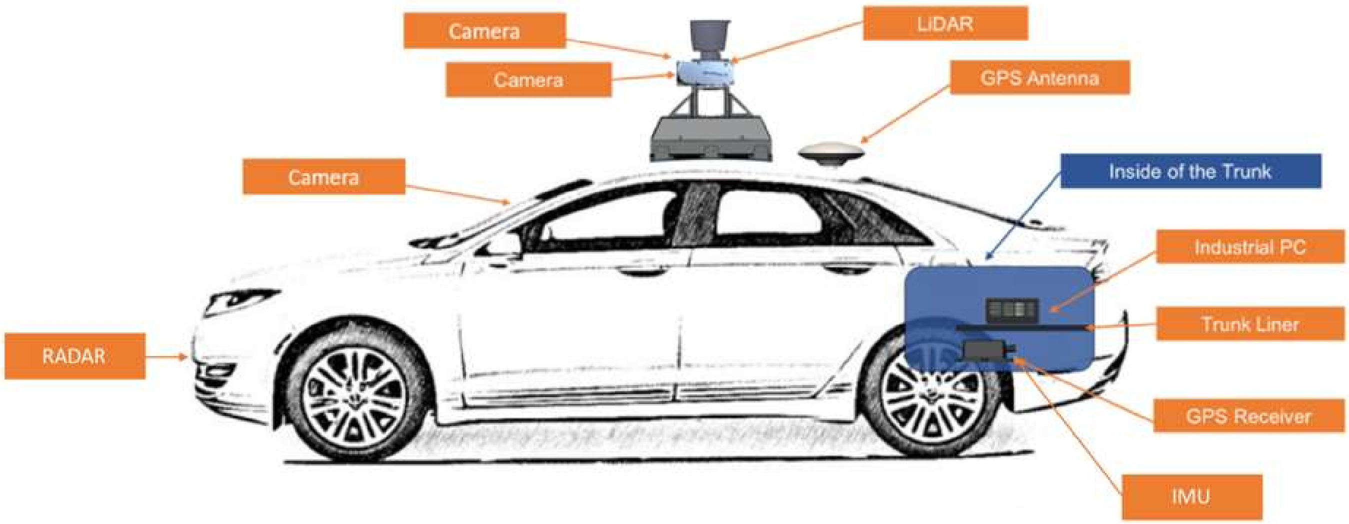
Solving Non-trivial problems in Autonomous Driving
Using Efficient and Reliable AI Algorithms

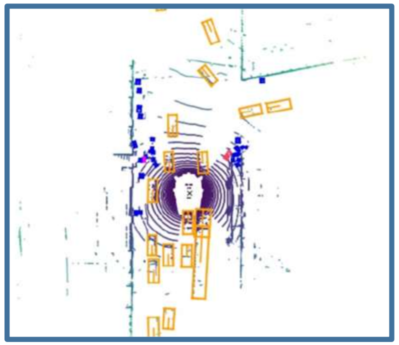
Yuexin Ma
ShanghaiTech University

Autonomous Car

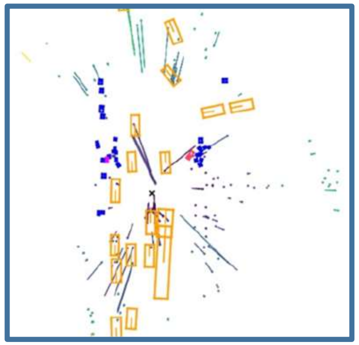


Camera

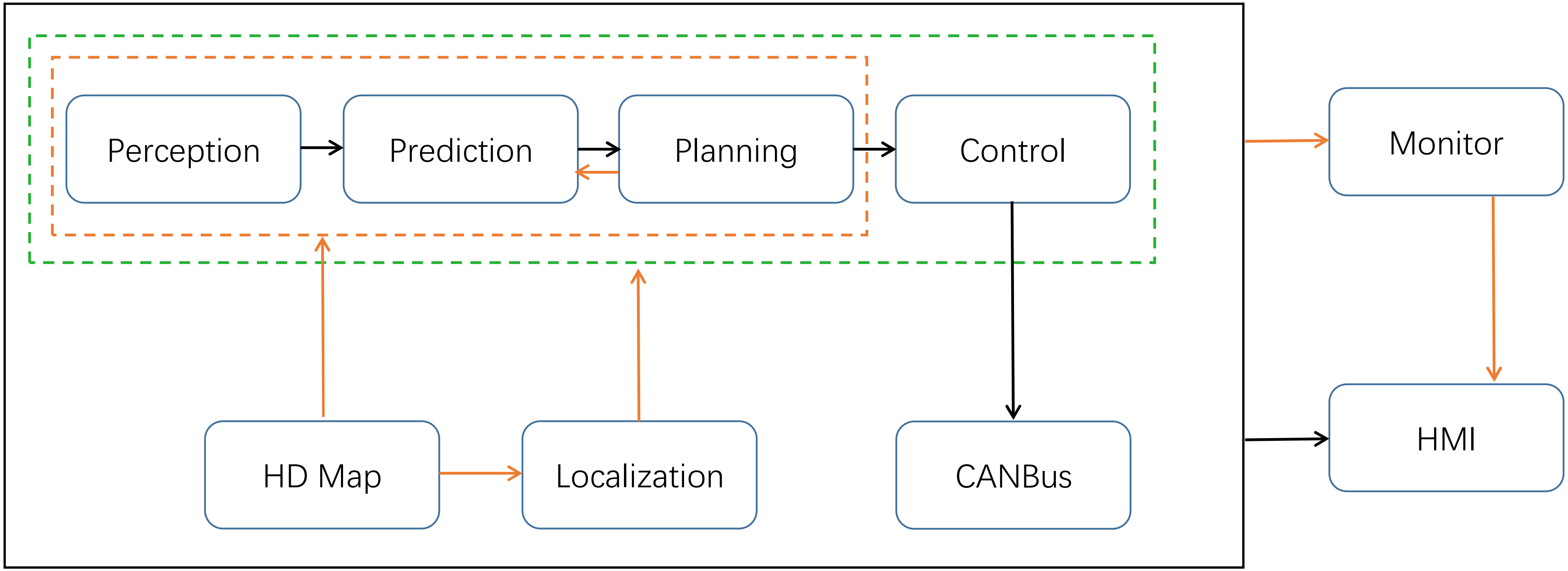
LiDAR



Radar



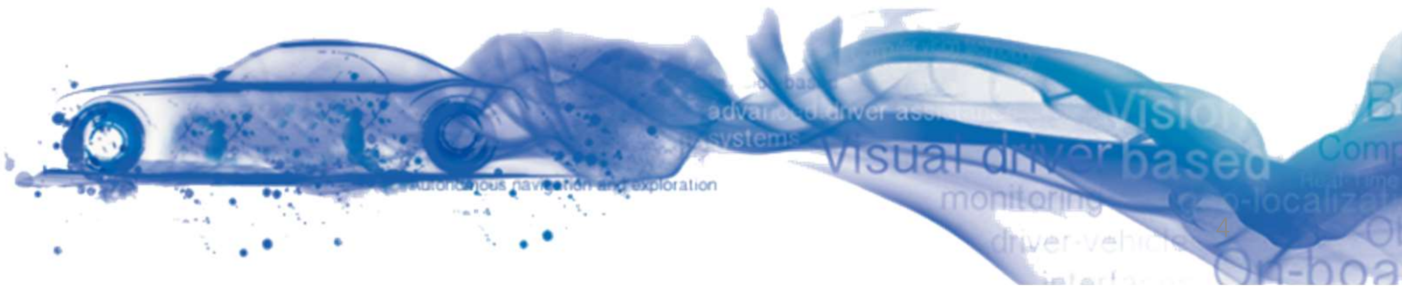
Autonomous Driving System



Key: Data lines Control lines

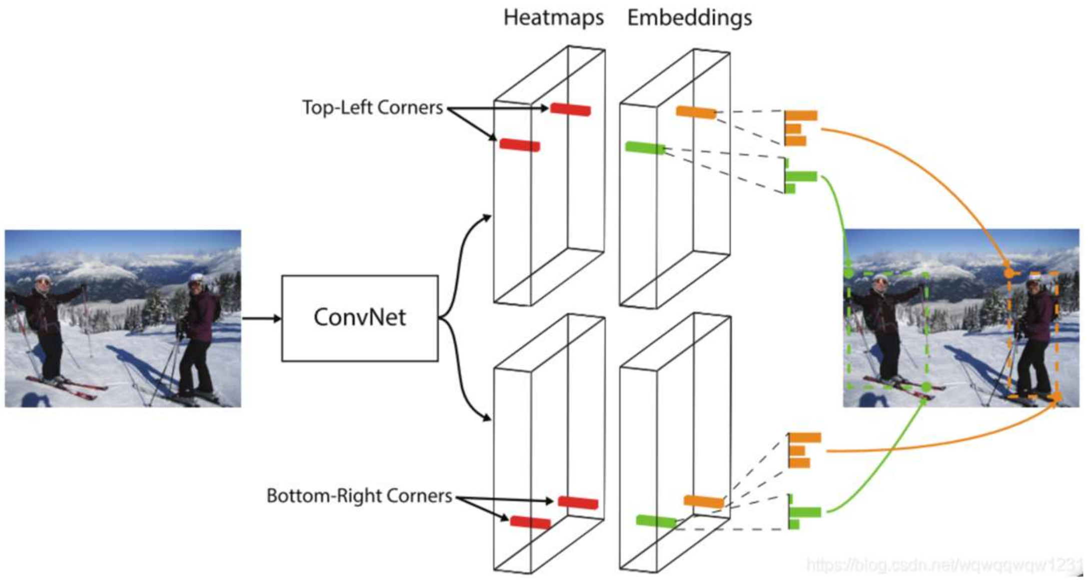
Research for Autonomous Driving

Perception

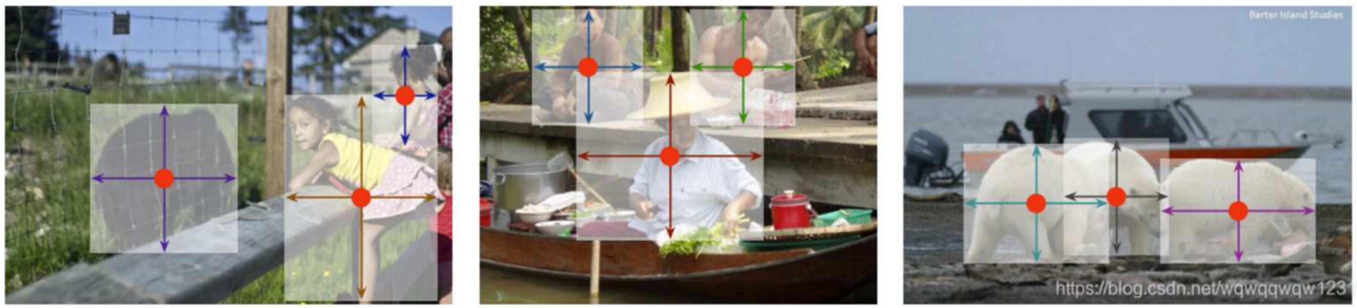


2D Detection

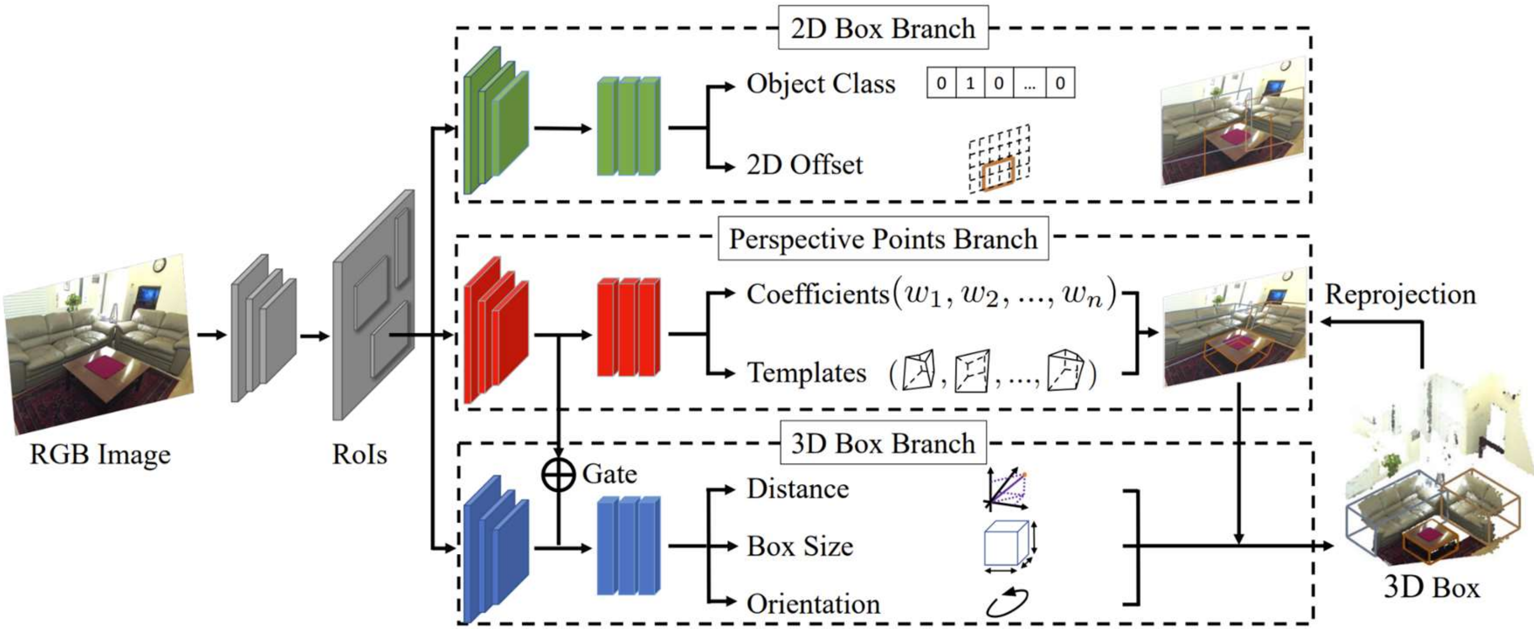
CornerNet



CenterNet

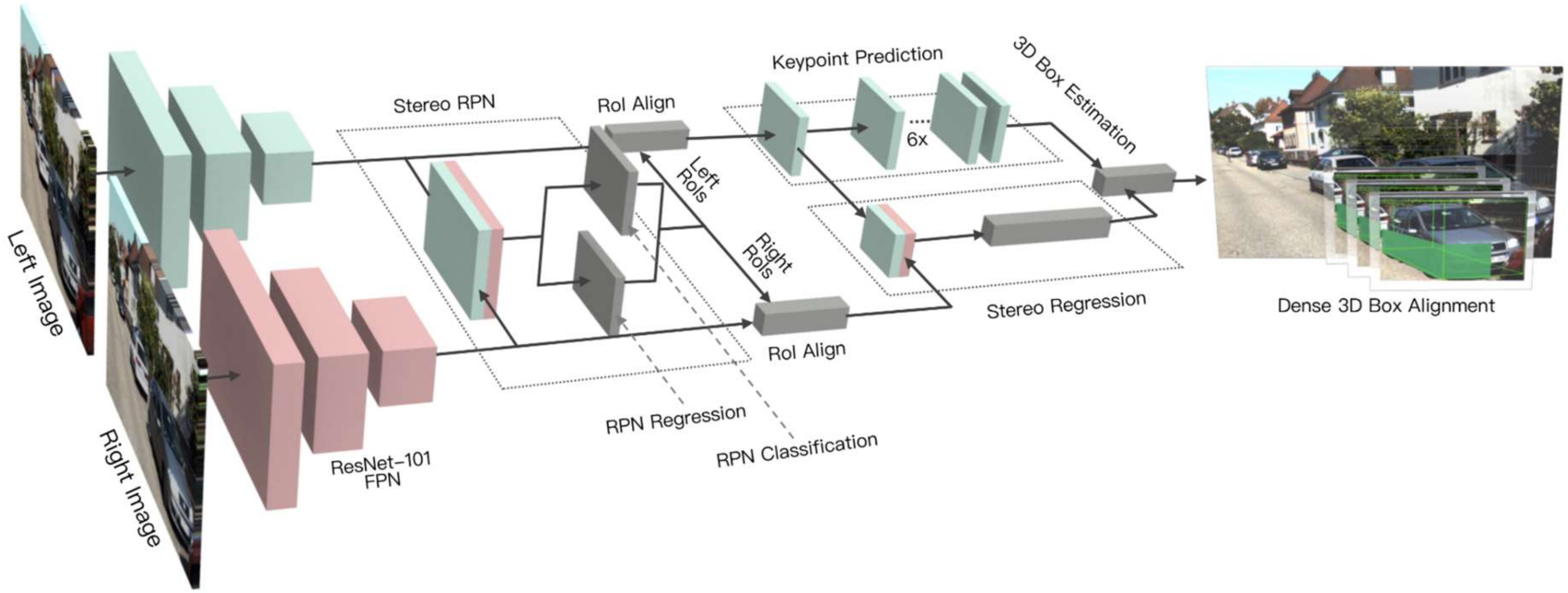


3D Detection from Single Image

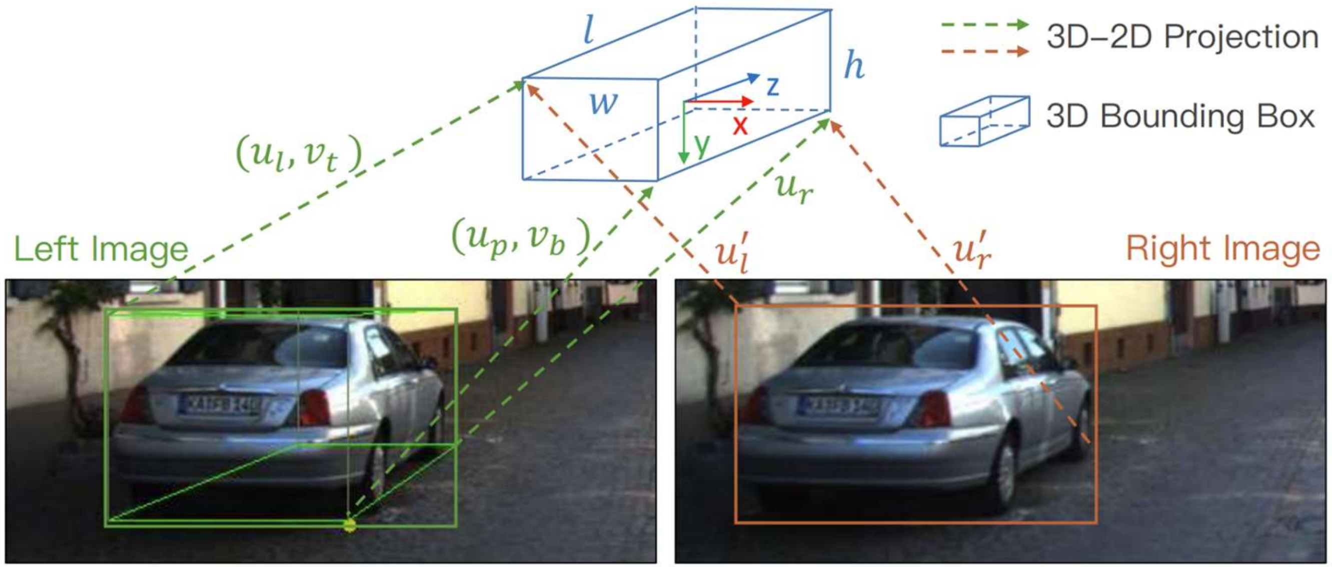


3D Detection from Stereo Cameras

Stereo R-CNN based 3D Object Detection for Autonomous Driving

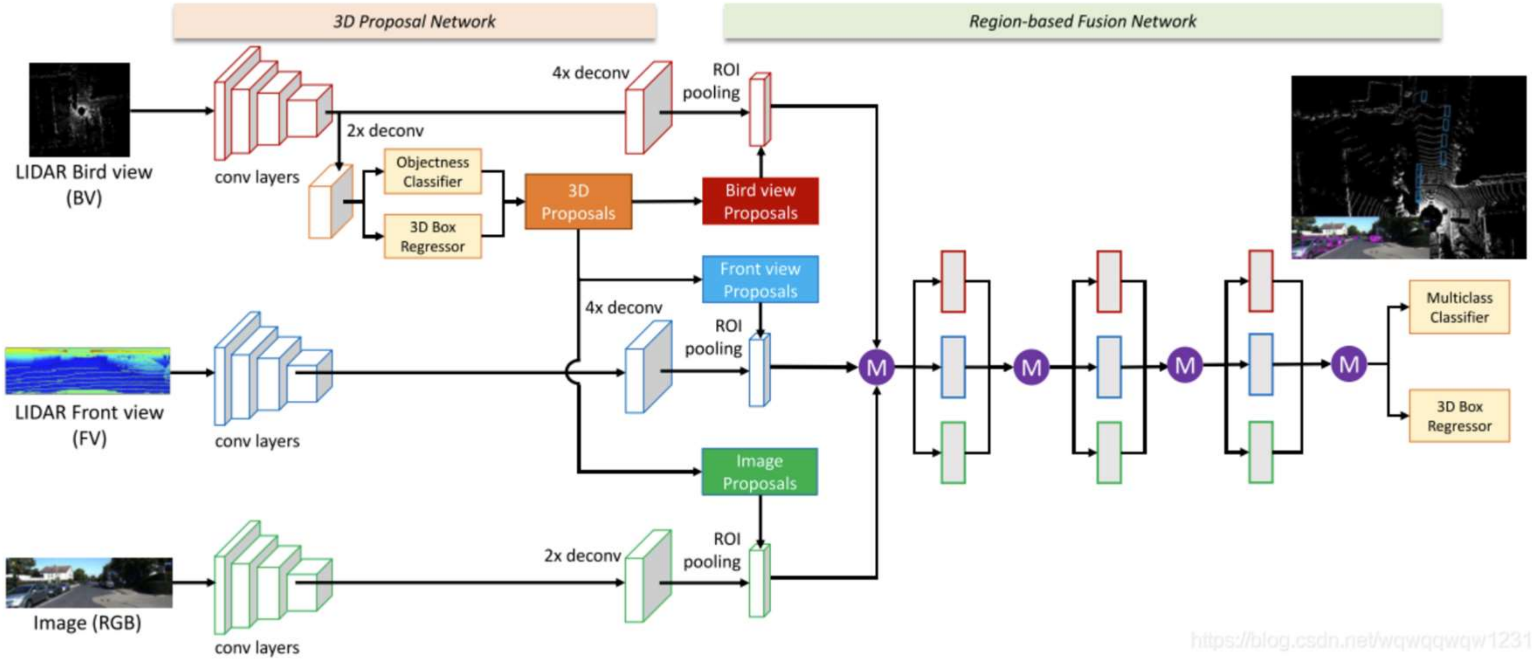


3D Detection from Stereo Cameras



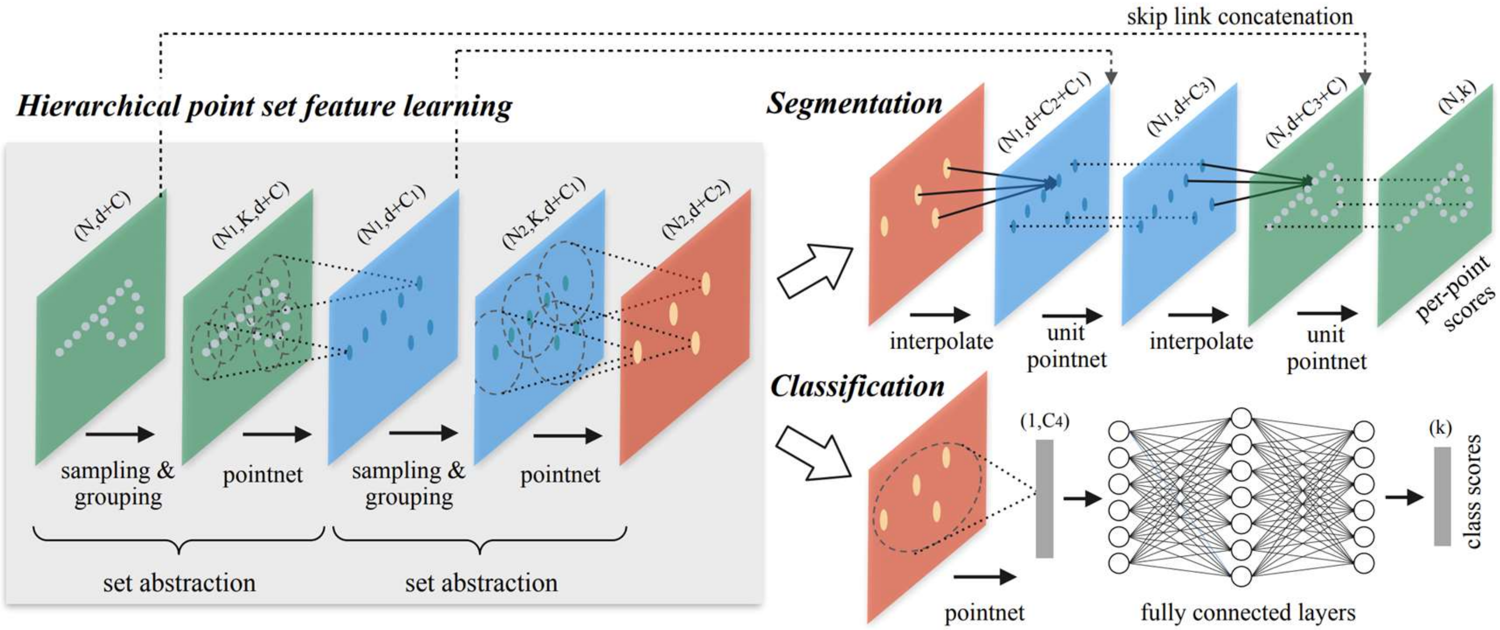
3D Detection from LiDAR Point Cloud

MVNet



3D Detection from LiDAR Point Cloud

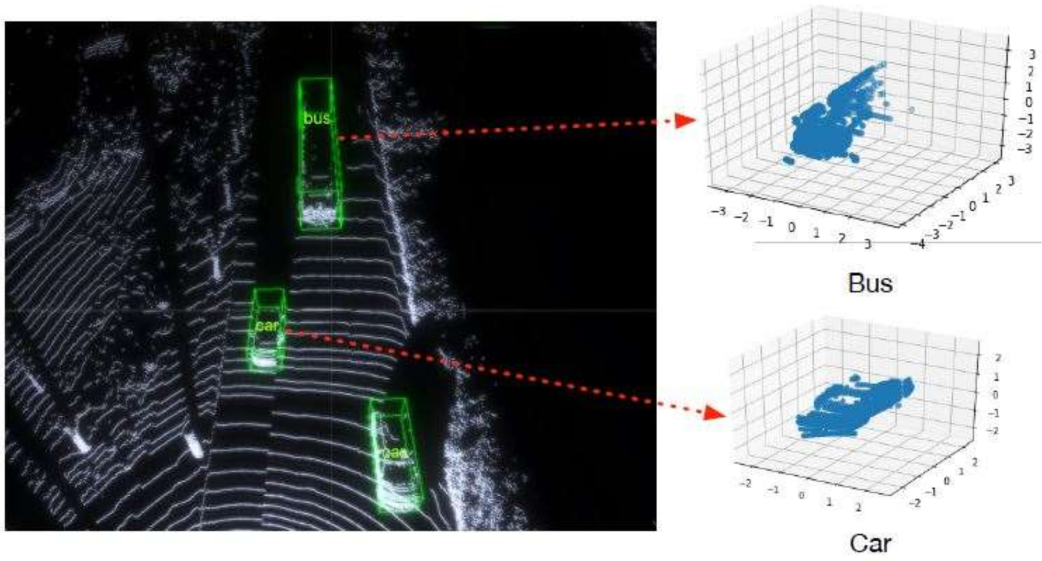
PointNet++



SSN: Shape Signature Networks for Multi-class Object Detection from Point Clouds

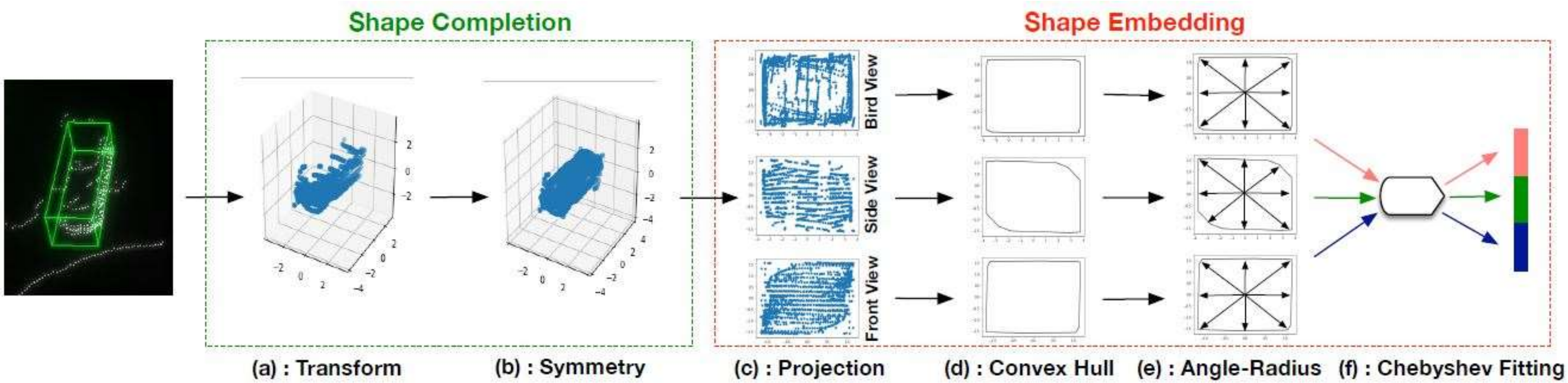
ECCV, 2020

- Accurate and robust perception is the **foundation** for autonomous driving.
- The mainstream 3D detection frameworks focus on the single-category detection.
- Point cloud is **unstructured**, **sparse** and **noisy** and **lacks texture and appearance**.



Shape and scale vary with categories.

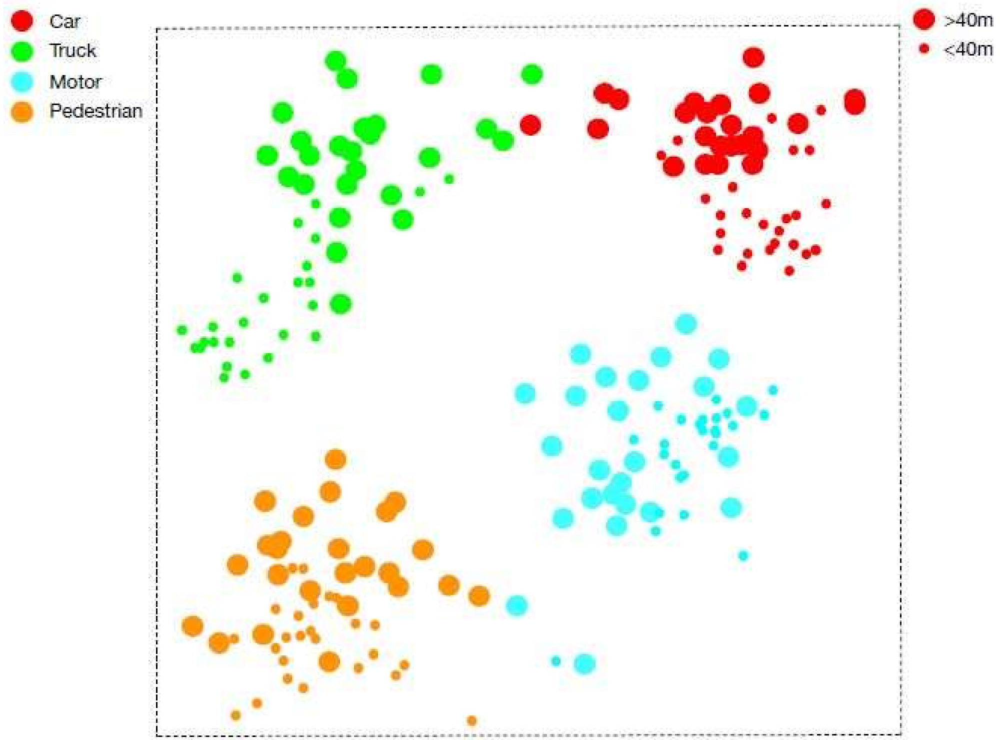
How to build shape encoding from sparse and noise point cloud?



Shape Signature

- **Compact** (effctive and short as the objective)
- **Robust** (robust against the sparsity and noise)

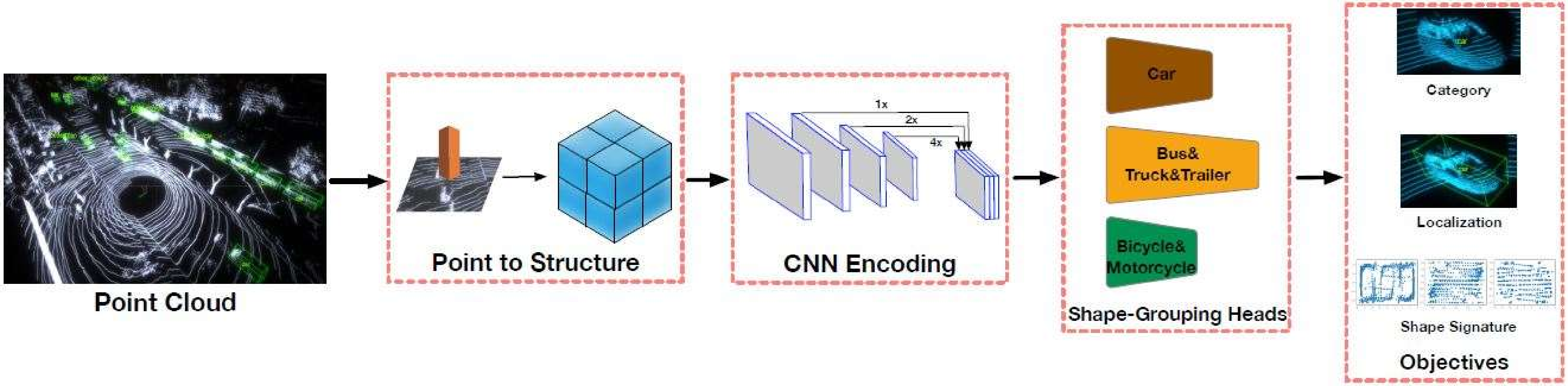
Distribution of our shape signature via TSNE



- Well separate the shape distribution across different categories.
- Keep the shape distribution consistent (not same).

We sample 50 instances for each category, where 25 of them are with distance < 40 meters and others are with distance > 40 meters.

SSN: Shape Signature Networks



Multi-task Objectives $\mathcal{L} = \beta_1 \mathcal{L}_{cls} + \beta_2 \mathcal{L}_{loc} + \beta_3 \mathcal{L}_{shape}$

1. multi-class classiffication $\mathcal{L}_{cls} = -\alpha_t (1 - p_t)^{\gamma} \log(p_t)$
2. localization regression $\mathcal{L}_{loc} = \text{SmoothL1}(\triangle b)$
3. shape vector regression $\mathcal{L}_{shape} = \text{SmoothL1}(\mathbb{S})$

Results

Table 1. Results of multi-class 3D detection on nuScenes dataset. Bold-face and underline numbers denote the best and second-best respectively.

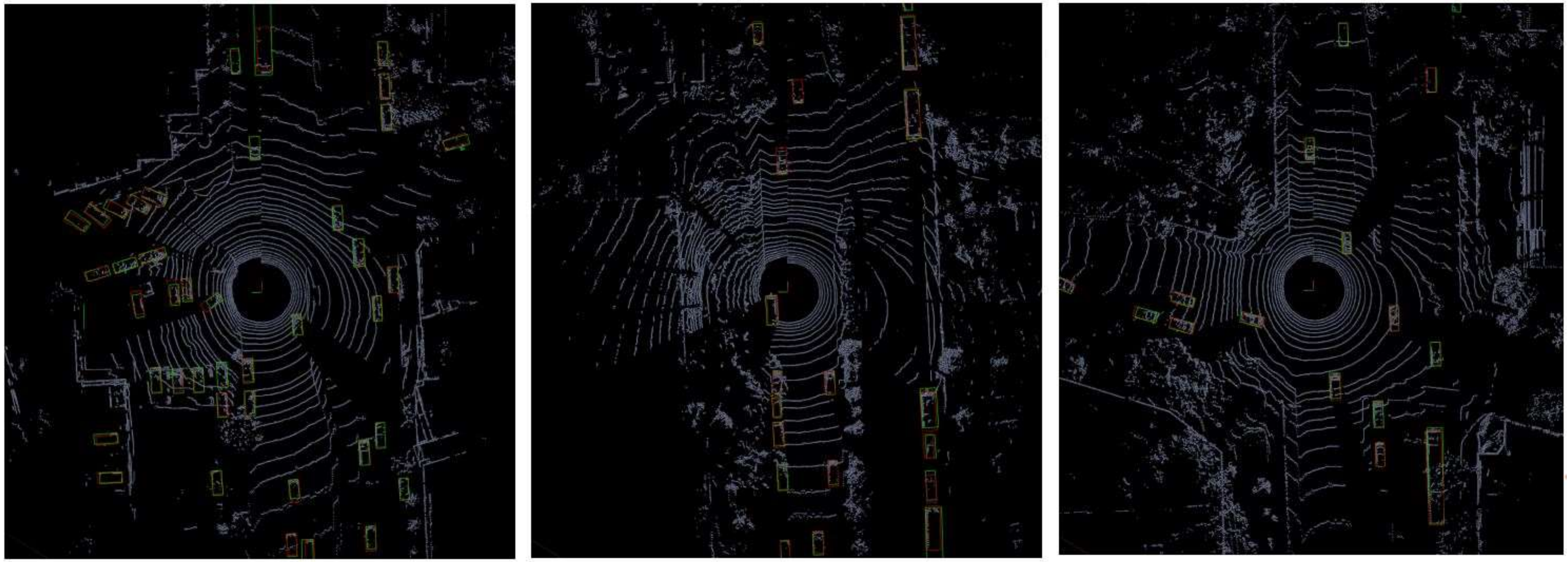
Methods	Modality	Car	Truck	Bus	Trail	CV	Ped	MC	Bicy	TC	Bar	mAP	NDS
Mono [29]	RGB	47.8	22.0	18.8	17.6	7.4	37.0	29.0	**24.5**	48.7	51.1	30.4	38.4
Second [34]	Lidar	73.1	25.2	30.5	31.5	8.5	59.3	21.7	4.9	18.0	43.3	31.6	46.8
PP [13]	Lidar	68.4	23.0	28.2	23.4	4.1	59.7	27.4	1.1	30.8	38.9	30.5	45.3
Painting [30]	Lidar&RGB	77.9	35.8	36.1	37.3	**15.8**	**73.3**	41.5	24.1	**62.4**	**60.2**	**46.4**	**58.1**
SSN	Lidar	**80.7**	**37.5**	**39.9**	**43.9**	14.6	72.3	**43.7**	20.1	54.2	56.3	46.3	56.9

Table 2. Results on Lyft dataset.

Methods	Modality	mAP-3D
Voxelnet [38]	Lidar	10.1
PointPillar [13]	Lidar	13.4
Second [34]	Lidar	13.0
SSN	Lidar	**17.9**

We got rank-1 on the leaderboard of Lyft for single model!

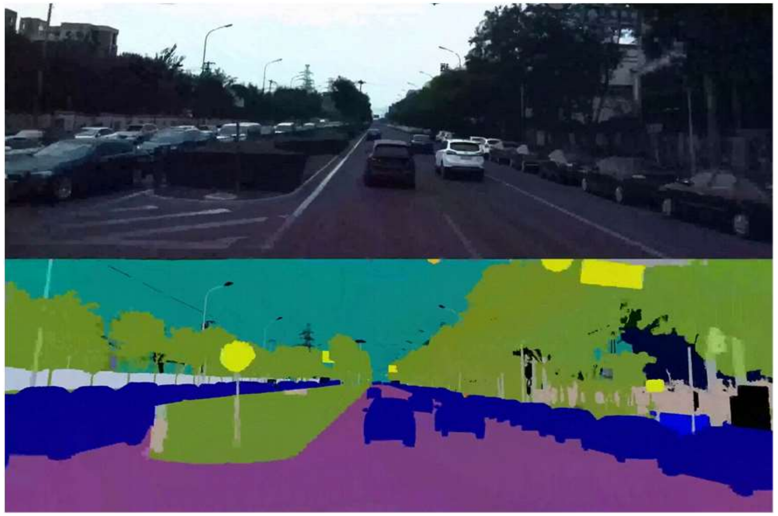
Qualitative analysis



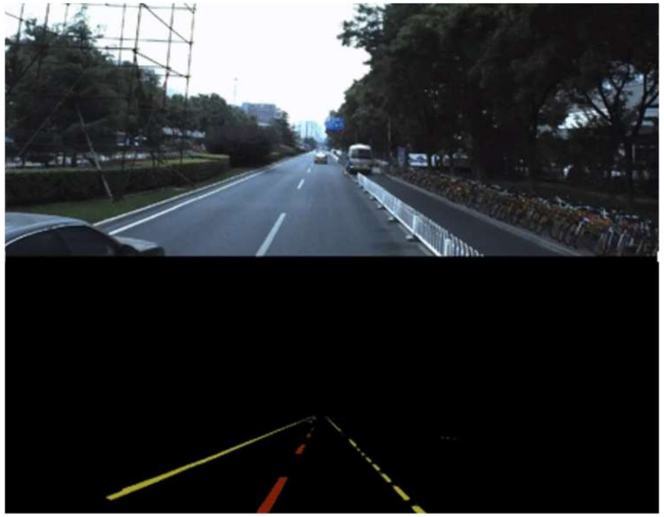
These point clouds are sampled from Lyft dataset. Green boxes are the ground truth and red boxes are the model's predictions.

Image-based Perception for Autonomous Driving

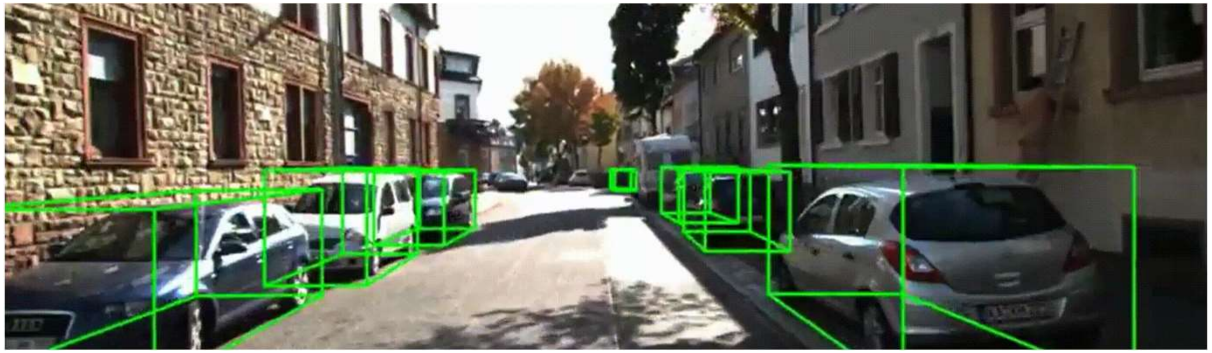
Scene Segmentation



Lane Segmentation

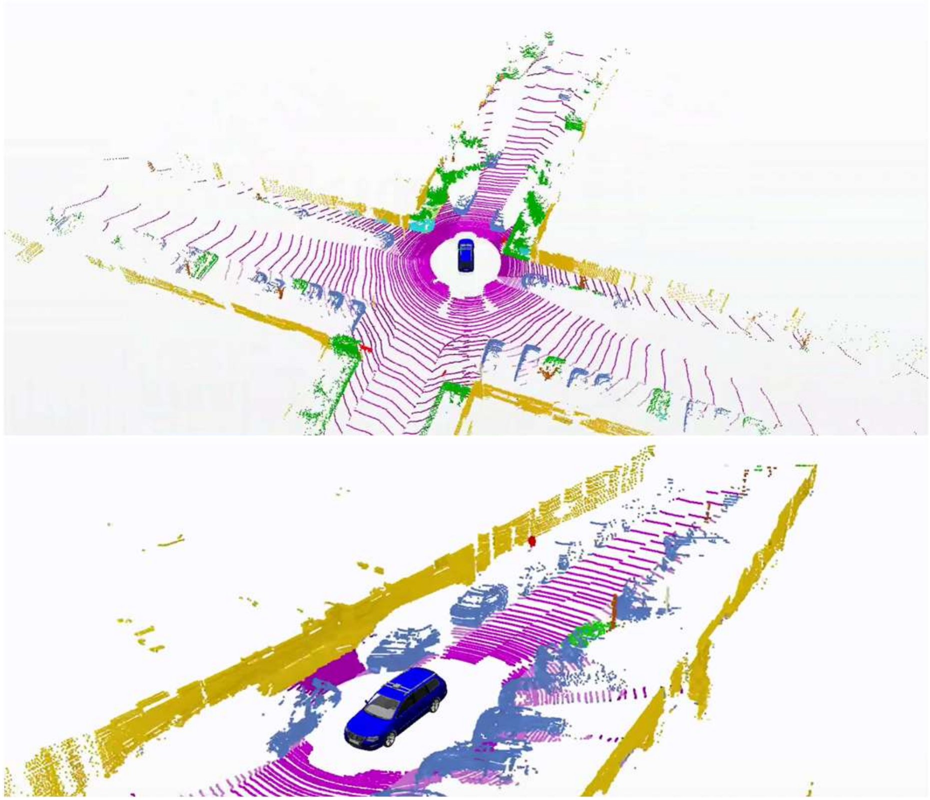


3D Detection

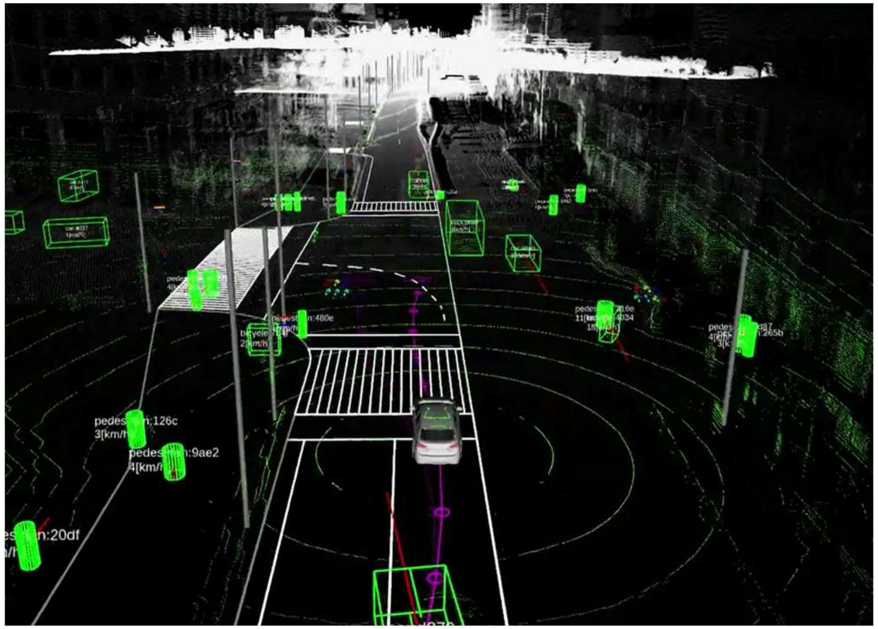


LiDAR-based Perception for Autonomous Driving

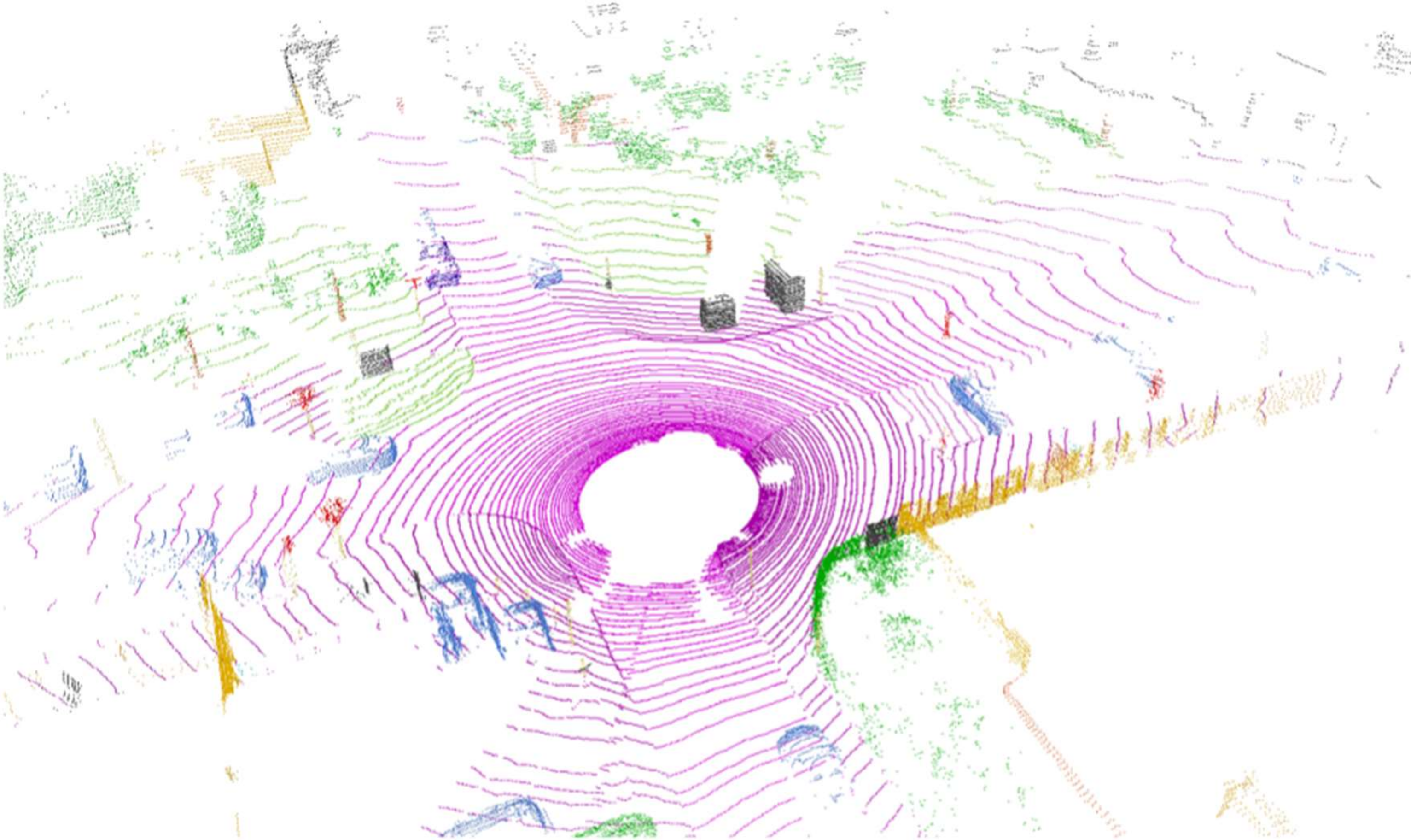
Segmentation



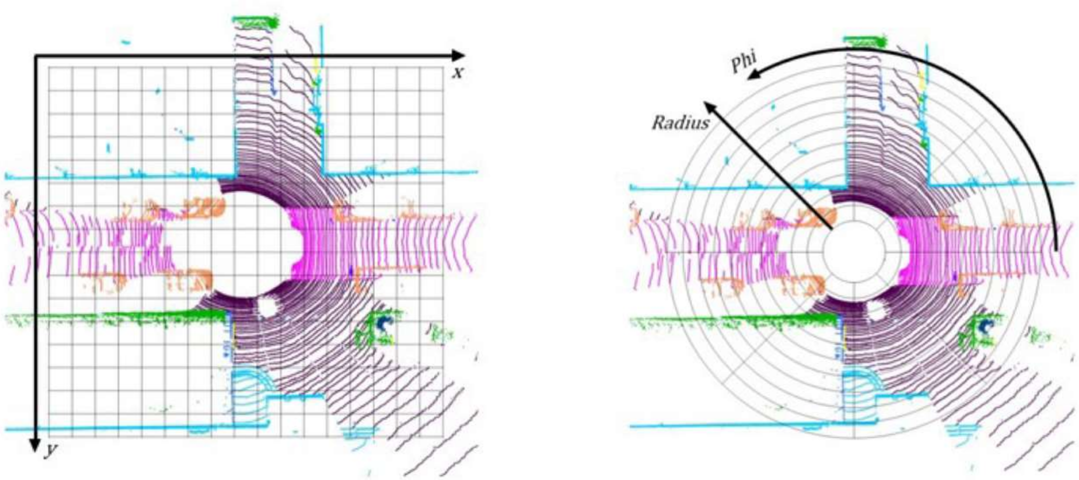
3D Detection



LiDAR Point Cloud



PolarNet Representation



(a) Cartesian BEV (b) Polar BEV

Figure 1. Two BEV quantization strategies

Zhang, Y., Zhou, Z., David, P., Yue, X., Xi, Z., & Foroosh, H. (2020). PolarNet: An Improved Grid Representation for Online LiDAR Point Clouds Semantic Segmentation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9598-9607.

PolarNet Representation

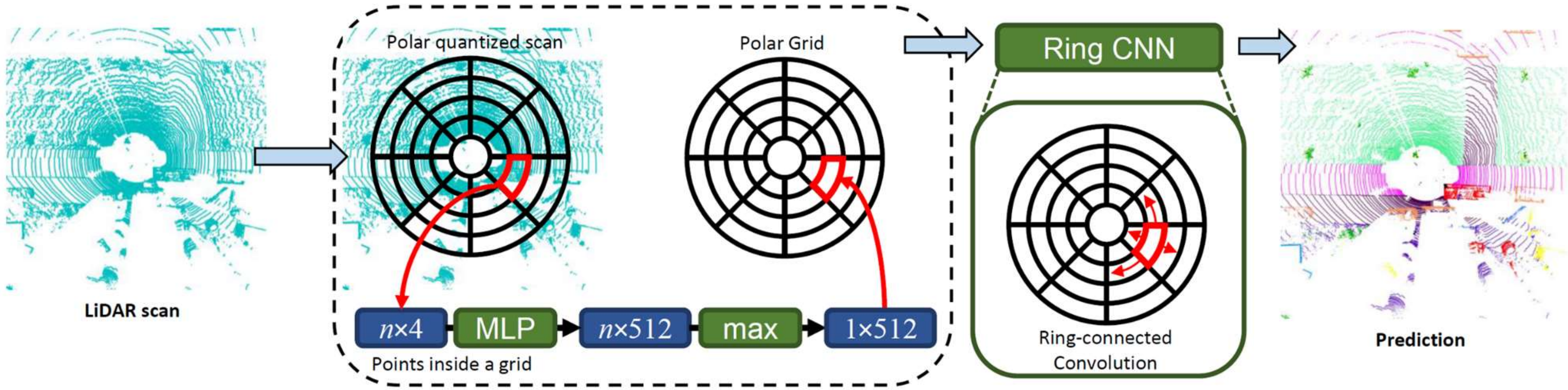


Figure 2. Overview of PolarNet model. Quantize the points into grids using their polar BEV coordinates. Then use a simplified KNN-free PointNet to transform points in it to a fixed-length representation. The representation is then assigned to its corresponding location in the ring matrix. Input the matrix to the ring CNN, which is composed of ring convolution modules and output a quantized prediction and finally decode it to the point domain.

Cylindrical and Asymmetrical 3D Convolution Networks

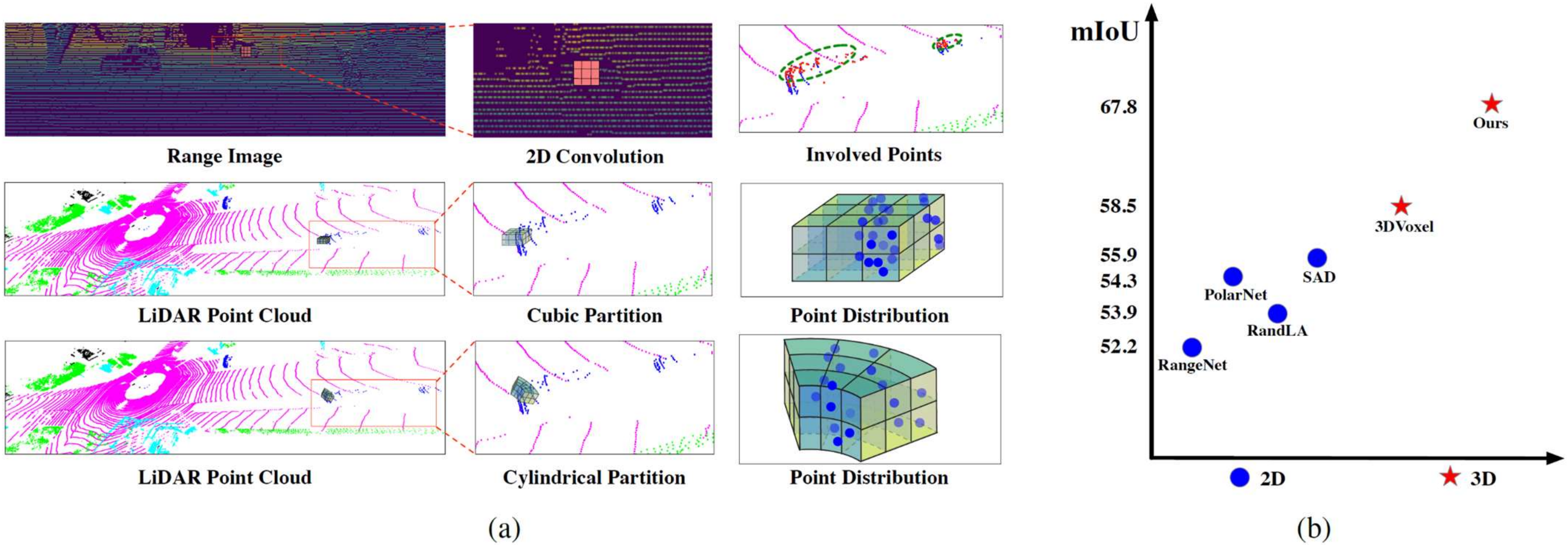


Figure 1: (a) Range Image (2D projection) v.s. Cubic Partition v.s. Cylindrical Partition.

Zhu, X., Zhou, H., Wang, T., Hong, F., Ma, Y., Li, W., Li, H., & Lin, D. (2020). Cylindrical and Asymmetrical 3D Convolution Networks for LiDAR Segmentation. *ArXiv, abs/2011.10033*.

Cylindrical and Asymmetrical 3D Convolution Networks

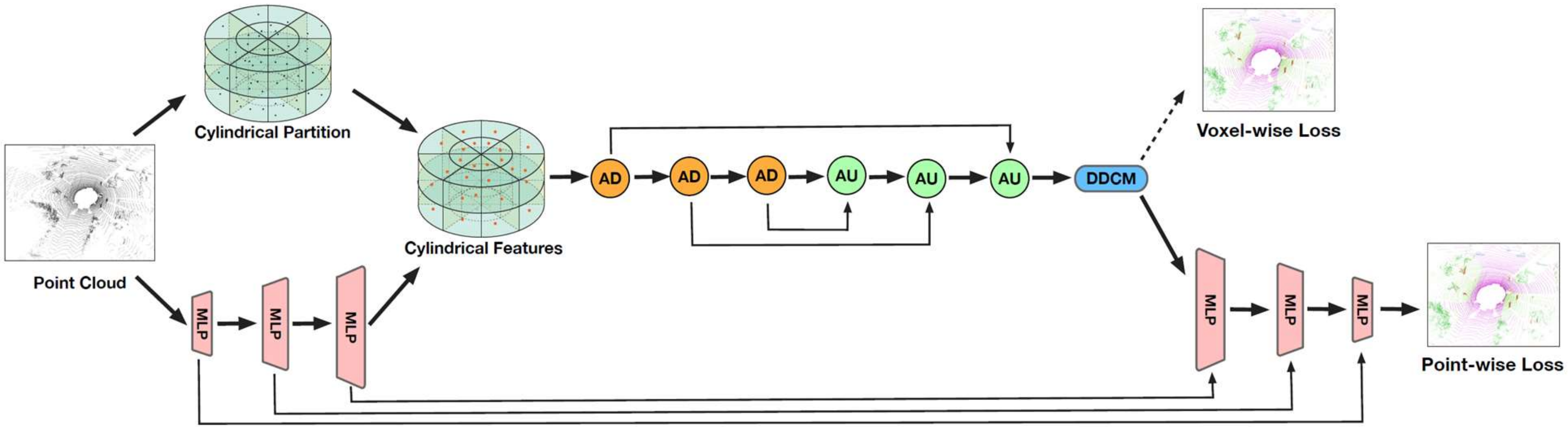
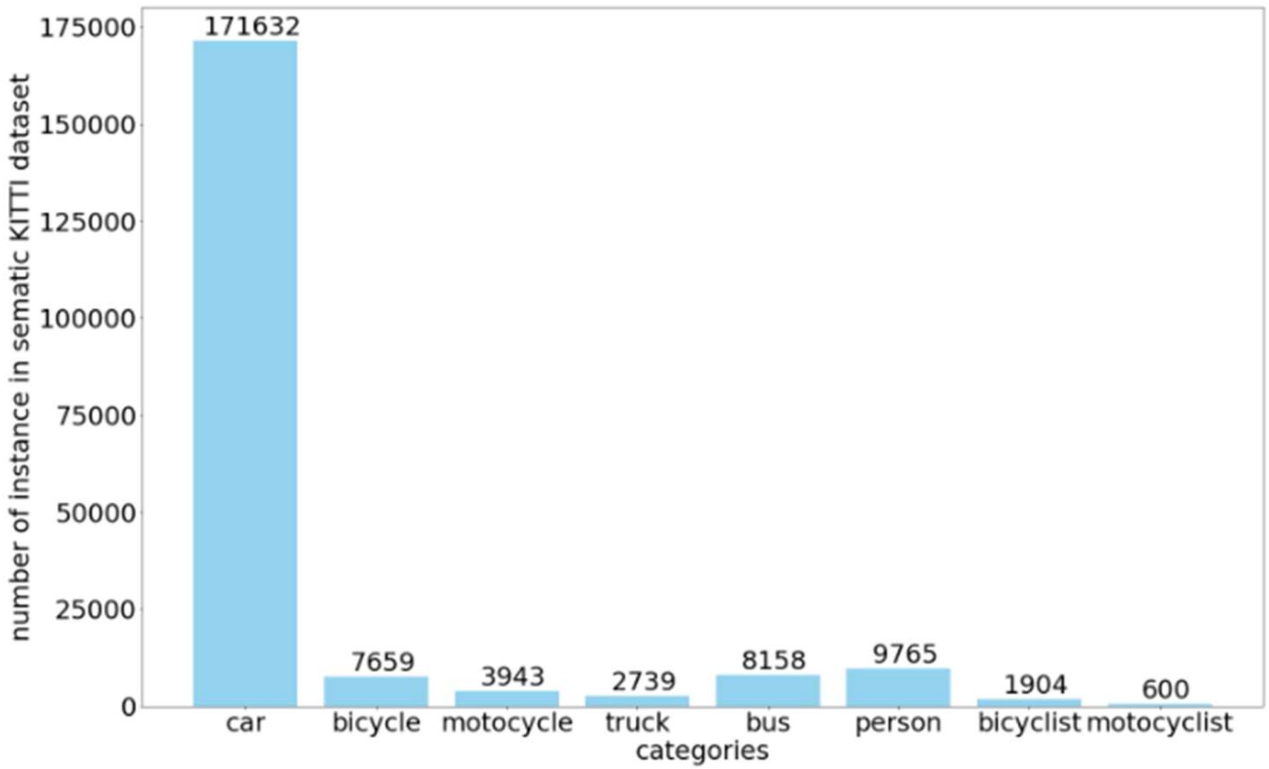
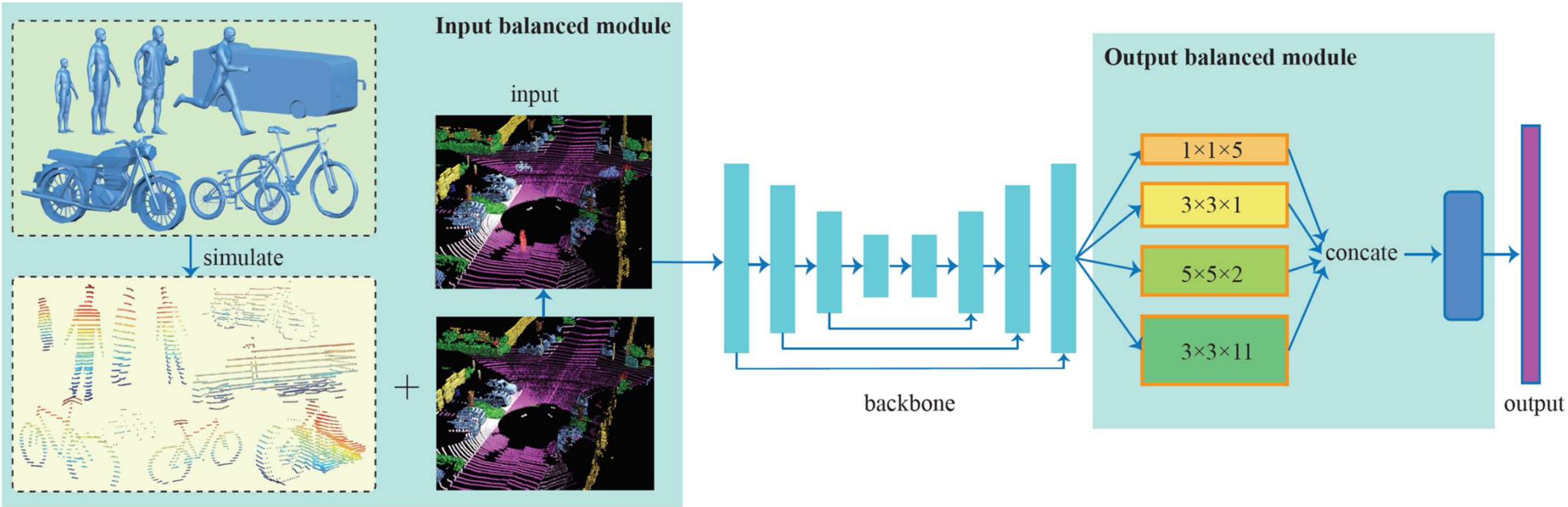


Figure 2: The overall framework. Here, LiDAR point cloud is fed into MLP to get the point-wise features and then these features are reassigned based on the cylinderical partition. Asymmetrical 3D convolution networks are then used to generate the voxel-wise outputs. Finally, a point-wise module is introduced to refine these outputs.

Long-tailed Point Cloud Segmentation

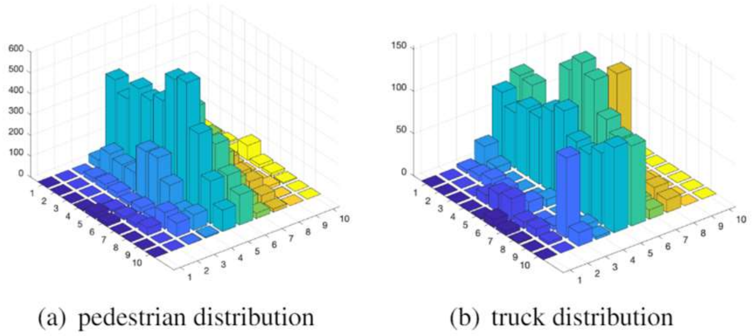


Long-tailed Point Cloud Segmentation

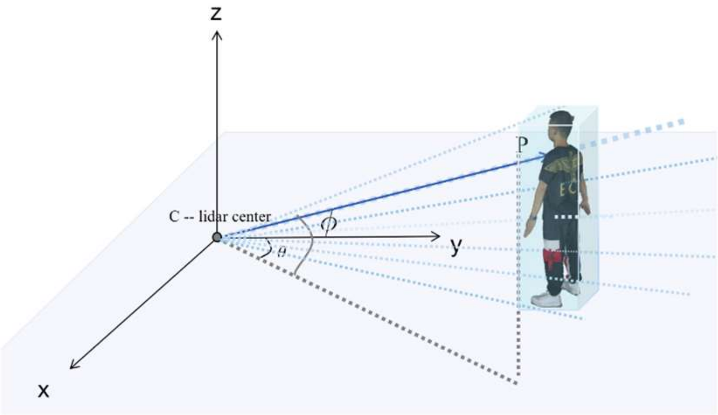


Input-output balanced model

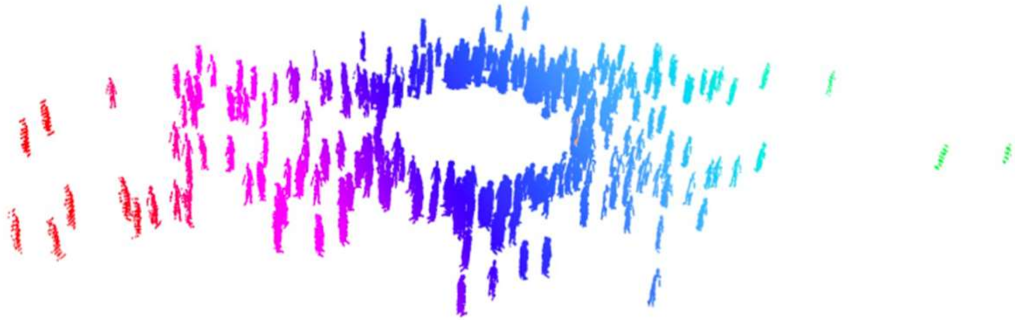
Long-tailed Point Cloud Segmentation



Instance distribution



Point cloud simulation



Database construction

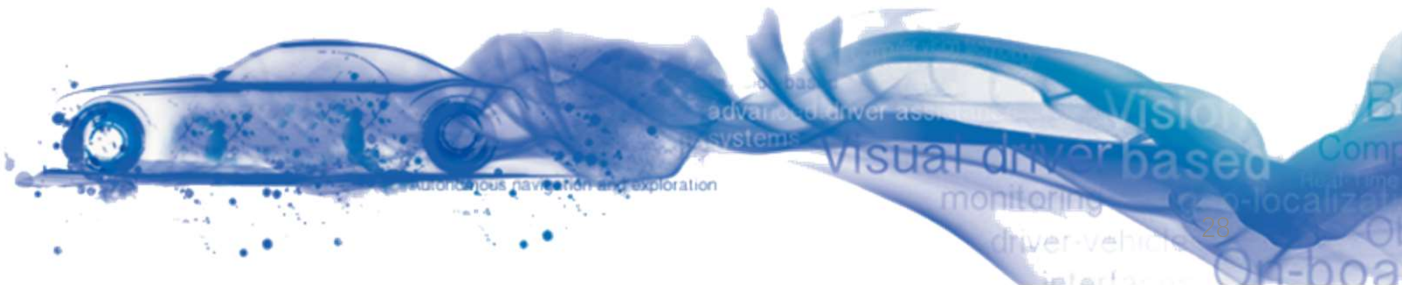
Long-tailed Point Cloud Segmentation

Table 3. Ablation studies for data agumentation on different backbone on SemanticKITTI validation set.

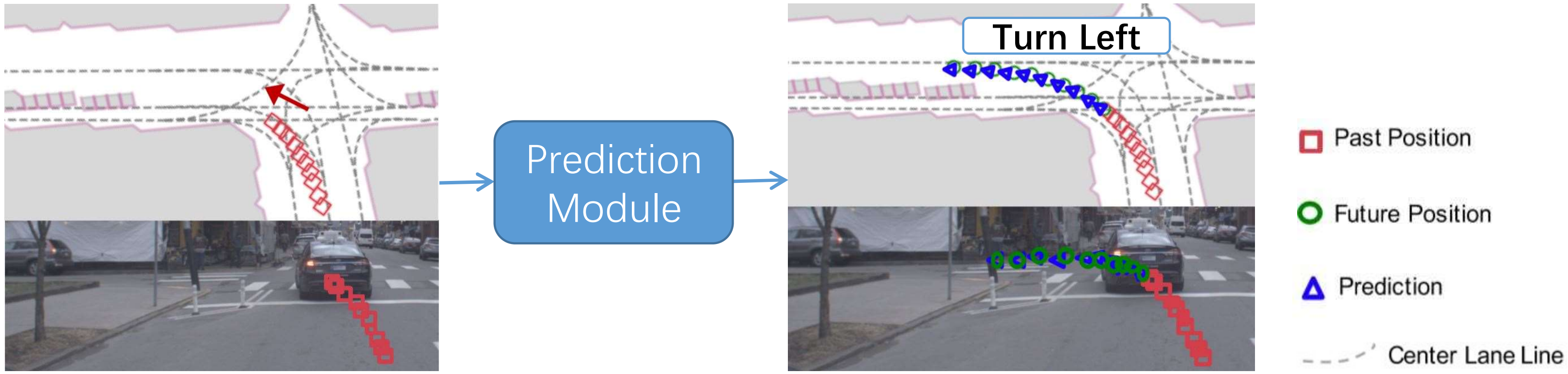
method	mIOU
Polarnet	56.46
Polarnet + data agumentation	58.237
Salsanext	57.547
Salsanext + data agumentation	58.452

Research for Autonomous Driving

Prediction



Prediction for Autonomous Driving

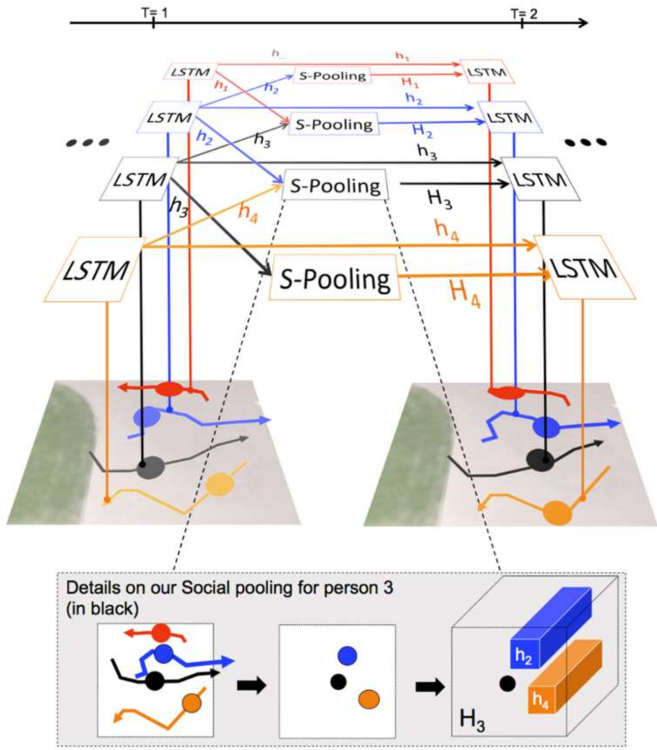


Observed trajectory

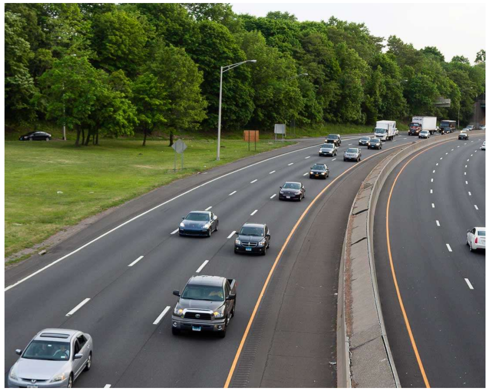
Predicted behavior and trajectory

Prediction for Autonomous Driving

Social-LSTM



Current test scenarios for autonomous vehicles



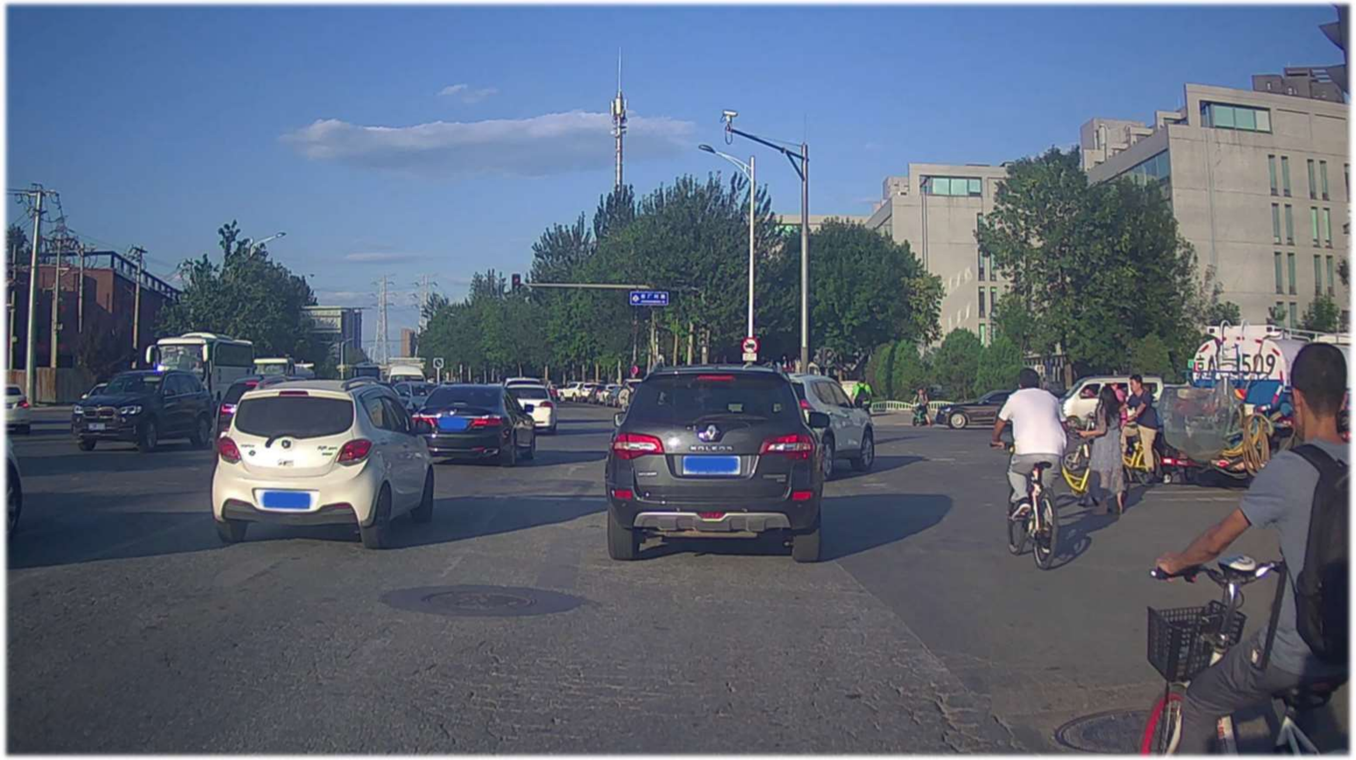
- clear lanes
- clear traffic lights
- monotonous scenes with few pedestrians, bicycles, etc.
- not too much interaction

Dense Heterogeneous Urban Traffic

Captured in India

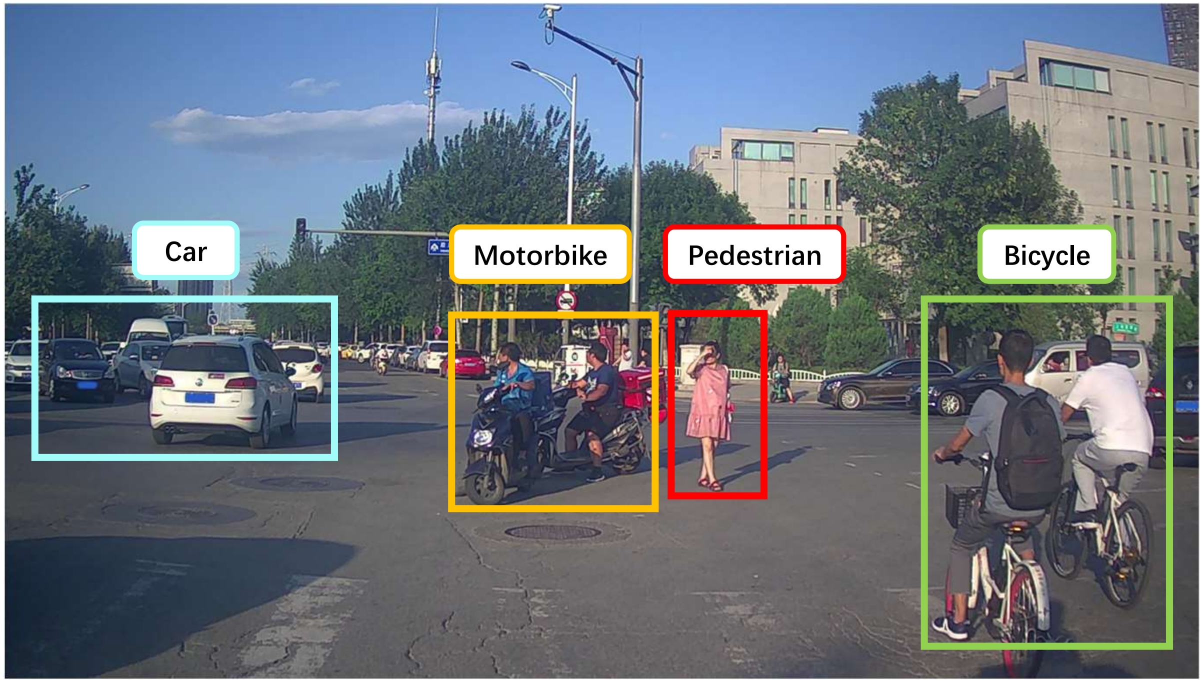
TrafficPredict: Trajectory Prediction for Heterogeneous Traffic-Agents

AAAI, Oral, 2019



Captured in Beijing

Dense Heterogeneous Urban Traffic



- **Different kinds of traffic-agents**
  (cars, bicycles, buses, pedestrians, etc..)
- **Different shapes, dynamics, motion patterns**
- **Different interactions with others**

Motivation

Challenges arise in dense urban environments!

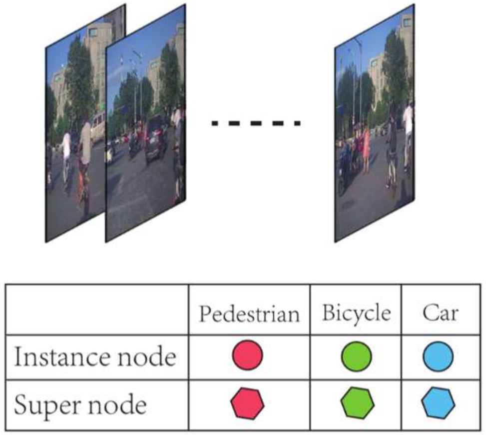
Better prediction → Better navigation → More safety

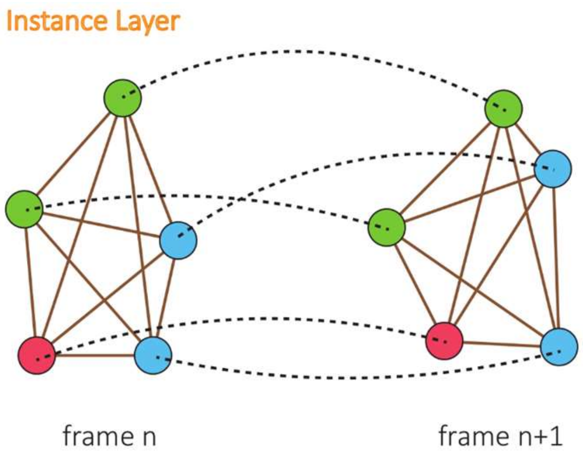
Problem Definition

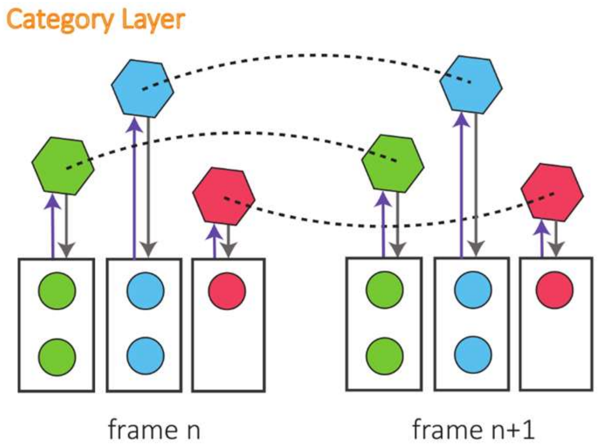
- The feature of traffic-agent A_i^t at time t is denoted as $f_i^t = (x_i^t, y_i^t, c_i^t)$, where the (x_i, y_i) are coordinates, c_i is the category.

- Our task is to observe features of all the traffic-agents in the time interval $[1 : T_{obs}]$, and then predict their discrete positions at $[T_{obs}+1 : T_{pred}]$.

4D Graph





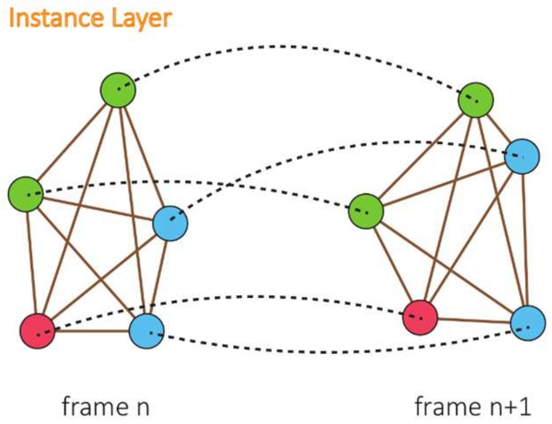


For a traffic sequence

Each node is a traffic-agent.

Each super node is a category.

Model Architecture for Instance Layer

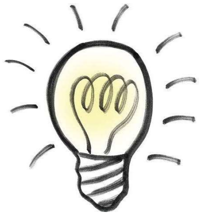


Three main components:

- Temporal edges pass temporal info.
- Spatial edges pass interaction info.
- Instance nodes combine info from edges and themselves to do prediction.

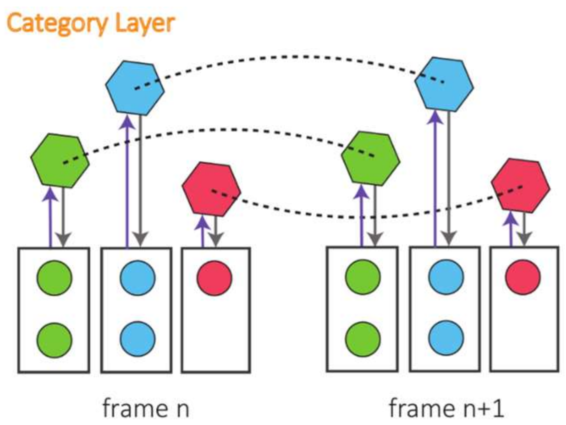
Capture the movement pattern of instances and their interactions in traffic scenarios.

Model Architecture for Category Layer



Usually traffic-agents of the same category have similar dynamic properties, including the speed, acceleration, steering, etc., and similar reactions to other kinds of traffic-agents or the whole environment.

Model Architecture for Category Layer

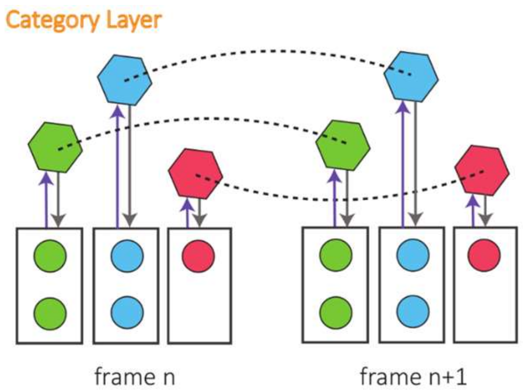


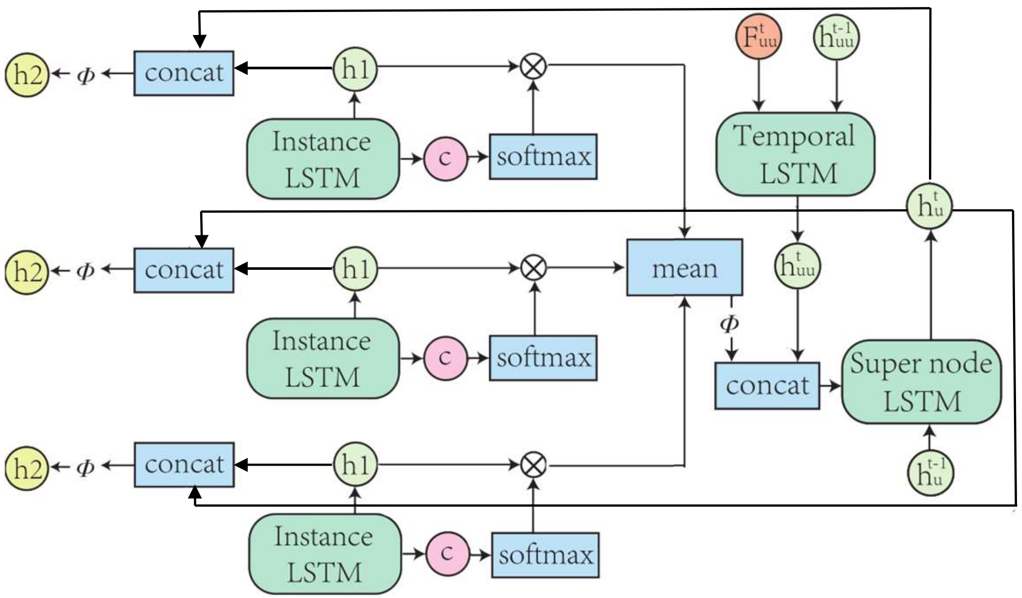
Four main components:

- the super node for a specified category
- the directed edge from a group of instances to the super node
- the directed edge from the super node to instances
- the temporal edge for super node

Capture similarities of the movement pattern of instances belonging to the same category and then refine the prediction for instances.

Model Architecture for Category Layer

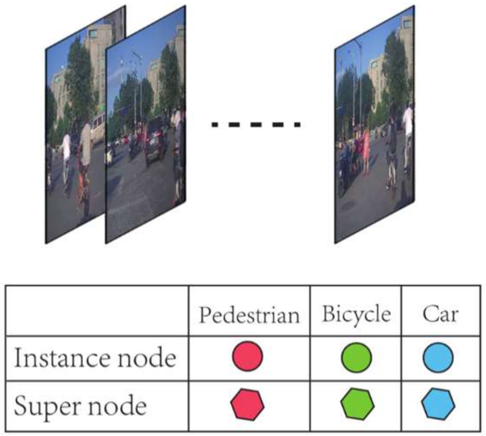


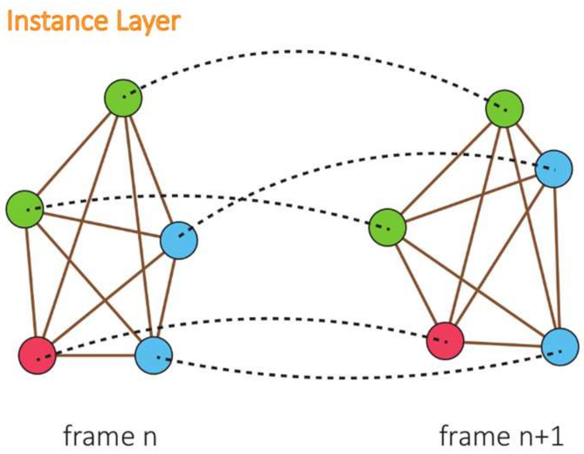


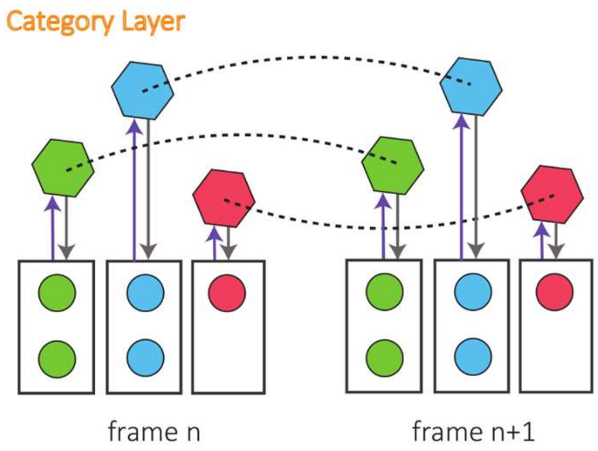
Pass the information from the super node to instance.

Self-attention $d_n^t = h1_n^t \bigotimes softmax(c_n^t)$

4D Graph







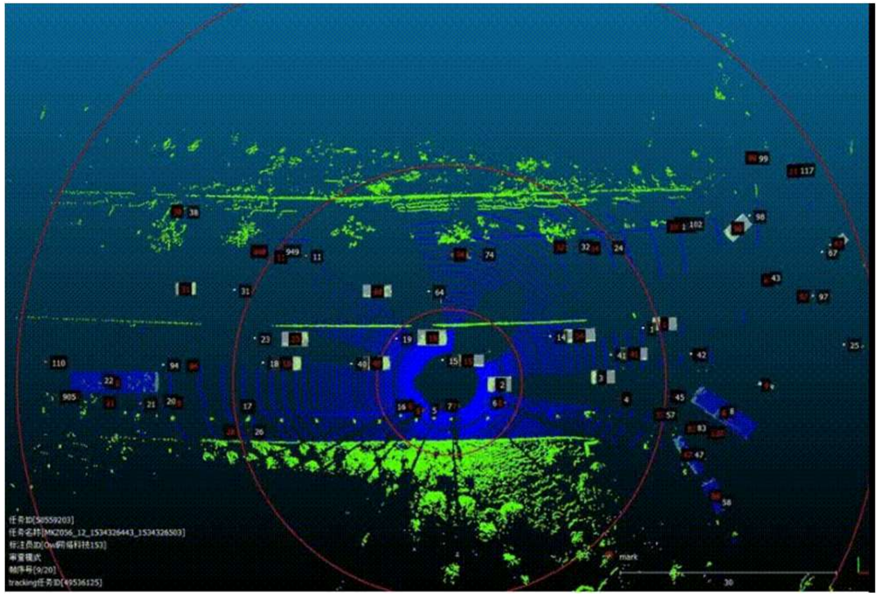
For a traffic sequence

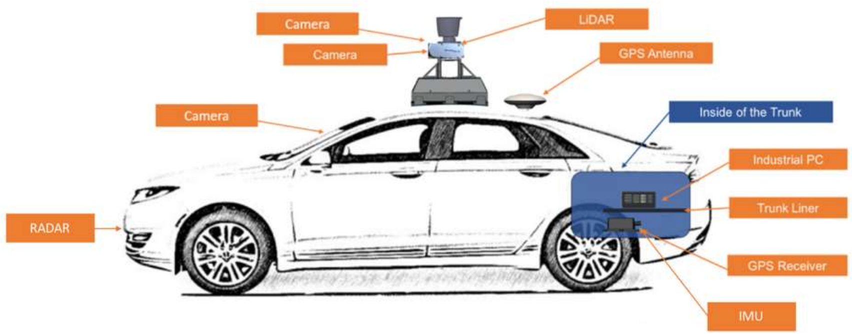
Each node is a traffic-agent.

Each super node is a category.

New Trajectory Dataset

We use an Apollo acquisition car to collect traffic data in rush hours in Beijing.





Samples in New Trajectory Dataset

New Trajectory Dataset

- A large-scale trajectories dataset for urban streets
- Useful for planning, prediction and simulation tasks

Table 1: The acquisition time, total frames, total instances (count ID), average instances per frame, acquisition devices of NGSIM, KITTI (with tracklets) and our dataset.

Count		NGSIM	KITTI	Our Dataset
duration (min)		45	22	**155**
frames ($\times 10^3$)		11.2	13.1	**93.0**
total ($\times 10^3$)	pedestrian	0	0.09	**16.2**
	bicycle	0	0.04	**5.5**
	vehicle	2.91	0.93	**60.1**
average (1/f)	pedestrian	0	1.3	**1.6**
	bicycle	0	0.24	**1.9**
	vehicle	**845**	3.4	12.9
device	camera	yes	yes	yes
	lidar	no	yes	yes
	GPS	no	yes	yes

Results

Table 2: The average displacement error and the final displacement error of the prior methods (ED, SL, SA) and variants of our method (TP) on our new dataset. For each evaluation metric, we show the values on pedestrians, bicycles, vehicles, and all the traffic-agents. We set the observation time as 2 seconds and the prediction time as 3 seconds for these measurements.

Metric	Methods	ED	SL	SA	TP-NoCL	TP-NoSA	TrafficPredict
Avg. disp. error	pedestrian	0.121	0.135	0.112	0.125	0.118	**0.091**
	bicycle	0.112	0.142	0.111	0.115	0.110	**0.083**
	vehicle	0.122	0.147	0.108	0.101	0.096	**0.080**
	total	0.120	0.145	0.110	0.113	0.108	**0.085**
Final disp. error	pedestrian	0.255	0.173	0.160	0.188	0.178	**0.150**
	bicycle	0.190	0.184	0.170	0.193	0.169	**0.139**
	vehicle	0.195	0.202	0.189	0.172	0.150	**0.131**
	total	0.214	0.198	0.178	0.187	0.165	**0.141**

Results

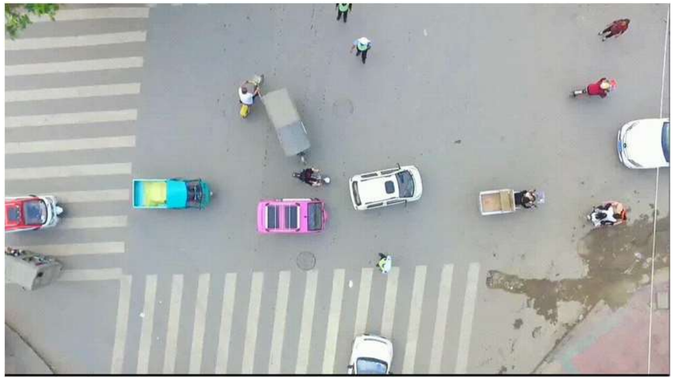
Illustration of comparison results on camera-based images.

AutoTrajectory: Label-free Trajectory Extraction and Prediction from Videos using Dynamic Points

ECCV, 2020

- Current **prediction methods** are supervised, which rely heavily on **labeled trajectory data**.
- For **supervised trackers**, performance depends largely on the **supervised detector**, which also needs large-scale labeled data and is always trained with fixed categories and domains.
- For **unsupervised trackers**, they could not handle common bird's-eye view videos.



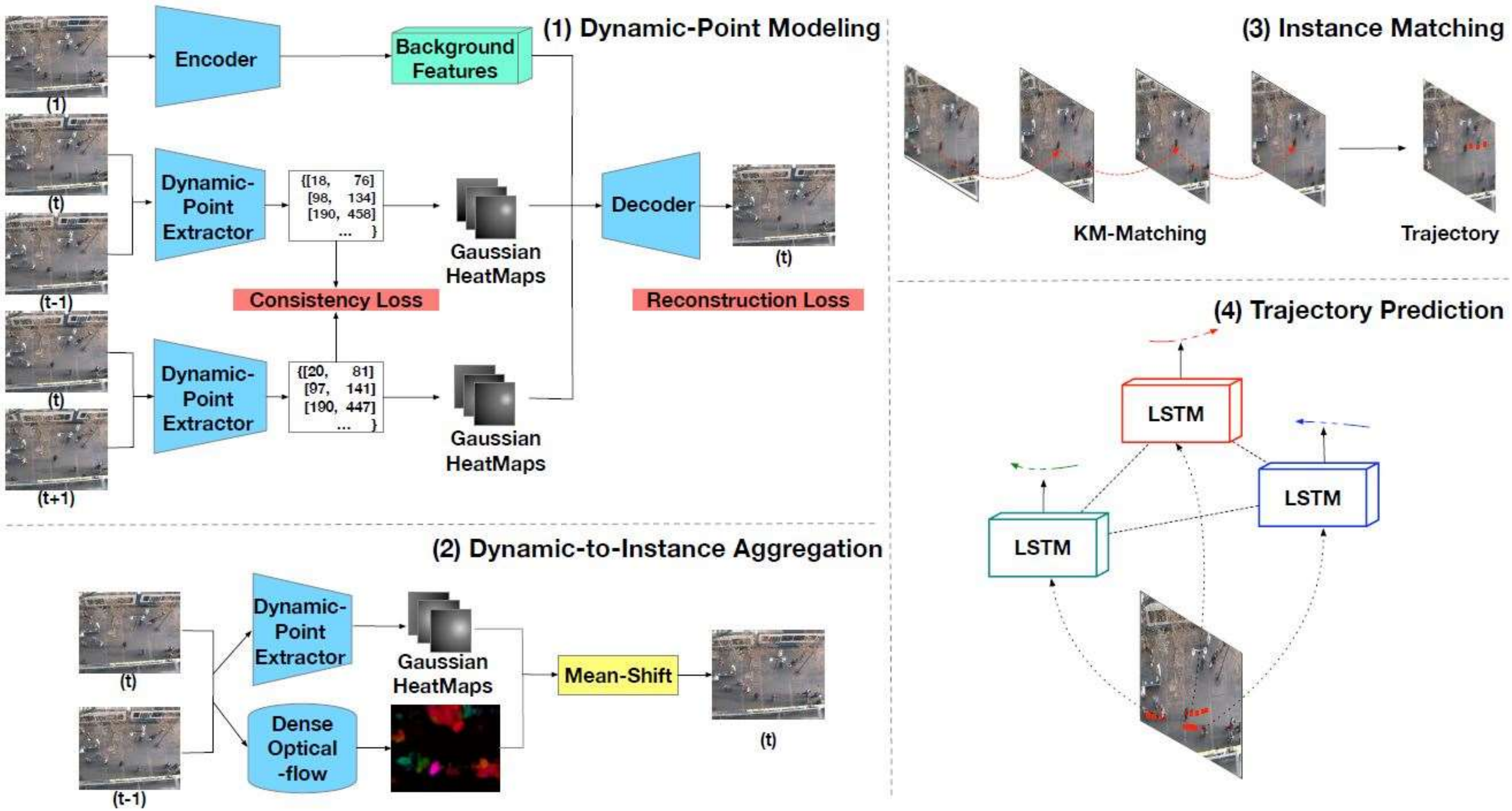


Method

raw videos | trajectory extractor | trajectory predictor

- Our approach is **label-free**.
- Our approach focuses on exploring the nature of video, i.e., **the dynamic information**, which is naturally **category-free** and works well on **all domains**.
- We achieved **SOTA** performance for unsupervised tracking
- Our approach can further **improve prediction** methods by providing more trajectory data.

Method

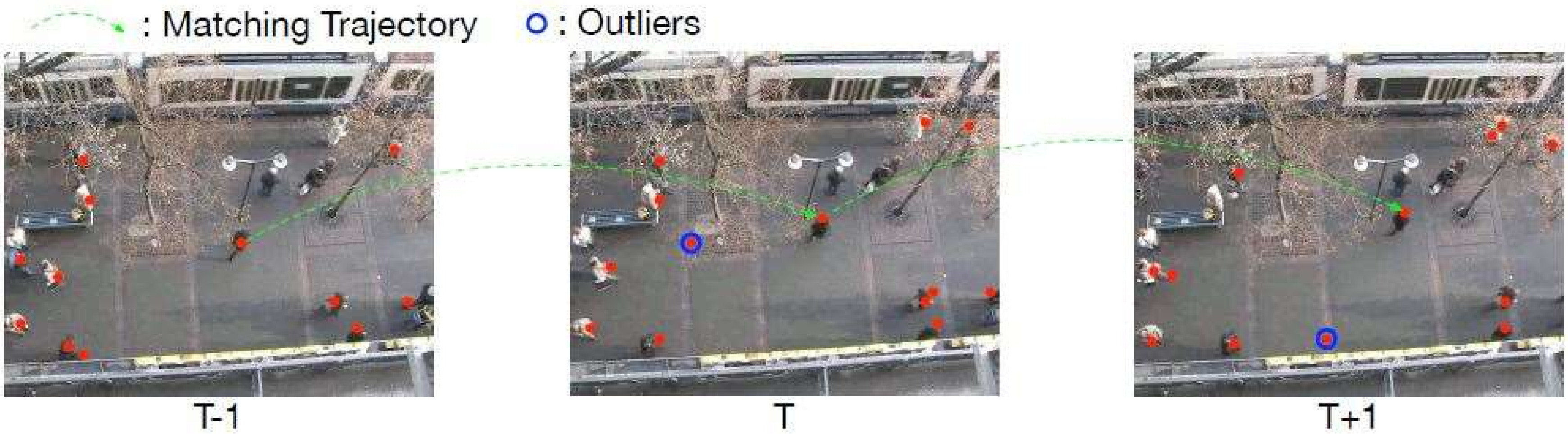


(1) Dynamic-Point Modeling
Encoder
Background Features
(1)
(t)
(t-1)
(t)
(t+1)
Dynamic-Point Extractor
{[18, 76] [98, 134] [190, 458] ... }
Gaussian HeatMaps
Decoder
(t)
Consistency Loss
Reconstruction Loss
Dynamic-Point Extractor
{[20, 81] [97, 141] [190, 447] ... }
Gaussian HeatMaps
(2) Dynamic-to-Instance Aggregation
(t)
(t-1)
Dynamic-Point Extractor
Gaussian HeatMaps
Dense Optical -flow
Mean-Shift
(t)
(3) Instance Matching
KM-Matching
Trajectory
(4) Trajectory Prediction
LSTM
LSTM
LSTM

Reconstruction results

The input image vs. the reconstructed image from the decoder. a and b are the input images, and rec_a and rec_b are the reconstructed images.

Instance matching results



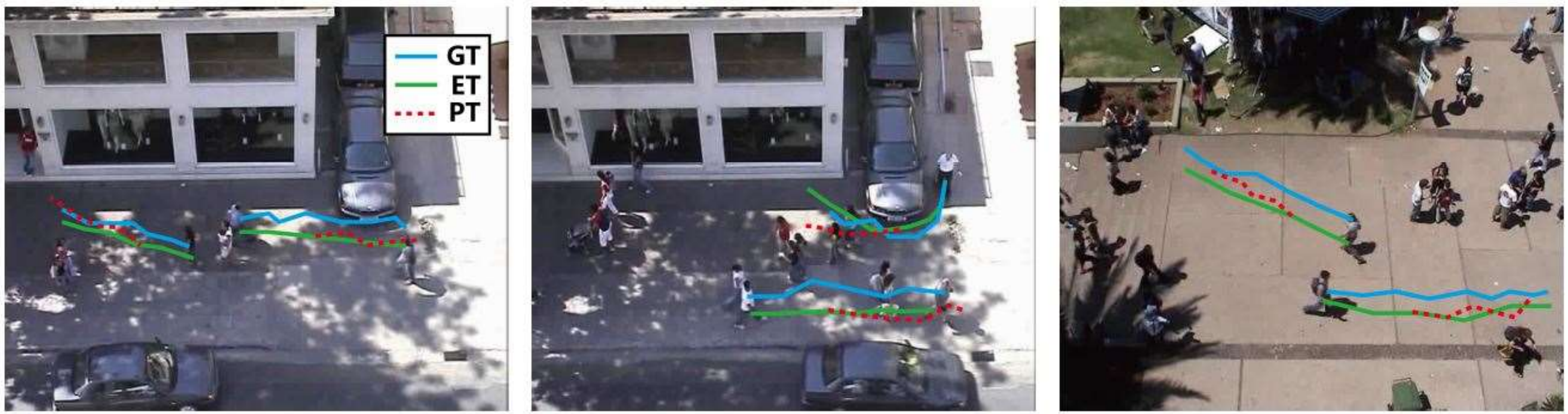
An example of instance matching. Green dashed line denotes the instance points matching across timesteps. Blue circles denote the outliers of the instance points (also mean missmatching points).

Trajectory extraction results

Table 1. Evaluation results of detected instance points. We compare the proposed method with the unsupervised tracking [14] method and unsupervised keypoint modeling method [24]. '-' indicates the model cannot converge in the dataset

Metric	Ins-Precision					Ins-Recall				
Dataset	ETH	Hotel	Univ	Zara1	Zara2	ETH	Hotel	Univ	Zara1	Zara2
Un-Tracking [14]	8.3%	-	-	19.6%	21.4%	12.7%	-	-	10.1%	14.8%
Un-Keypoint [24]	16.8%	11.2%	-	33.1%	36.7%	14.1%	14.6%	-	39.4%	41.0%
Ours	**47.9%**	**37.1%**	**36.4%**	**58.7%**	**60.3%**	**58.3%**	**42.0%**	**31.4%**	**63.1%**	**67.9%**

Trajectory extraction and prediction results



(a) (b) (c)

We display three examples with the ground truth trajectory (GT in green line), the extracted trajectory by our method (ET in blue line), and the predicted trajectory by our method (PT in red dashed line).

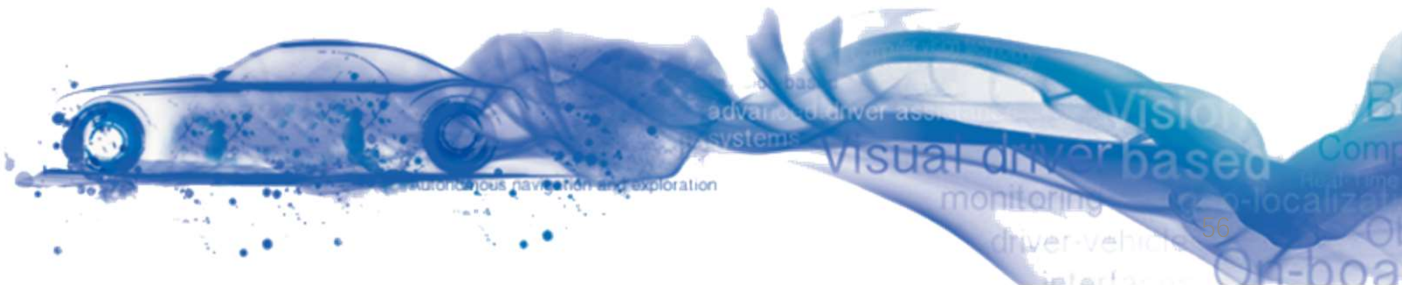
Semi-supervised prediction results

Dataset	Zara1		+Univ(Gen)		+Univ(Gen)+Zara2(Gen)	
Method	LSTM	S-LSTM	LSTM	S-LSTM	LSTM	S-LSTM
ADE	0.598	0.347	0.578	0.341	0.521	0.320
FDE	1.25	0.69	1.157	0.687	1.094	0.659

Adding more our extracted trajectories in the training process will make the prediction results more accurate.

Research for Autonomous Driving

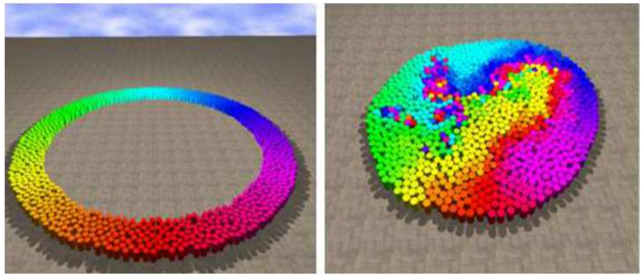
Navigation



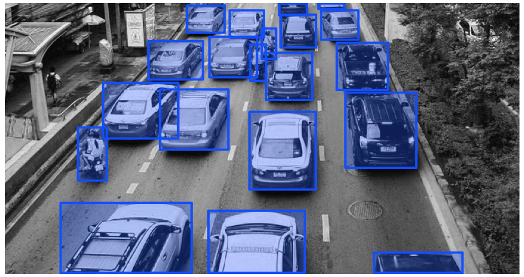
Efficient Reciprocal Collision Avoidance with Heterogeneous Agents Using CTMAT

AAMAS, 2018

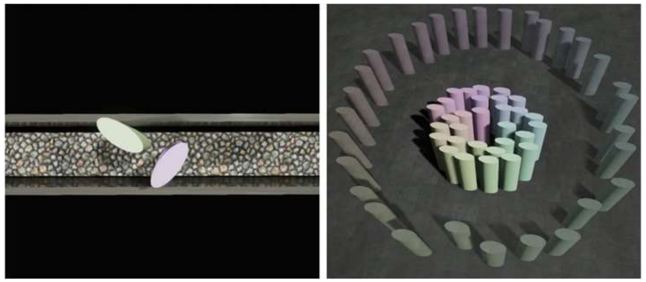
1 Disc



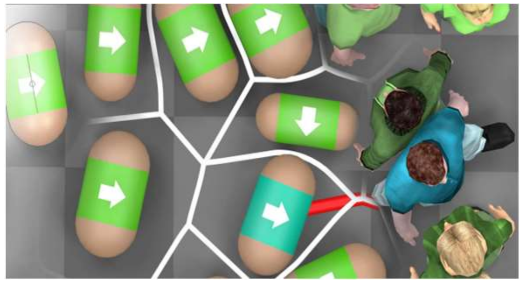
2 Rectangle



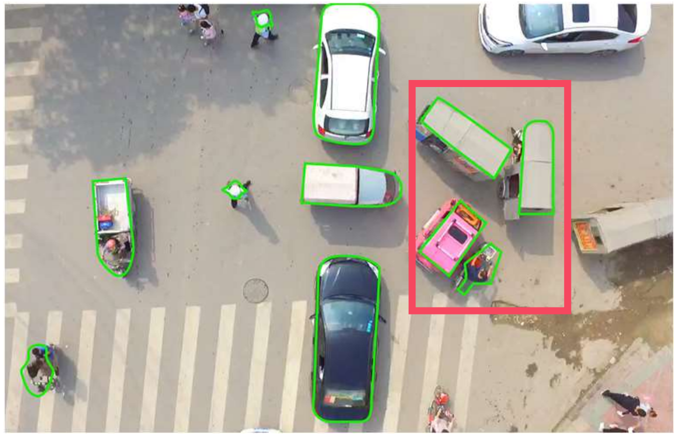
3 Ellipse

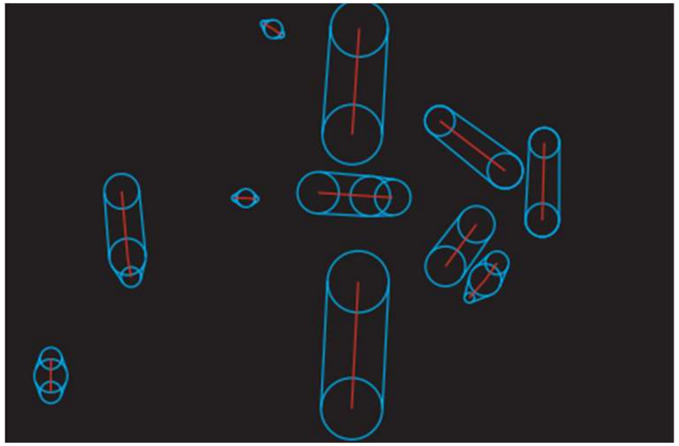


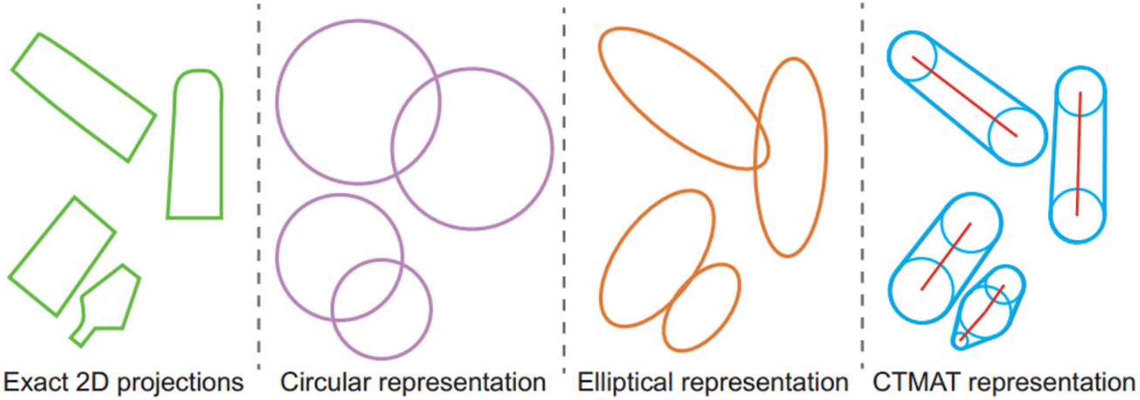
4 Capsule



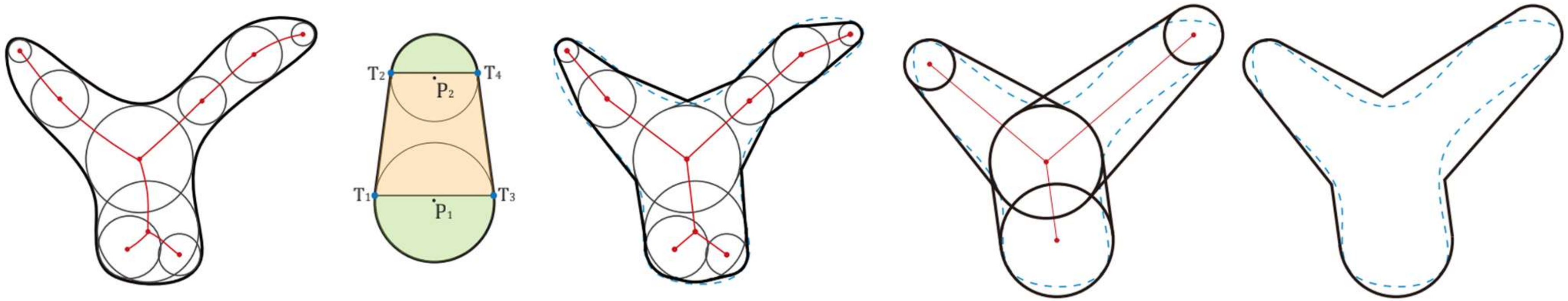
Comparison of Different Representations







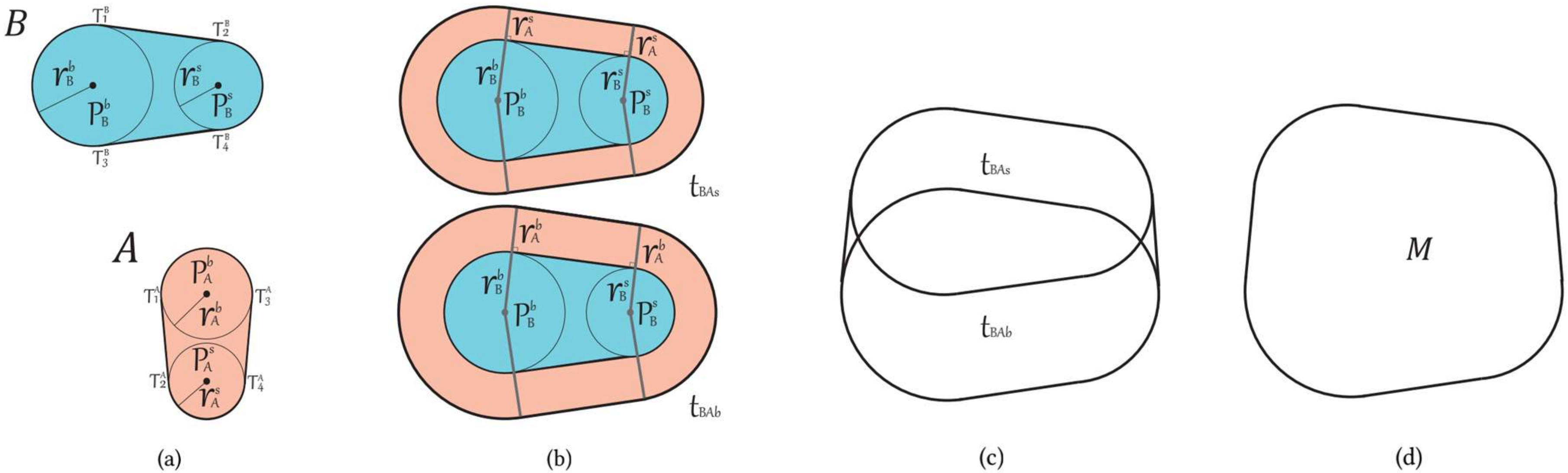
Medial Axis Transformation (MAT)



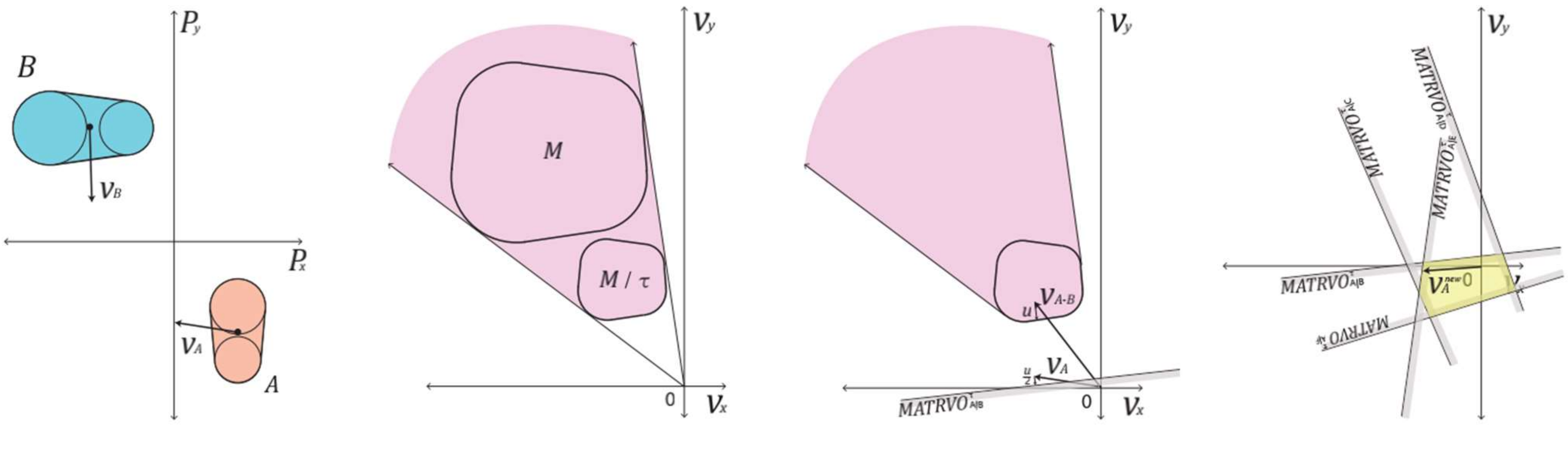
(a) (b) (c) (d) (e)

In 2D, the medial axis of a shape which is bounded by planar curve C is the locus of the centers of circles that are tangent to curve C in two or more points.

Minkowski Sum of Two Tuples



Velocity Obstacle for Heterogeneous Agents



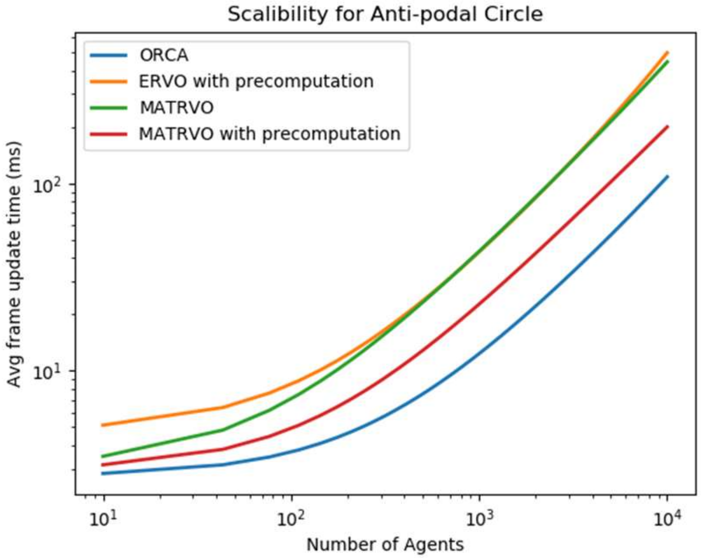
(a) (b) (c) (d)

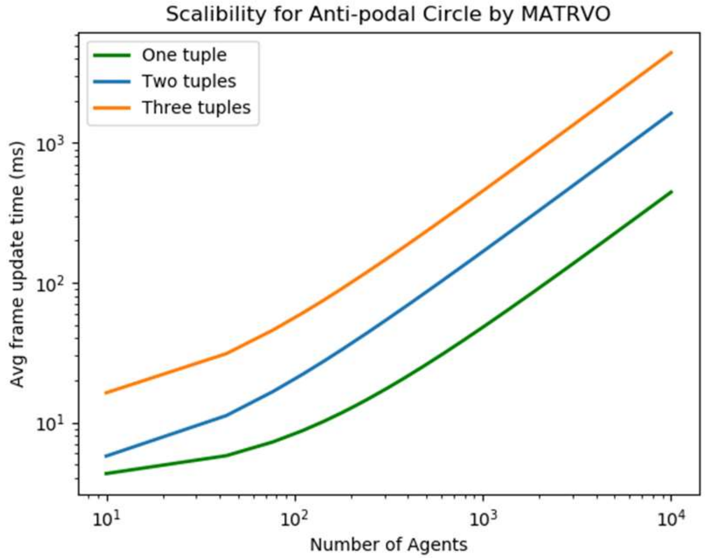
Theorem. If MATRVO algorithm is able to compute a feasible velocity, the resulting motion for agent is collision-free.

$$MATRVO_A^{\tau} = \bigcap_{B \neq A} MATRVO_{A|B}^{\tau}$$

$$\vec{v}_A^{new} = \underset{\vec{v} \in MATRVO_A^{\tau}}{\arg\min} \left\| \vec{v} - \vec{v}_A^{0} \right\|$$

Results

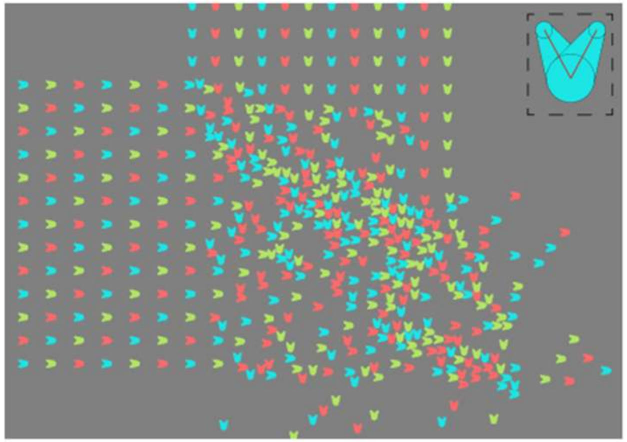


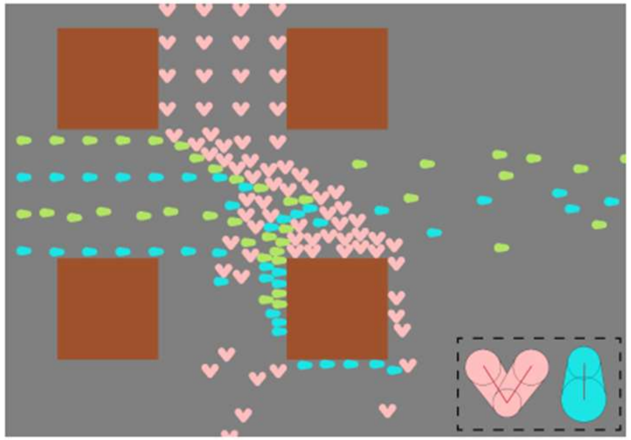


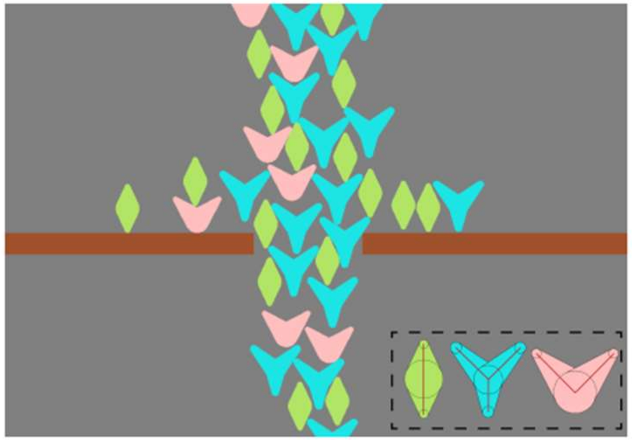
Test	Agent	ORCA [48]	ERVO [10]	MATRVO
1				
		54.5%	37.9%	8.5%
2				
		60.4%	31.4%	9.8%
3				
		47.3%	46.2%	9.2%

Comparison of ratios of false positives.

Results







AutoRVO: Local Navigation with Dynamic Constraints in Dense Heterogeneous Traffic

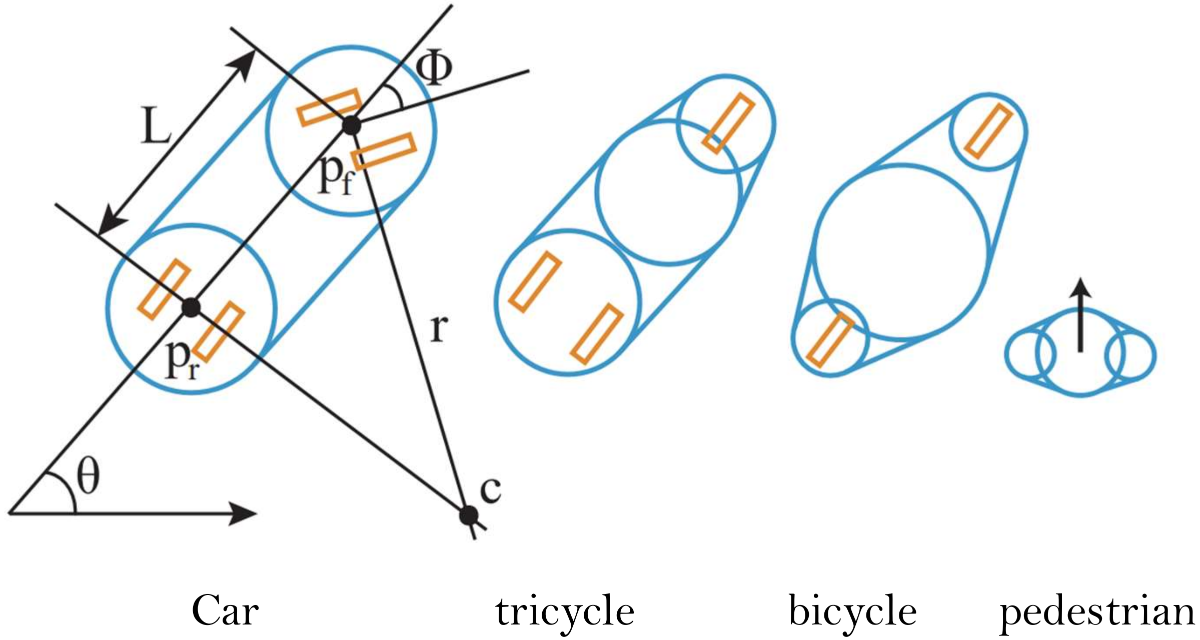
CSCS, 2018

Kinematics

Dynamics

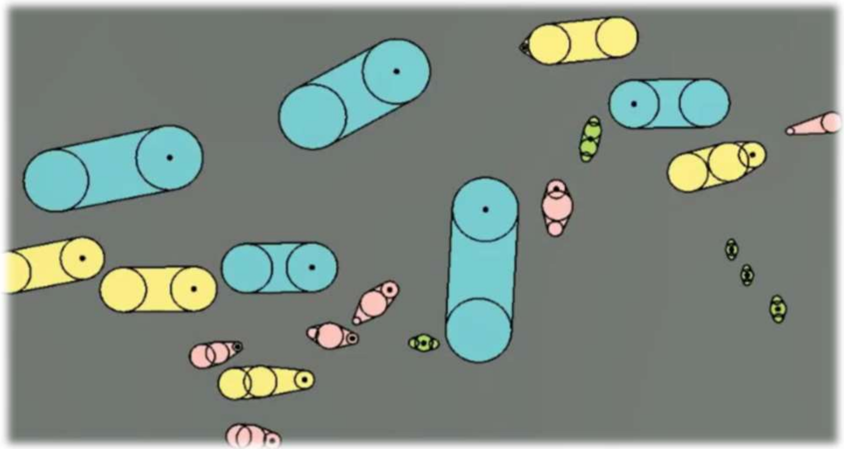
To make the simulation more real.

Representation and Kinematic Models



$$\dot{\vec{p}} = (v\cos(\theta), v\sin(\theta)), \qquad \dot{\theta} = \frac{\tan(\phi)}{L} v.$$

Result

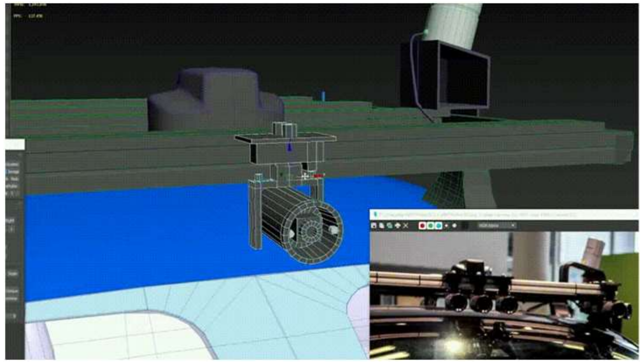


AADS: Augmented Autonomous Driving Simulation using Data-driven Algorithms

Science Robotics, 2019

- **Simulation systems** have become an **essential component** in the development and validation of autonomous driving technologies.

- **Current simulation approaches** use game engines or high-fidelity computer graphics (CG) models to create driving scenarios.
  - Remains a manual task that can be **costly and time-consuming**.
  - **Lacks the richness and authenticity** of real-world.

AADS: Augmented Autonomous Driving Simulation using Data-driven Algorithms

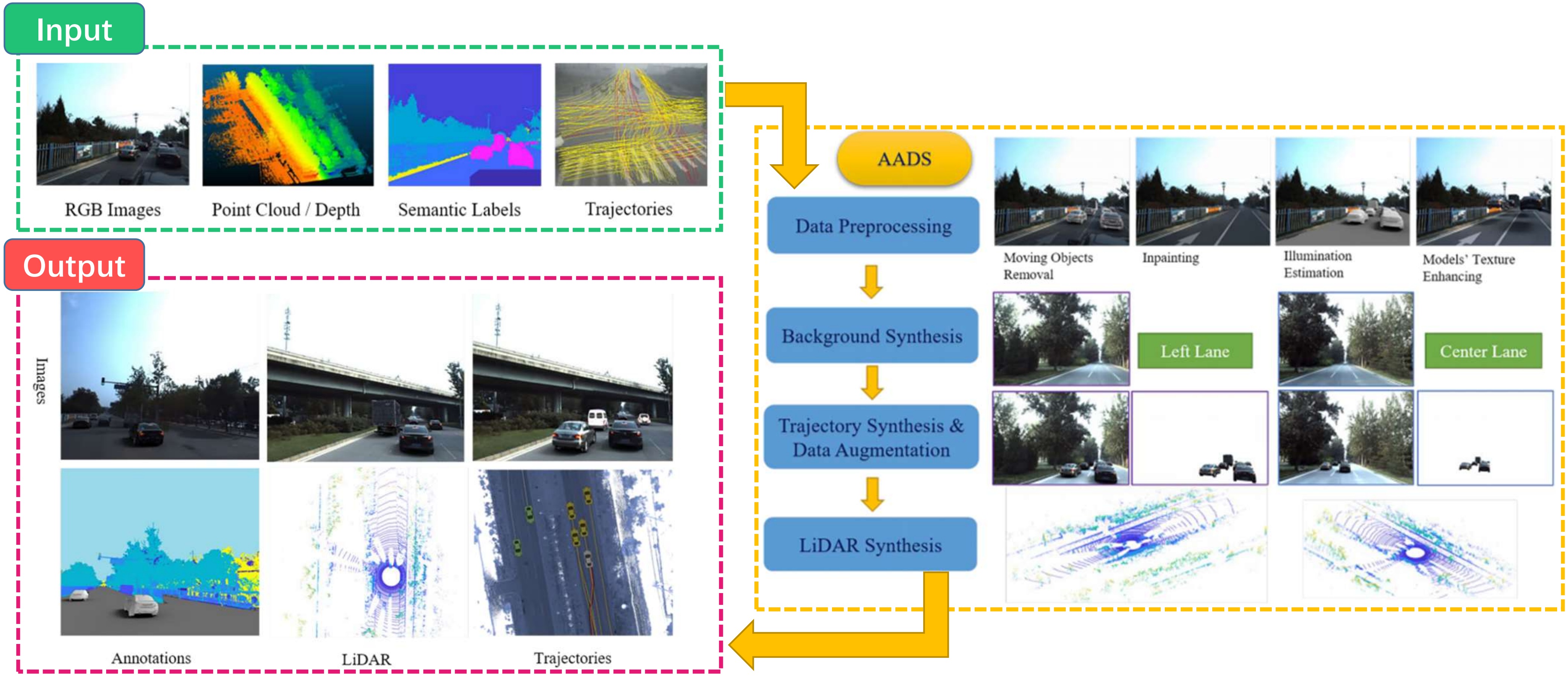
Science Robotics, 2019

- **A new data-driven approach for autonomous driving simulation.** This direct **scan-to-simulation** pipeline, enables large-scale testing of autonomous cars virtually anywhere and anytime within a closed-loop simulation.

- **A novel view synthesis method** to enable view interpolation and extrapolation with only a few images.

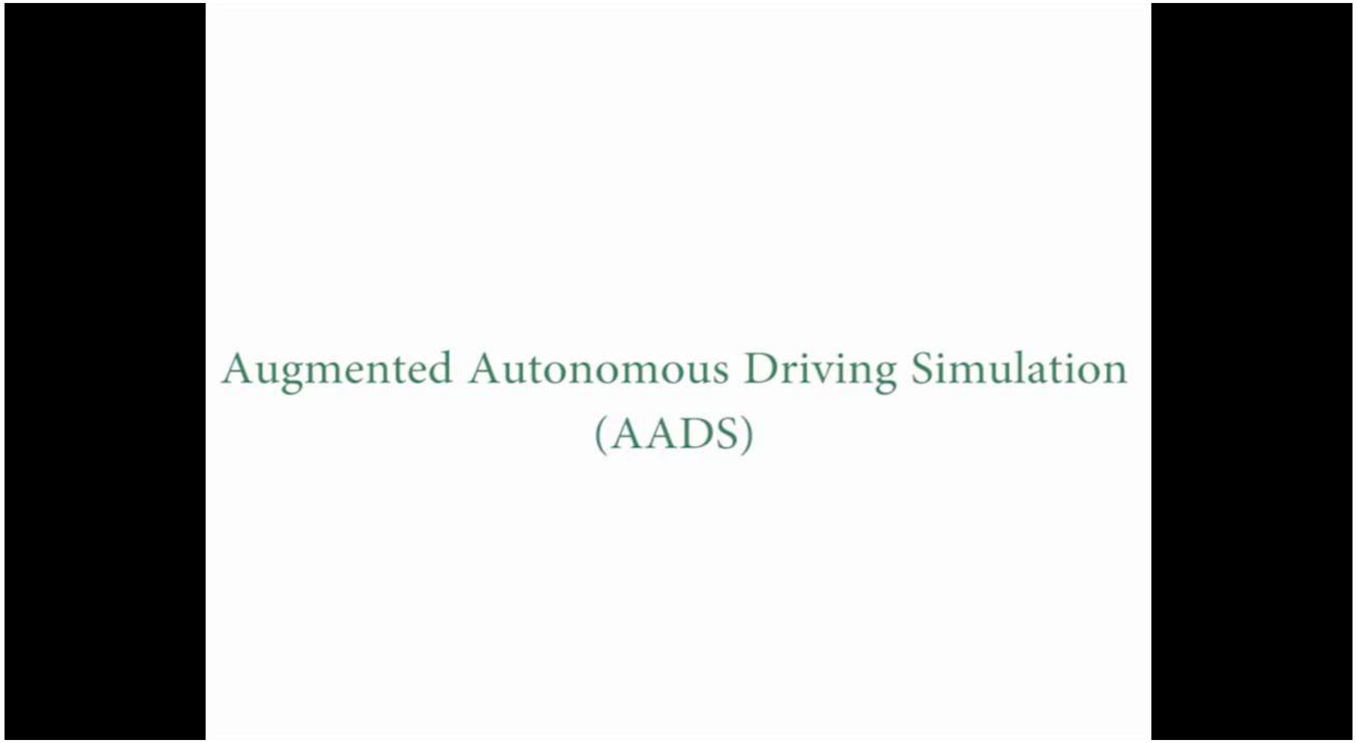
- **A new set of datasets**, including the largest set of traffic trajectories and the largest 3D street-view dataset with pixel/point level annotation.

Pipeline



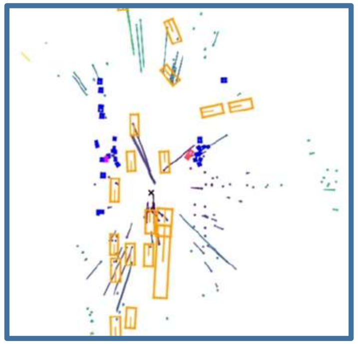
Input
RGB Images
Point Cloud / Depth
Semantic Labels
Trajectories
Output
Images
Annotations
LiDAR
Trajectories
AADS
Data Preprocessing
Background Synthesis
Trajectory Synthesis & Data Augmentation
LiDAR Synthesis
Moving Objects Removal
Inpainting
Illumination Estimation
Models' Texture Enhancing
Left Lane
Center Lane

Result

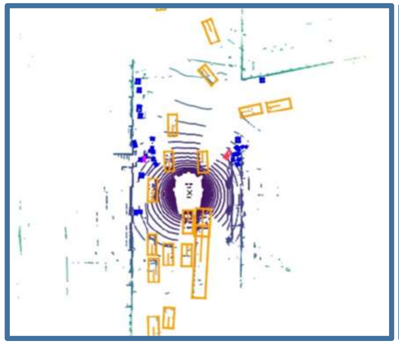


ShanghaiTech Autonomous Car

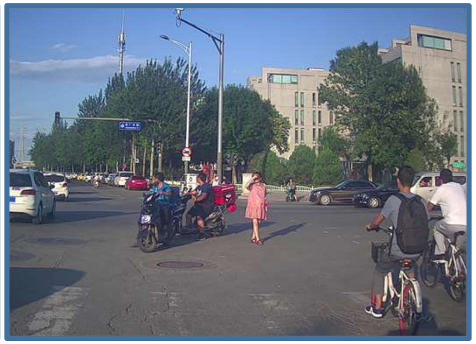
Radar



LiDAR



Camera

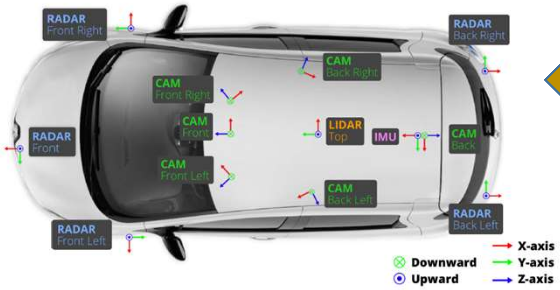


ego sensors

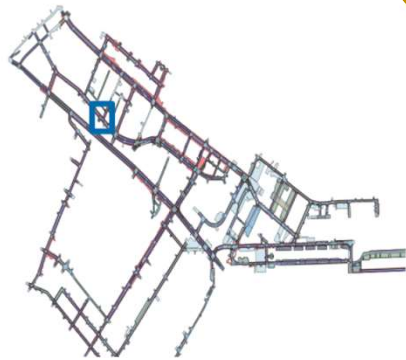
Street camera

Drone

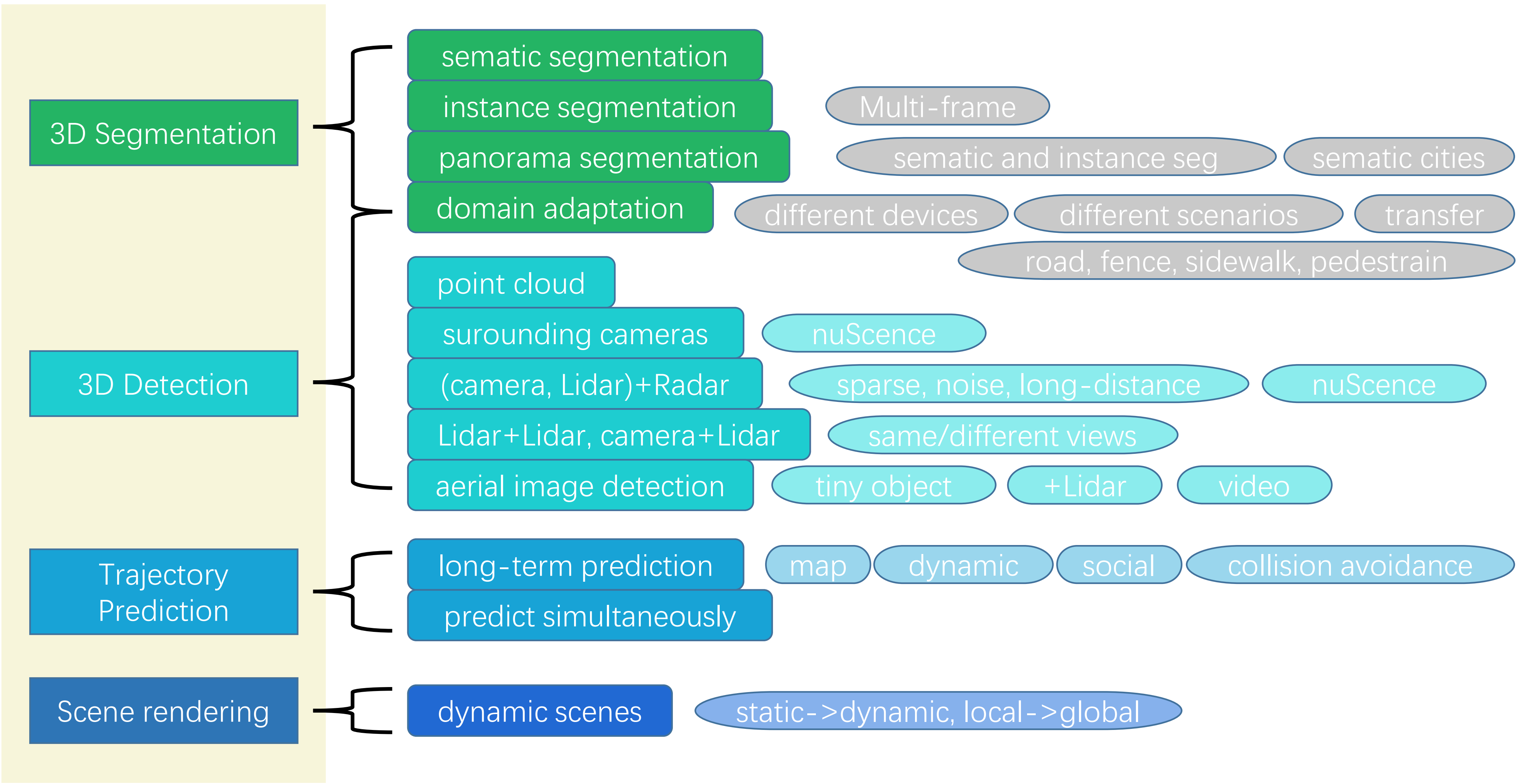
outer sensors



HD map



3D Scene Understanding



Applications for 3D scene understanding

Intelligent campus

- Virtual campus (static+dynamic)
- Service robots (intelligent courier)
- Autonomous vehicles (low speed, campus tour)

Thanks