resolution and being efficient at the same time
    - But performance drop sharply if resolution decreases
- Approach: Allows different view to enhance each other in a deeper and flexible way
  - using points as middle hosts, and transfer features on range-pixels and voxel-cells to points.
  - Then apply an adaptive feature selection to choose the best feature representation for each point
  - Transfer the fused features on points back to range-image and voxels



(a) methods vs mIoU (b) voxel resolution vs mIoU

- 3 branches:
  - Voxel-, point- and range-branch from top to down (see image on next page)
  - .Unet for both voxel- and range-branch
    - Use a stem to extract contextual information from original input, then perform down-sampling, and finally up-sampling to restore the original points
  - PointNet for point-branch
- RPV-fusion takes place at
  - After stem, 4th downsample, 2nd up-sample, and last upsampling

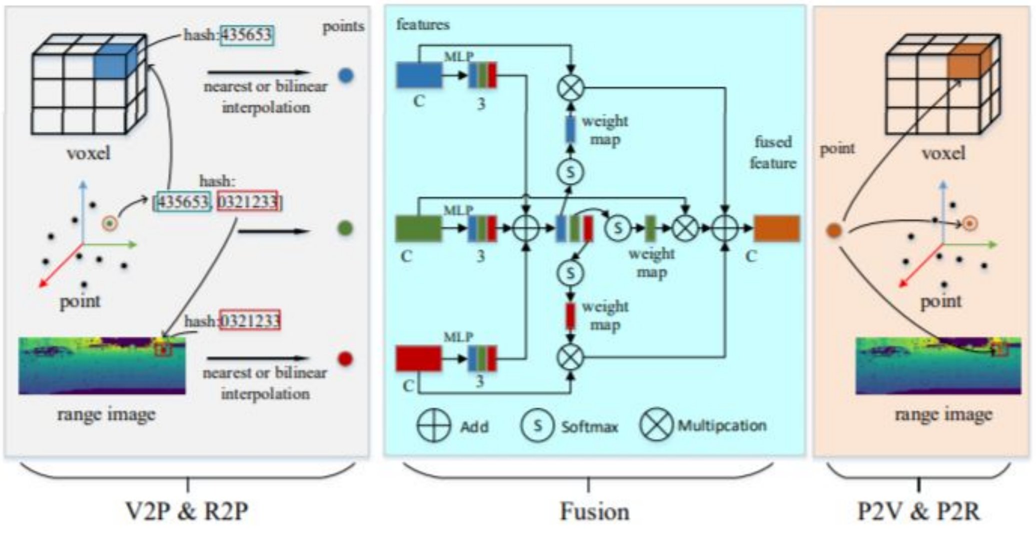


Figure 4. Details of RPV fusion. In left block: Given a point and its hash code, we need to find the corresponding voxel or pixel. In middle block: Given features on point from different views, we need to merge them adaptively. In right block: Given the fused point feature, we need to project it back to other views.

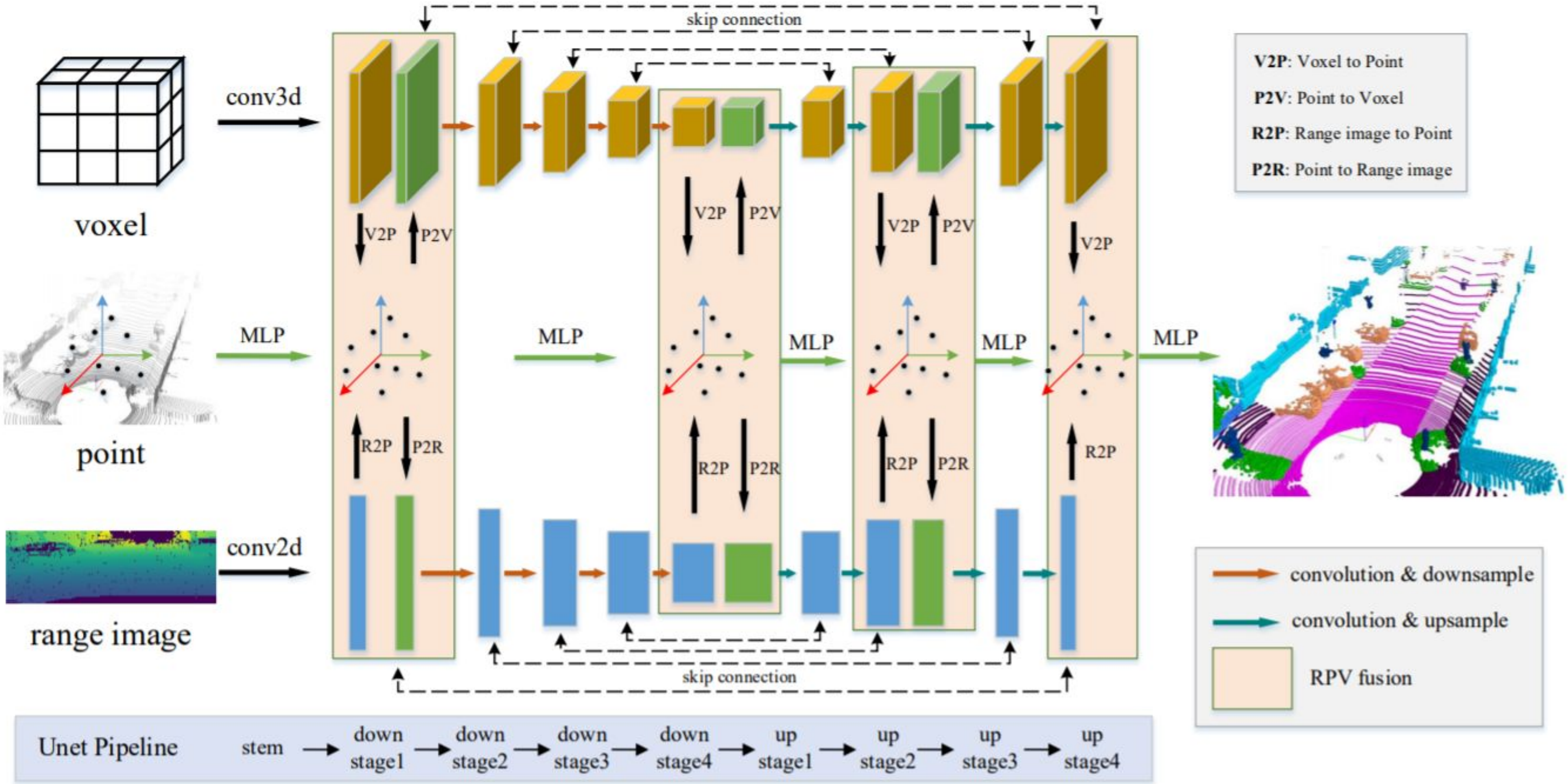


Figure 3. Overview of RPVNet. It is a three-branch network with multiple interactions among them, where voxel- and range-branch share the similar Unet architecture, and point-branch only utilize per-point MLPs.

RESULT

Methods	mIoU	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic-sign
PointNet [24]	14.6	46.3	1.3	0.3	0.1	0.8	0.2	0.2	0.0	61.6	15.8	35.7	1.4	41.4	12.9	31.0	4.6	17.6	2.4	3.7
RandLANet [15]	53.9	94.2	26.0	25.8	40.1	38.9	49.2	48.2	7.2	90.7	60.3	73.7	20.4	86.9	56.3	81.4	61.3	66.8	49.2	47.7
KPConv [31]	58.8	96.0	30.2	42.5	33.4	44.3	61.5	61.6	11.8	88.8	61.3	72.7	31.6	90.5	64.2	84.8	69.2	69.1	56.4	47.4
SqueezeSegv3 [38]	55.9	92.5	38.7	36.5	29.6	33.0	45.6	46.2	20.1	91.7	63.4	74.8	26.4	89.0	59.4	82.0	58.7	65.4	49.6	58.9
RangeNet++ [23]	52.2	91.4	25.7	34.4	25.7	23.0	38.3	38.8	4.8	91.8	65.0	75.2	27.8	87.4	58.6	80.5	55.1	64.6	47.9	55.9
SalsaNext [12]	59.5	91.9	48.3	38.6	38.9	31.9	60.2	59.0	19.4	91.7	63.7	75.8	29.1	90.2	64.2	81.8	63.6	66.5	54.3	62.1
PolarNet [41]	54.3	93.8	40.3	30.1	22.9	28.5	43.2	40.2	5.6	90.8	61.7	74.4	21.7	90.0	61.3	84.0	65.5	67.8	51.8	57.5
MinkowskiNet [10]	63.1*	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Cylinder3D [44]	67.8	97.1	67.6	64.0	**59.0**	58.6	73.9	67.9	36.0	91.4	65.1	75.5	32.3	91.0	66.5	85.4	71.8	68.5	62.6	65.6
AF2S3 [8]	69.7	94.5	65.4	**86.8**	39.2	41.1	**80.7**	**80.4**	**74.3**	91.3	68.8	72.5	**53.5**	87.9	63.2	70.2	68.5	53.7	61.5	**71.0**
FusionNet [39]	61.3	95.3	47.5	37.7	41.8	34.5	59.5	56.8	11.9	91.8	68.8	77.1	30.8	92.5	69.4	84.5	69.8	68.5	60.4	66.5
TornadoNet [14]	63.1	94.2	55.7	48.1	40.0	38.2	63.6	60.1	34.9	89.7	66.3	74.5	28.7	91.3	65.6	85.6	67.0	71.5	58.0	65.9
AMVNet [20]	65.3	96.2	59.9	54.2	48.8	45.7	71.0	65.7	11.0	90.1	**71.0**	75.8	32.4	92.4	69.1	85.6	71.7	69.6	62.7	67.2
SPVCNN [30]	63.8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
SPVNAS [30]	67.0	97.2	50.6	50.4	56.6	58.0	67.4	67.1	50.3	90.2	67.6	75.4	21.8	91.6	66.9	86.1	73.4	71.0	64.3	67.3
RPVNet	**70.3**	**97.6**	**68.4**	68.7	44.2	**61.1**	75.9	74.4	73.4	**93.4**	70.3	**80.7**	33.3	**93.5**	**72.1**	**86.5**	**75.1**	**71.7**	**64.8**	61.4

Table 1. Class-wise and mean IOU of our proposed method and state-of-the-art methods on SemanticKITTI leaderboard. The methods are grouped as point-based, range-based, voxel-based and fusion networks. $*$: result reproduced by [30]. Note that, our result uses the *instance CutMix* augmentation (see Sec. 3.4), and voxel resolution is set to 0.05m, but **without** extra tricks. Accessed on 18 March 2021.