PAPER ANALYSIS

Presented by Yannis He

-

Paper: **Center-based 3D Object Detection and Tracking** (aka. CenterPoint)

Conference: CVPR 2021
Publication Date: 6th Jan, 2021
Authors: Tianwei Yin, Xingyi Zhou, Philipp Krähenbühl | University of Texas at Austin
https://arxiv.org/abs/2006.11275
https://github.com/tianweiy/CenterPoint

ABSTRACT

- Background:
  - 3D object commonly represented as 3D bounding-boxes in point cloud, which mimics the well-studied image-based 2D bounding-box detection.
    - Has limitation for 3D since the orientation for 3D is not fixed
- Proposal:
  - Idea: Represent, detect, track 3D objects as points
  - CenterPoint:
    - Firstly, detect centers of objects using a keypoint detector and regresses to other attributes, including 3D size, 3D orientation, and velocity.
    - Secondly, refines the estimations using additional point features on the object
  - 3D object tracking simplifies to greedy closest-point matching
- Result:
  - State-of-the-art performance on the **nuScenes** benchmark for both 3D detection and tracking
    - 65.5 NDS and 63.8 AMOTA for a single model
  - Outperform all previous single model method by a large margin and ranks first among all LiDAR-only submission on **Waymo Open Dataset**

INTRODUCTION

- Background:
  - 3D detection on point-clouds offers a series of interesting challenges:
    - Point-clouds are sparse, and most regions of 3D space are without measurements
    - The resulting output, 3D box, is often not well aligned with any global coordinate frame
    - 3D objects come in a wide range of sizes, shapes, and aspect ratios
  - Issue: axis-aligned 2D box is a poor proxy of a free-form 3D object
    - Potential solution: classify a different template (anchor) for each object orientation
      - But this unnecessarily increases the computational burden and may introduce a large number of false-positive detections
  - Main Challenge: linking up the 2D and 3D domains in representing objects
- Proposal:
  - Representing objects as points greatly simplifies 3D recognition.
- Advantages:
  - Points have no intrinsic orientation, which reduces the object detector's search space while allowing the backbone to learn the rotational invariance of objects and rotational equivariance of their relative rotation
  - Simplifies downstream tasks such as tracking
  - Point-based feature extraction is much faster than previous refinement module.

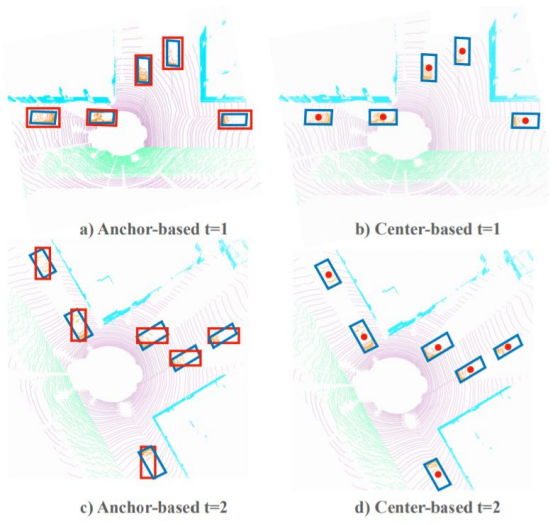

Figure 1: We present a center-based framework to represent, detect and track objects. Previous anchor-based methods use axis-aligned anchors with respect to ego-vehicle coordinate. When the vehicle is driving in straight roads, both anchor-based and our center-based method are able to detect objects accurately (top). However, during a safety-critical left turn (bottom), anchor-based methods have difficulty fitting axis-aligned bounding boxes to rotated objects. Our center-based model accurately detect objects through rotationally invariant points. Best viewed in color.

- **2D CenterNet:**
  - Rephrase object detection as keypoint estimation.
  - It takes an input image and predicts a w x h heatmap $\hat{Y} \in [0, 1]^{w \times h \times K}$ for each of K classes
  - Each local max in the output heatmap corresponds to the center of a detected object.
  - To retrieve a 2D box, CenterNet regresses to a size map $\hat{S} \in R^{w \times h \times 2}$ shared between all categories.
  - For each detected object , the size-map stores its width and height at the center location.
  - The architecture uses a standard fully convolutional image backbone and adds a desne prediction head on top
  - In training: CenterNet learns to predict heatmaps with rendered Gaussian kernels at each annotated object center $\mathbf{q}_i$ for each class $c_i \in \{1 \dots K\}$
    - Regress to object size S at the center of the annotated bounding box.
    - Also regress to a local offset, $\hat{O}$, to make up for quantization errors introduced by the striding
  - In testing, Non-Maxima suppression may be warranted, as the detector produce K heatmaps and dense class-agnostic regression maps.
- **3D Detection:**
  - Say a point-cloud is $P = \{(x,y,z,r)_i\}$, 3D object detection aims to predict a set of 3D object bounding box $B = \{b_k\}$ in the bird eye view from this point-cloud, where b = (u, v, d, w, l, h, α).
    - Center location relative to the objects ground plane: (u, v, d). 3D size (w, l, h), rotation yaw: α
  - Modern 3D object detectors use a 3D encoder that quantizes the point-cloud into regular bins
  - A point-base network then extracts features for all points inside a bin
  - The 3D encoder then pools these features into its primary feature representation.
  - Anchor-based detector have difficulty fitting axis-aligned 2D box to 3D object due to various size

CENTERPOINT

- Center-based detection head but rely on existing 3D backbone (VoxelNet or PointPillars)

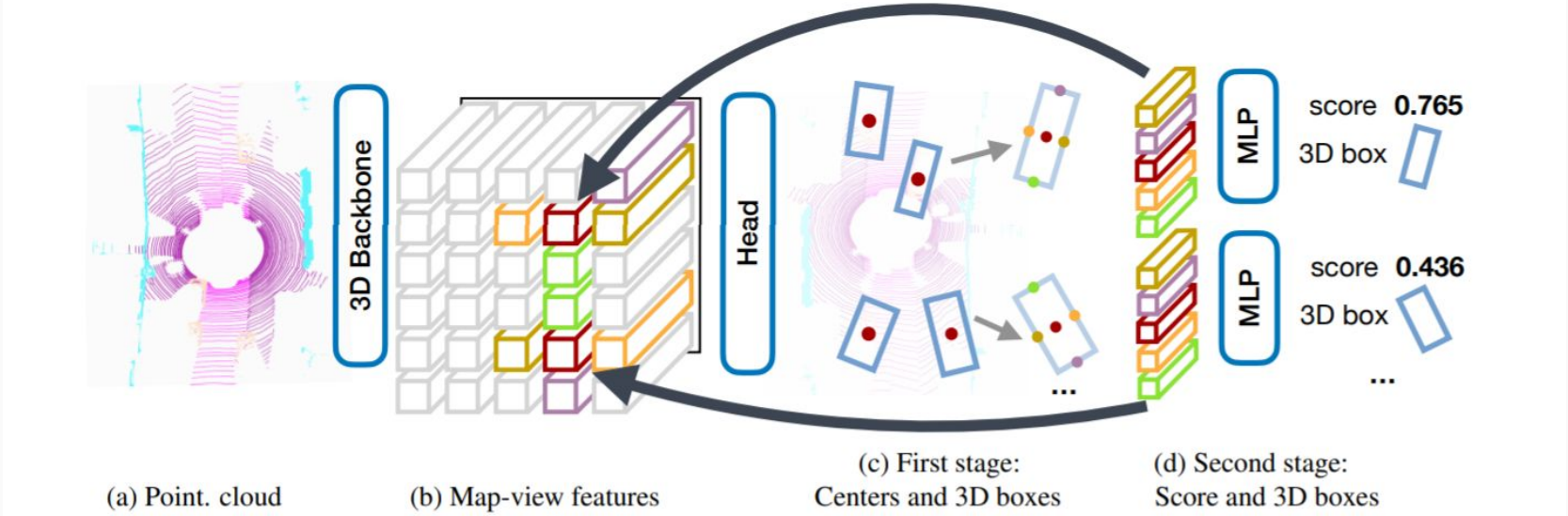

Figure 2: Overview of our CenterPoint framework. We rely on a standard 3D backbone that extracts map-view feature representation from Lidar point-clouds. Then, a 2D CNN architecture detection head finds object centers and regress to full 3D bounding boxes using center features. This box prediction is used to extract point features at the 3D centers of each face of the estimated 3D bounding box, which are passed into MLP to predict an IoU-guided confidence score and box regression refinement. Best viewed in color.

CENTERPOINT (CONT')

- **Overview**:
  - Let $M \in R^{W \times H \times F}$ be the output of the 3D backbone
  - First stage of CenterPoint predicts a class-specific heatmap, object size, a sub-voxel location refinement, rotation, and velocity
  - All outputs are dense predictions
  - CenterPoint combines all heatmap and regression losses in one common objective and jointly optimize them
    - It simplifies and improves previous anchor-based 3D detectors.
  - However, all properties of the object are currently inferred from the object's center-feature, which may not contain sufficient information for accurate object localization (which will be improved at the second stage with a light-weight point-feature extractors)
- **Center heatmap head**:
  - Goal: produce a heatmap peak at the center location of any detected object
  - This head produces a K-channel heatmap $\hat{Y}$, one channel for each of K classes
  - In training: targets 2D Gaussian produced by the projection of 3D centers of annotated bounding boxes into the map-view
  - Objects in a top-down map view are sparser than in an image.
  - From CenterNet's result, we know: most locations are considered as background in a very sparse supervisory signal
    - To counteract this, we increase the positive supervision for the target heatmap Y by enlarging the Gaussian peak rendered at each ground truth object center. I.e. set Gaussian radius to $\sigma = \max(f(wl), \tau)$, where τ is the smallest allowable Gaussian radius, and f is a radius function defined in CornerNet.
    - → model gets denser supervision from nearby pixels

CENTERPOINT (CONT')

- **Regression Heads:**
  - We store several object properties at center-features of objects:
    - A sub-voxel location refinement $o \in R^2$
      - Reduces the quantization error from voxelization and striding of the backbone network
    - Height-above-ground $h_g \in R$
      - Helps localize the object in 3D and adds the missing elevation information removed by map-view projection
    - 3D size $s \in R^3$
    - Yaw rotation angle $(\sin(\alpha), \cos(\alpha)) \in R^2$
  - Combined with box size, theses regression heads provides the full state information of the 3D bounding box
  - Each output uses its own head.
  - During training, only ground truth centers are supervised using an L1 regression loss.
  - During inference, we extract all properties by indexing into dense regression head outputs at each object's peak location.
- **Velocity Head and Tracking:**
  - To track objects through time, we learn to predict a two-dimensional velocity estimation $v \in R^2$ for each detected object as an additional regression output.
    - It requires two input map views (previous & current) → predicts the difference in position of two frames
    - Also supervised using L1 loss at ground truth object's location at the current time-step.

CENTERPOINT (CONT')

- **Velocity Head and Tracking (cont'):**
  - At inference time, we use this offset to associate current detections to past ones in a greedy fashion.
    - We project object centers in the current frame back to the previous frame by applying the negative velocity estimate
    - Then matching them to the tracked objects by closest distance matching

TWO-STAGE CENTERPOINT

- Recall:
  - *all properties of the object are currently inferred from the object's center-feature, which may not contain sufficient information for accurate object localization*
    - E.g., in autonomous driving, the sensor often only sees the side of the object, but not its center
  - A light-weight point-feature extractor can be used to improve the result
- First stage remains unchanged
- Second stage extracts additional point-features from the output of the backbone:
  - Extract one point-feature from the 3D center of each face of the predicted bounding box.
  - Note that the bounding box center, top, bottom face centers all project to the same point in map-view
    - Thus, only consider the four outward-facing box-faces together with the predicted object center.
    - For each point, we extract a feature using bilinear interpolation from the backbone map-view output M
    - Next, we concatenate the extracted point-features and pass then through an MLP
  - The second stage predicts a class-agnostic confidence score and box refinement on top of one-stage CenterPoint's prediction results.
  - For box regression, the model predicts a refinement on top of first stage proposals, and we train the model with L1 loss.
- The two-stage CenterPoint simplifies and accelerates previous two-stage 3D detecotrs that use expensive PointNet-based feature extractor and RoIAlign operations

ARCHITECTURE

- **First stage:**
  - All first-stage outputs share a first 3x3 convolutional layer, Batch Normalization, and ReLU.
  - Each output then uses its own branch of two 3x3 convolutions separated by a batch norm and ReLU
- **Second Stage:**
  - Second-stage uses a shared two-layer MLP, with a batch norm, ReLU, and Dropout with a drop rate of 0.3, followed by two branches of three fully-connected layers
    - One for confidence score
    - Another one for box regression prediction

RESULTS

Difficulty	Method	Vehicle mAP	Vehicle mAPH	Pedestrian mAP	Pedestrian mAPH
Level 1	StarNet [36]	61.5	61.0	67.8	59.9
	PointPillars [28]	63.3	62.8	62.1	50.2
	PPBA [36]	67.5	67.0	69.7	61.7
	RCD [5]	72.0	71.6	-	-
	Ours	**80.2**	**79.7**	**78.3**	**72.1**
Level 2	StarNet [36]	54.9	54.5	61.1	54.0
	PointPillars [28]	55.6	55.1	55.9	45.1
	PPBA [36]	59.6	59.1	63.0	55.8
	RCD [5]	65.1	64.7	-	-
	Ours	**72.2**	**71.8**	**72.2**	**66.4**

Table 1: State-of-the-art comparisons for 3D detection on Waymo test set. We show the mAP and mAPH for both level 1 and level 2 benchmarks.

Difficulty	Method	MOTA↑ Vehicle	MOTA↑ Ped.	MOTP↓ Vechile	MOTP↓ Ped.
Level 1	AB3D [48, 53]	42.5	38.9	18.6	34.0
	Ours	**62.6**	**58.3**	**16.3**	**31.1**
Level 2	AB3D [48, 53]	40.1	37.7	18.6	34.0
	Ours	**59.4**	**56.6**	**16.4**	**31.2**

Table 3: State-of-the-art comparisons for 3D tracking on Waymo test set. We show MOTA, and MOTP. ↑ is for higher better and ↓ is for lower better.

Method	AMOTA↑	FP↓	FN↓	IDS↓
AB3D [53]	15.1	**15088**	75730	9027
Chiu et al. [10]	55.0	17533	33216	950
Ours	**63.8**	18612	**22928**	**760**

Table 4: State-of-the-art comparisons for 3D tracking on nuScenes test set. We show AMOTA, the number of false positives (FP), false negatives (FN), id switches (IDS), and per-category AMOTA. ↑ is for higher better and ↓ is for lower better.

Method	mAP↑	NDS↑	PKL↓
WYSIWYG [23]	35.0	41.9	1.14
PointPillars [28]	40.1	55.0	1.00
CVCNet [7]	55.3	64.4	0.92
PointPainting [49]	46.4	58.1	0.89
PMPNet [62]	45.4	53.1	0.81
SSN [68]	46.3	56.9	0.77
CBGS [67]	52.8	63.3	0.77
Ours	**58.0**	**65.5**	**0.69**

Table 2: State-of-the-art comparisons for 3D detection on nuScenes test set. We show the nuScenes detection score (NDS), and mean Average Precision (mAP).

Encoder	Method	Vehicle	Pedestrain	mAPH
VoxelNet	Anchor-based	66.1	54.4	60.3
	Center-based	**66.5**	**62.7**	**64.6**
PointPillars	Anchor-based	64.1	50.8	57.5
	Center-based	**66.5**	**57.4**	**62.0**

Table 5: Comparison between anchor-based and center-based methods for 3D detection on Waymo validation. We show the per-calss and average LEVEL_2 mAPH.

Encoder	Method	mAP	NDS
VoxelNet	Anchor-based	52.6	63.0
	Center-based	**56.4**	**64.8**
PointPillars	Anchor-based	46.2	59.1
	Center-based	**50.3**	**60.2**

Table 6: Comparison between anchor-based and center-based methods for 3D detection on nuScenes validation. We show mean average precision (mAP) and nuScenes detection score (NDS).

	Vehicle			Pedestrian		
Rel. yaw	0°-15°	15°-30°	30°-45°	0°-15°	15°-30°	30°-45°
# annot.	81.4%	10.5%	8.1%	71.4%	15.8%	12.8%
Anchor-based	67.1	**47.7**	45.4	55.9	32.0	26.5
Center-based	**67.8**	46.4	**51.6**	**64.0**	**42.1**	**35.7**

Table 7: Comparison between anchor-based and center based methods for detecting objects of different heading angles. The ranges of the rotation angle and their corresponding portion of objects are listed in line 2 and line 3. We show the LEVEL_2 mAPH for both methods on the Waymo validation.

Method	Vehicle			Pedestrian		
	small	medium	large	small	medium	large
Anchor-based	58.5	**72.8**	64.4	29.6	60.2	60.1
Center-based	**59.0**	72.4	**65.4**	**38.5**	**69.5**	**69.0**

Table 8: Effects of object size for the performance of anchor-based and center-based methods. We show the per-class LEVEL_2 mAPH for objects in different size range: small 33%, middle 33%, and large 33%

Encoder	Method	Vehicle	Ped.	$T_{proposal}$	T_{refine}
VoxelNet	First Stage	66.5	62.7	71ms	-
	+ Box Center	68.0	64.9	71ms	5ms
	+ Surface Center	**68.3**	65.3	71ms	6ms
	Dense Sampling	68.2	**65.4**	71ms	8ms
PointPillars	First Stage	66.5	57.4	56ms	-
	+ Box Center	67.3	57.4	56ms	6ms
	+ Surface Center	**67.5**	57.9	56ms	7ms
	Dense Sampling	67.3	**57.9**	56ms	8ms

Table 9: Compare 3D LEVEL_2 mAPH for VoxelNet and PointPillars encoders using single stage, two stage with 3D center features, and two stage with 3D center and surface center features on Waymo validation.

Methods	Vehicle	Pedestrian	Runtime
BEV Feature	68.3	65.3	77ms
w/ VSA [44]	68.3	65.2	98ms
w/ RBF Interpolation [20, 41]	**68.4**	**65.7**	89ms

Table 10: Ablation studies of different feature components for two stage refinement module. VSA stands for Voxel Set Abstraction, the feature aggregation methods used in PV-RCNN [44]. RBF uses a radial basis function to interpolate 3 nearest neighbors. We compare bird-eye view and 3D voxel features using LEVEL_2 mAPH on Waymo validation.

RESULTS

Difficulty	Method	Vehicle mAP	Vehicle mAPH	Pedestrian mAP	Pedestrian mAPH
Level 1	DOPS [35]	56.4	-	-	
	PointPillars [28]	56.6	-	59.3	-
	PPBA [36]	62.4	-	66.0	-
	MVF [65]	62.9	-	65.3	-
	Huang et al. [24]	63.6	-	-	
	AFDet [14]	63.7	-	-	
	CVCNet [7]	65.2	-	-	
	Pillar-OD [52]	69.8	-	72.5	-
	PV-RCNN [44]	74.4	73.8	61.4	53.4
	CenterPoint-Pillar(ours)	76.1	75.5	76.1	65.1
	CenterPoint-Voxel(ours)	**76.7**	**76.2**	**79.0**	**72.9**
Level 2	PV-RCNN [44]	65.4	64.8	53.9	46.7
	CenterPoint-Pillar(ours)	68.0	67.5	68.1	57.9
	CenterPoint-Voxel(ours)	**68.8**	**68.3**	**71.0**	**65.3**

Table 11: State-of-the-art comparisons for 3D detection on Waymo validation.

Detector	Tracker	AMOTA↑	AMOTP↓	T_{track}	T_{tot}
CenterPoint-Voxel	Point	**63.7**	**0.606**	1ms	62ms
CBGS [67]	Point	59.8	0.682	1ms	$> 182ms$
CenterPoint-Voxel	M-KF	60.0	0.765	73ms	135ms
CBGS [67]	M-KF	56.1	0.800	73ms	>254ms

Table 12: Ablation studies for 3D tracking on nuScenes validation. We show combinations of different detectors and trackers. CenterPoint-* are our detectors. Point is our proposed tracker. M-KF is short for Mahalanobis distance-based Kalman filter, as is used in the last challenge winner Chiu et al. [10]. T_{track} denotes tracking time and T_{tot} denotes total time for both detection and tracking.

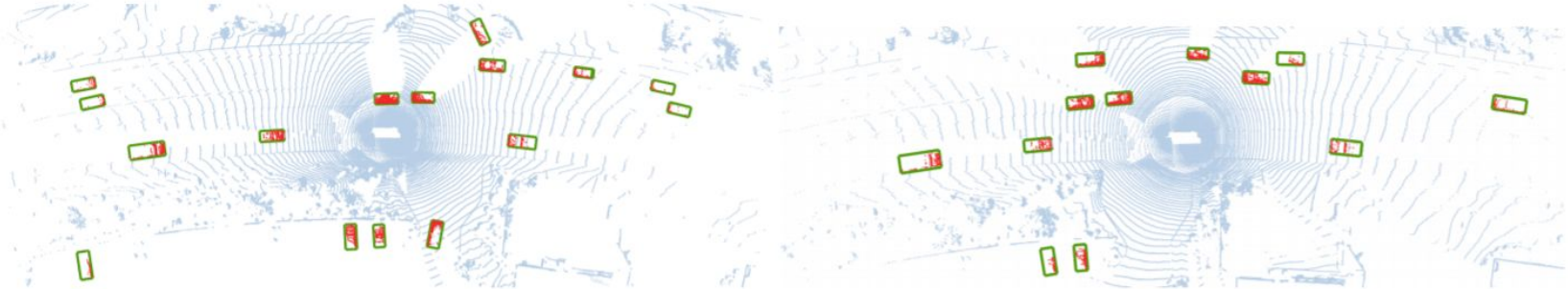

Figure 3: Example qualitative results of CenterPoint on the Waymo validation. We show the raw point-cloud in blue, our detected objects in green bounding boxes, and Lidar points inside bounding boxes in red. Best viewed on screen.