DSDNet: Deep Structured self-Driving Network

Wenyuan Zeng[1,2], Shenlong Wang[1,2], Renjie Liao[1,2], Yun Chen[1], Bin Yang[1,2], and Raquel Urtasun[1,2]

[1] Uber ATG
[2] University of Toronto
{wenyuan,slwang,rjliao,yun.chen,byang10,urtasun}@uber.com

Abstract. In this paper, we propose the Deep Structured self-Driving Network (DSDNet), which performs object detection, motion prediction, and motion planning with a single neural network. Towards this goal, we develop a deep structured energy based model which considers the interactions between actors and produces socially consistent multimodal future predictions. Furthermore, DSDNet explicitly exploits the predicted future distributions of actors to plan a safe maneuver by using a structured planning cost. Our sample-based formulation allows us to overcome the difficulty in probabilistic inference of continuous random variables. Experiments on a number of large-scale self driving datasets demonstrate that our model significantly outperforms the state-of-the-art.

Keywords: Autonomous driving, motion prediction, motion planning

1 Introduction

The self-driving problem can be described as safely, comfortably and efficiently maneuvering a vehicle from point A to point B. This task is very complex; Even the most intelligent agents to date (*i.e.*, humans) are very frequently involved in traffic accidents. Despite the development of Advanced Driver-Assistance Systems (ADAS), 1.3 million people die every year on the road, and 20 to 50 million are severely injured.

Avoiding collisions in complicated traffic scenarios is not easy, primarily due to the fact that there are other traffic participants, whose future behaviors are unknown and very hard to predict. A vehicle that is next to our lane and blocked by its leading vehicle might decide to stay in its lane or cut in front of us. A pedestrian waiting on the edge of the road might decide to cross the road at any time. Moreover, the behavior of each actor depends on the actions taken by other actors, making the prediction task even harder. Thus, it is extremely important to model the future motions of actors with multi-modal distributions that also consider the interactions between actors.

To safely drive on the road, a self-driving vehicle (SDV) needs to detect surrounding actors, predict their future behaviors, and plan safe maneuvers. Despite the recent success of deep learning for perception, the prediction task, due to the aforementioned challenges, remains an open problem. Furthermore,

there is also a need to develop motion planners that can take the uncertainty of the predictions into account. Previous works have utilized parametric distributions to model multimodality of motion prediction. Mixture of Gaussians [11, 20] are a natural approach due to their close-form inference. However, it is hard to decide the number of modes in advance. Furthermore, these approaches suffer from mode collapse during training [22, 20, 38]. An alternative is to learn a model distribution from data using, *e.g.*, neural networks. As shown in [39, 25, 46], a CVAE [44] can be applied to capture multi-modality, and the interactions between actors can be modeled through latent variables. However, it is typically hard/slow to do probabilistic inference and the interaction mechanism does not explicitly model collision which humans want to avoid at all causes. Besides, none of these works have shown the effects upon planning on real-world datasets.

In this paper we propose the **D**eep **S**tructured self-**D**riving **Net**work (DSD-Net), a single neural network that takes raw sensor data as input to jointly detect actors in the scene, predict a multimodal distribution over their future behaviors, and produce safe plans for the SDV. This paper has three key contributions:

- Our prediction module uses an energy-based formulation to explicitly capture the interactions among actors and predict multiple future outcomes with calibrated uncertainty.
- Our planning module considers multiple possibilities of how the future might unroll, and outputs a safe trajectory for the self-driving car that respects the laws of traffic and is compliant with other actors.
- We address the costly probabilistic inference with a sample-based framework.

DSDNet conducts efficient inference based on message passing over a sampled set of continuous trajectories to obtain the future motion predictions. It then employs a structured motion planning cost function, which combines a cost learned in a data-driven manner and a cost inspired by human prior knowledge on driving (*e.g.*, traffic rules, collision avoidance) to ensure that the SDVs planned path is safe. We refer the reader to Fig. 1 for an overview of our full model.

We demonstrate the effectiveness of our model on two large-scale real-world datasets: **nuScenes** [6] and **ATG4D**, as well as one simulated dataset **CARLA-Precog** [15, 39]. Our method significantly outperforms previous state-of-the-art results on both prediction and planning tasks.

2   Related Work

Motion Prediction: Two of the main challenges of prediction are modeling interactions among actors and making accurate multi-modal predictions. To address these, [1, 18, 25, 39, 46, 57, 7, 26] learn per-actor latent representations and model interactions by communicating those latent representations among actors. These methods can naturally work with VAE [23] and produce multi-modal predictions. However, they typically lack interpretability and it is hard to encode prior knowledge, such as the traffic participants' desire to avoid collisions.
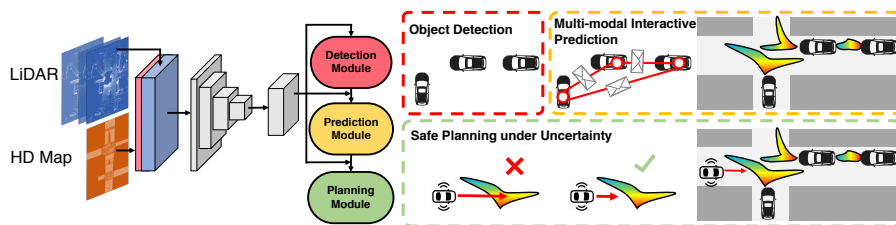
Fig. 1: **DSDNet overview:** The model takes LiDAR and map as inputs, processes them with a CNN backbone, and jointly performs object detection, multimodal socially-consistent prediction, and safe planning under uncertainty. *Rainbow* patterns mean highly likely actors' future positions predicted by our model.

Different from building implicit distributions with VAE, [11, 27] build explicit distributions using mixture of modes (e.g., GMM) where it is easier to perform efficient probabilistic inference. In this work, we further enhance the model capacity with a non-parametric explicit distribution constructed over a dense set of trajectory samples. In concurrent work [36] use similar representation to ours, but they do not model social interactions and do not demonstrate how such a representation can benefit planning.

Recently, a new prediction paradigm of performing joint detection and prediction has been proposed [54, 28, 47, 29, 10, 8, 9], in which actors' location information is not known a-priori, and needs to be inferred from the sensors. In this work, we will demonstrate our approach in both settings: using sensor data or history of actors' locations as input.

Motion Planning: Provided with perception and prediction results, planing is usually formulated as a cost minimization problem over trajectories. The cost function can be either manually engineered to guarantee certain properties [5, 16, 33, 61], or learned from data through imitation learning or inverse reinforcement learning [41, 49, 54, 60]. However, most of these planners assume detection and prediction to be accurate and certain, which is not true in practice. Thus, [2, 19, 40, 56] consider uncertainties in other actors' behaviors, and formulate collision avoidance in a probabilistic manner. Following this line of work, we also conduct uncertainty-aware motion planning.

End-to-end self-driving methods try to fully utilize the power of data-driven approaches and enjoy simple inference. They typically use a neural network to directly map from raw sensor data to planning outputs, and are learned through imitation learning [4, 37], or reinforcement learning [13, 34] when a simulator [15, 30] is available. However, most of them lack interpretability and do not explicitly ensure safety. While our method also benefits from the power of deep learning, in contrast to the aforementioned approaches, we explicitly model interactions between the SDV and the other dynamic agents, achieving safer planning. Furthermore, safety is explicitly accounted for in our planning cost functions.
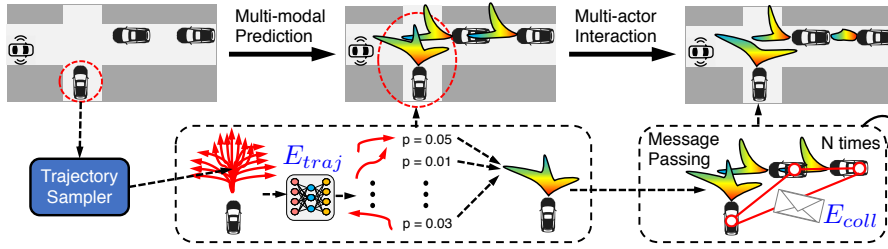
Fig. 2: **Details of the multimodal social prediction module:** For each actor, we sample a set of physically valid trajectories, and use a neural network E_{traj} to assign energies (probabilities) to them. To make different actors' behaviors socially consistent, we employ message passing steps which explicitly model interactions and can encode human prior knowledge (collision avoidance). The final predicted socially-consistent distribution is shown on top right.

Structured models and Belief Propagation: To encode prior knowledge, there is a recent surge of deep structured models [3, 12, 17, 43], which use deep neural networks (DNNs) to provide the energy terms of a probabilistic graphical models (PGMs). Combining the powerful learning capacity of DNNs and the task-specific structure imposed by PGMs, deep structured models have been successfully applied to various computer vision problems, *e.g.*, semantic segmentation [43], anomaly detection [55], contour segmentation [31]. However, for continuous random variables, inference is very challenging. Sample-based belief propagation (BP) [48, 53, 21, 45, 51, 50], address this issue by first constructing the approximation of the continuous distribution via Markov Chain Monte Carlo (MCMC) samples and then performing inference via BP. Inspired by these works, we design a deep structured model that can learn complex human behaviors from large data while incorporating our prior knowledge. We also bypass the difficulty in continuous variable inference using a physically valid sampling procedure.

3   Deep Structured self-Driving Network

Given sensor measurements and a map of the environment, the objective of a self-driving vehicle (SDV) is to select a trajectory to execute (amongst all feasible ones) that is safe, comfortable, and allows the SDV to reach its destination. In order to plan a safe maneuver, a self-driving vehicle has to first understand its surroundings as well as predict how the future might evolve. It should then plan its motion by considering all possibilities of the future weighting them properly. This is not trivial as the future is very multi-modal and actors interact with each other. Moreover, the inference procedure needs to be performed in a fraction of a second in order to have practical value.

In this paper we propose DSDNet, a single neural network that jointly detects actors in the scene, predicts a socially consistent multimodal distribution

over their future behaviors, and produces safe motion plans for the SDV. Fig. 1 gives an overview of our proposed approach. We first utilize a backbone network to compute the intermediate feature-maps, which are then used for detection, prediction and planning. After detecting actors with a detection header, a deep structured probabilistic inference module computes the distributions of actors' future trajectories, taking into account the interactions between them. Finally, our planning module outputs the planned trajectory by considering both the contextual information encoded in the feature-maps as well as possible futures predicted from the model.

In the following, we first briefly explain the input representation, backbone network and detection module. We then introduce our novel probabilistic prediction and motion planning framework in sections 3.2 and 3.3 respectively. Finally, we illustrate how to train our model end-to-end in section 3.4.

3.1 Backbone Feature Network and Object Detection

Let $\mathbf{X}$ be the LiDAR point clouds and the HD map given as input to our system. Since LiDAR point clouds can be very sparse and the actors' motion is an important cue for detection and prediction, we use the past 10 LiDAR sweeps (e.g., 1s of measurements) and voxelize them into a 3D tensor [29, 52, 58, 54]. We utilize HD maps as they provide a strong prior about the scene. Following [54], we rasterize the lanes with different semantics (*e.g.*, straight, turning, blocked by traffic light) into different channels and concatenate them with the 3D LiDAR tensor to form our input representation. We then process this 3D tensor with a deep convolutional network backbone and compute a backbone feature map $\mathbf{F} \in \mathbb{R}^{H \times W \times C}$, where H, W correspond to the spatial resolution after downsampling (backbone) and C is the channel number. We then employ a single-shot detection header on this feature map to output detection bounding boxes for the actors in the scene. We apply two Conv2D layers separately on $\mathbf{F}$, one for classifying if a location is occupied by an actor, the other for regressing the position offset, size, orientation and speed of each actor. Our prediction and planning modules will then take these detections and the feature map as input to produce both a distribution over the actors' behaviors and a safe planning maneuver. For more details on our detector and backbone network please refer to the supplementary material.

3.2 Probabilistic Multimodal Social Prediction

In order to plan a safe maneuver, we need to predict how other actors could potentially behave in the next few seconds. As actors move on the ground, we represent their possible future behavior using a trajectory defined as a sequence of 2D waypoints on birds eye view (BEV) sampled at T discrete timestamps. Note that T is the same duration as our planning horizon, and we compute the motion prediction distribution and a motion plan each time a new sensor measurement arises (i.e., every 100ms).

Output Parameterization: Let $\mathbf{s}_i \in \mathbb{R}^{T \times 2}$ be the future trajectory of the i-th actor. We are interested in modeling the joint distribution of all actors condition on the input, that is $p(\mathbf{s}_1, \cdots, \mathbf{s}_N | \mathbf{X})$. Modeling this joint distribution and performing efficient inference is challenging, as each actor has a high-dimensional continuous action space. Here, we propose to approximate this high-dimensional continuous space with a finite number of samples, and construct a non-parametric distribution over the sampled space. Specifically, for each actor, we randomly sample K possible future trajectory $\{\hat{\mathbf{s}}_i^1, \cdots, \hat{\mathbf{s}}_i^K\}$ from the original continuous trajectory space $\mathbb{R}^{T \times 2}$. We then constrain the possible future state of each actor to be one of those K samples. To ensure samples are always diverse, dense[3] and physically plausible, we follow the **N**eural **M**otion **P**lanner (NMP) [54] and use a combination of straight, circle, and clothoid curves. More details and analysis of the sampler can be found in the supplementary material.

Modeling Future Behavior of All Actors: We employ an energy formulation to measure the probability of each possible future configuration of all actors in the scene: a configuration $(\mathbf{s}_1, \cdots, \mathbf{s}_N)$ has low energy if it is likely to happen. We can then compute the joint distribution of all actors' future behaviors as

$$p(\mathbf{s}_1, \cdots, \mathbf{s}_N | \mathbf{X}, \mathbf{w}) = \frac{1}{Z} \exp\left(-E(\mathbf{s}_1, \cdots, \mathbf{s}_N | \mathbf{X}, \mathbf{w})\right), \qquad (1)$$

where $\mathbf{w}$ are learnable parameters, $\mathbf{X}$ is the raw sensor data and Z is the partition function $Z = \sum \exp(-E(\hat{\mathbf{s}}_1^{k_1}, \cdots, \hat{\mathbf{s}}_N^{k_N}))$ summing over all actors' possible states.

We construct the energy $E(\mathbf{s}_1, \cdots, \mathbf{s}_N | \mathbf{X}, \mathbf{w})$ inspired by how humans drive, e.g., following common sense as well as traffic rules. For example, humans drive smoothly along the road and avoid collisions with each other. Therefore, we decompose the energy E into two terms. The first term encodes the goodness of a future trajectory (independent of other actors) while the second term explicitly encodes the fact that pairs of actors should not collide in the future.

$$E(\mathbf{s}_1, \cdots, \mathbf{s}_N | \mathbf{X}, \mathbf{w}) = \sum_{i=1}^{N} E_{traj}(\mathbf{s}_i | \mathbf{X}, \mathbf{w}_{traj}) + \sum_{i=1}^{N}\sum_{i \neq j}^{N} E_{coll}(\mathbf{s}_i, \mathbf{s}_j | \mathbf{X}, \mathbf{w}_{coll}) \ (2)$$

where N is the number of detected actors and $\mathbf{w}_{traj}$ and $\mathbf{w}_{coll}$ are the parameters.

Since the goodness $E_{traj}(\mathbf{s}_i | \mathbf{X}, \mathbf{w}_{traj})$ is hard to define manually, we use a neural network to learn it from data (see Fig. 3). Given the sensor data $\mathbf{X}$ and a proposed trajectory $\mathbf{s}_i$, the network will output a scalar value. Towards this goal, we first use the detected bounding box of the i-th actor and apply ROIAlign to the backbone feature map, followed by several convolution layers to compute the actor's feature. Note that the backbone feature map is expected to encode rich information about both the environment and the actor.

---

[3] We would like the samples to cover the original continuous space and have high recall wrt the ground-truth future trajectories.

We then index (bilinear interpolation) T features on the backbone feature map at the positions of trajectory's waypoints, and concatenate them together with $(x_t, y_t, cos(\theta_t), sin(\theta_t), distance_t)$ to form the trajectory feature of $\mathbf{s}_i$. Finally, we feed both actor and trajectory features into an MLP which outputs $E_{traj}(\mathbf{s}_i|\mathbf{X}, \mathbf{w}_{traj})$. Fig.. 3 shows an illustration of the architecture.

Fig. 3: Neural header for evaluating E_{traj} and C_{traj}.

We use a simple yet effective collision energy: $E(\mathbf{s}_i, \mathbf{s}_j) = \gamma$ if $\mathbf{s}_i$ collides with $\mathbf{s}_j$, and $E(\mathbf{s}_i, \mathbf{s}_j) = 0$ otherwise, to explicitly model the collision avoidance interaction between actors as explained in the next paragraph. We found this simple pairwise energy empirically works well, the exploration of other learnable pairwise energy is thus left as future work.
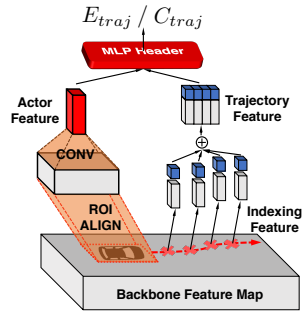
Message Passing Inference: For safety, our motion planner needs to take all possible actor's future into consideration. Therefore, motion forecasting needs to infer the probability of each actor taking a particular future trajectory: $p(\mathbf{s}_i = \hat{\mathbf{s}}_i^k|\mathbf{X}, \mathbf{w})$. We thus conduct marginal inference over the joint distribution. Note that the joint probability defined in Eq. (2) represents a deep structured model (i.e., a Markov random field with potentials computed with deep neural networks). We utilize sum-product message passing [53] to estimate the marginal distribution per actor, taking into account the effects of all other actors by marginalization. The marginal $p(\mathbf{s}_i|\mathbf{X}, \mathbf{w})$ reflects the uncertainty and multimodality in an actor's future behavior and will be leveraged by our planner. We use the following update rule in an iterative manner for each actor ($\mathbf{s}_i$):

$$m_{ij}(\mathbf{s}_j) \propto \sum_{\mathbf{s}_i \in \{\mathbf{s}_i^k\}} e^{-E_{traj}(\mathbf{s}_i) - E_{coll}(\mathbf{s}_i, \mathbf{s}_j)} \prod_{n \neq i,j} m_{ni}(\mathbf{s}_i) \qquad (3)$$

where m_{ij} is the message sent from actor i to actor j and $\propto$ means equal up to a normalization constant. Through this message passing procedure, actors communicate with each others their future intentions $\mathbf{s}_i$ and how probable those intentions are $E_{traj}(\mathbf{s}_i)$. The collision energy E_{coll} helps to coordinate intentions from different actors such that the behaviors are compliant and do not result in collision. After messages have been passed for a fixed number of iterations, we compute the approximated marginal as

$$p(\mathbf{s}_i = \hat{\mathbf{s}}_i^k|\mathbf{X}, \mathbf{w}) \propto e^{-Etraj(\hat{\mathbf{s}}_i^k)} \prod_{j \neq i} m_{ji}(\hat{\mathbf{s}}_i^k). \qquad (4)$$

Since we typically have a small graph (less than 100 actors in a scene) and each $\mathbf{s}_i$ only has K possible values $\{\hat{\mathbf{s}}_i^1, \cdots, \hat{\mathbf{s}}_i^K\}$, we can efficiently evaluate Eq. (3) and Eq. (4) via matrix multiplication on GPUs. In practice we find that our energy

in Eq. (2) usually results in sparse graphs: most actor will only interact with nearby actors, especially the actors in the front and in the back. As a result, the message passing converges within 5 iterations [4]. With our non-highly-optimized implementation, the prediction module takes less than 100 ms on average, and thus it satisfies our real-time requirements.

3.3   Safe Motion Planning under Uncertain Future

The motion planning module fulfills our final goal, that is, navigating towards a destination while avoiding collision and obeying traffic rules. Towards this goal, we build a cost function C, which assigns lower cost values to "good" trajectory proposals and higher values to "bad" ones. Planning is then performed by finding the optimal trajectory with the minimum cost

$$\boldsymbol{\tau}^* = \arg \min_{\boldsymbol{\tau} \in \mathcal{P}} C\left(\boldsymbol{\tau} | p(\mathbf{s}_1, \cdots, \mathbf{s}_N), \mathbf{X}, \mathbf{w}\right), \tag{5}$$

with $\boldsymbol{\tau}^*$ the planned optimal trajectory and $\mathcal{P}$ the set of physically realizable trajectories that do not violate the SDV's dynamics. In practice, we sample a large number of future trajectories for the SDV conditioned on its current dynamic state (*e.g.*, velocity and acceleration) to form $\mathcal{P}$, which gives us a finite set of feasible trajectories $\mathcal{P} = \{\hat{\boldsymbol{\tau}}^1, \cdots, \hat{\boldsymbol{\tau}}^K\}$. We use the same sampler as described in section 3.2 to ensure we get a wide variety of physically possible trajectories.

Planning Cost: Given a SDV trajectory $\boldsymbol{\tau}$, we compute the cost based on how good $\boldsymbol{\tau}$ is 1) conditioned on the scene, (*e.g.*, traffic lights and road topology); 2) considering all other actors' future behaviors (*i.e.*, marginal distribution estimated from the prediction module). We thus define our cost as

$$C(\boldsymbol{\tau} | p(\mathbf{s}_1, \cdots, \mathbf{s}_N), \mathbf{X}, \mathbf{w}) = C_{traj}(\boldsymbol{\tau} | \mathbf{X}, \mathbf{w}) + \sum_{i=1}^{N} \mathbb{E}_{p(\mathbf{s}_i | \mathbf{X}, \mathbf{w})} \left[C_{coll}(\boldsymbol{\tau}, \mathbf{s}_i | \mathbf{X}, \mathbf{w})\right], \tag{6}$$

where C_{traj} models the goodness of a SDV trajectory using a neural network. Similar to E_{traj} in the prediction module, we use the trajectory feature and ROIAlign extracted from the backbone feature map to compute this scalar cost value. The collision cost C_{coll} is designed for guaranteeing safety and avoid collision: *i.e.*, $C_{coll}(\boldsymbol{\tau}, \mathbf{s}_i) = \lambda$ if $\boldsymbol{\tau}$ and $\mathbf{s}_i$ collide, and 0 otherwise. This ensures a dangerous trajectory $\boldsymbol{\tau}$ will incur a very high cost and will be rejected by our cost minimization inference process. Furthermore, Eq. (6) evaluates the expected collision cost $\mathbb{E}_{p(\mathbf{s}_i | \mathbf{X}, \mathbf{w})}[C_{coll}]$. Such a formulation is helpful for safe motion planning since the future is uncertain and we need to consider all possibilities, properly weighted by how likely they are to happen.

---

[4] Although the sum-product algorithm is only exact for tree structures, it is shown to work well in practice for graphs with cycles [35, 48].

Inference: We conduct exact minimization over $\mathcal{P}$. C_{traj} is a neural network based cost and we can evaluate all K possible trajectories with a single batch forward pass. C_{coll} can be computed with a GPU based collision checker. As a consequence, we can efficiently evaluate $C(\boldsymbol{\tau})$ for all K samples and select the trajectory with minimum-cost as our final planning result.

3.4   Learning

We train the full model (backbone, detection, prediction and planning) jointly with a multi-class loss defined as follows

$$\mathcal{L} = \mathcal{L}_{\text{planning}} + \alpha \mathcal{L}_{\text{prediction}} + \beta \mathcal{L}_{\text{detection}}. \tag{7}$$

where α, β are constant hyper-parameters. Such a multi-task loss can fully exploit the supervision for each task and help the training[5].

Detection Loss: We employ a standard detection loss $\mathcal{L}_{detection}$, which is a sum of classification and regression loss. We use a cross-entropy classification loss and assign an anchor's label based on its IoU with any actor. The regression loss is a smooth ℓ_1 between our model regression outputs and the ground-truth targets. Those targets include position, size, orientation and velocity. We refer the reader to the supplementary material for more details.

Prediction Loss: As our prediction module outputs a discrete distribution for each actor's behavior, we employ cross-entropy between our discrete distribution and the true target. as our prediction loss. Once we sampled K trajectories per actor, this loss can be regarded as a standard classification loss over K classes (one for each trajectory sample). The target class is set to be the closest trajectory sample to the ground-truth future trajectory (in ℓ_2 distance).

Planning Loss: We expect our model to assign lower planning costs to better trajectories (*e.g.*, collision free, towards the goal), and higher costs to bad ones. However, we do not have direct supervision over the cost. Instead, we utilize a max-margin loss, using expert behavior as positive examples and randomly sampled trajectories as negative ones. We set large margins for dangerous behaviors such as trajectories with collisions. This allows our model to penalize dangerous behaviors more severely. More formally, our planning loss is defined as

$$\mathcal{L}_{\text{planning}} = \sum_{data} \max_k \left(\left[C\left(\boldsymbol{\tau}^{gt}|\mathbf{X}\right) - C\left(\hat{\boldsymbol{\tau}}^k|\mathbf{X}\right) + d^k + \gamma^k \right]_+ \right),$$

where $[\cdot]_+$ is a ReLU function, $\boldsymbol{\tau}^{gt}$ is the expert trajectory and $\hat{\boldsymbol{\tau}}^k$ is the k-th trajectory sample. We also define d^k as the ℓ_2 distance between $\hat{\boldsymbol{\tau}}^k$ and $\boldsymbol{\tau}^{gt}$, and γ^k is a constant positive penalty if $\hat{\boldsymbol{\tau}}^k$ behave dangerously, *e.g.*, $\gamma_{\text{collision}}$ if $\boldsymbol{\tau}^k$ collides with another actor and 0 otherwise.

---

[5] We find that using only $\mathcal{L}_{planning}$ without the other two terms prevents the model from learning reasonable detection and prediction.

nuScenes	L2 (m)			Col (‰)			ATG4D	L2 (m)			Col (‰)		
Method	1s	2s	3s	1s	2s	3s	Method	1s	2s	3s	1s	2s	3s
Social-LSTM [1]	0.71	-	1.85	0.8	-	9.6	FaF [29]	0.60	1.11	1.82	-	-	-
CSP [14]	0.70	-	1.74	0.4	-	5.8	IntentNet [10]	0.51	0.93	1.52	-	-	-
CAR-Net [42]	0.61	-	1.58	0.4	-	4.9	NMP [54]	0.45	0.80	1.31	0.2	1.1	5.9
NMP [54]	0.43	0.83	1.40	**0.0**	1.4	6.5							
DSDNet	**0.40**	**0.76**	**1.27**	**0.0**	**0.0**	**0.2**	**DSDNet**	**0.43**	**0.75**	**1.22**	**0.1**	**0.1**	**0.2**

(a) Prediction from raw sensor data: ℓ_2 and Col (collision rate), lower the better.

CARLA	DESIRE	SocialGAN	R2P2-MA	MultiPath	ESP (Flex)	MFP-4	**DSDNet**
Town 1	2.422	1.141	0.770	0.68	0.447	0.279	**0.195**
Town 2	1.697	0.979	0.632	0.69	0.435	0.290	**0.213**

(b) Prediction from ground-truth perception: minMSD with K = 12, lower the better.

Table 1: Prediction performance on nuScenes, ATG4D and CARLA

4   Experimental Evaluation

We evaluate our model on all three tasks: detection, prediction, and planning. We show results on two large scale real-world self-driving datasets: **nuScenes** [6] and our in-house dataset **ATG4D**, as well as the **CARLA** simulated dataset [15, 39]. We show that **1)** our prediction module largely outperforms the state-of-the-art on public benchmarks and we demonstrate the benefits of explicitly modeling the interactions between actors. **2)** Our planning module achieves the safest planning results and largely decreases the collision and lane violation rate, compared to competing methods. **3)** Although sharing a single backbone to speedup inference, our model does not sacrifice detection performance compared to the state-of-the-art. We provide datasets' details and implementation details in the supplementary material.

4.1   Multi-modal Interactive Prediction

Baselines: On CARLA, we compare with the state-of-the-art reproduced and reported from [11, 39, 46]. On nuScenes[6], we compare our method against several powerful multi-agent prediction approaches reproduced and reported from [7][7]: **Social-LSTM** [1], **Convolutional Social Pooling (CSP)** [14] and **CAR-Net** [42]. On ATG4D, we compare with LiDAR-based joint detection and prediction models, including **FaF** [29] and **IntentNet** [10]. We also compare with **NMP** [54] on both datasets.

---

[6] Numbers are reported on official validation split, since there is no joint detection and prediction benchmark.

[7] [7] replaced the original encoder (taking the ground-truth detection and tracking as input) with a learned CNN that takes LiDAR as input for a fair comparison.

Model	Collision Rate (%)			Lane Violation Rate (%)			L2 (m)		
	1s	2s	3s	1s	2s	3s	1s	2s	3s
Ego-motion	0.01	0.54	1.81	0.51	2.72	6.73	0.28	0.90	2.02
IL	0.01	0.55	1.72	0.44	2.63	5.38	**0.23**	**0.84**	**1.92**
Manual Cost	0.02	0.22	2.21	0.39	2.73	5.02	0.40	1.43	2.99
Learnable-PLT [41]	0.00	0.13	0.83	-	-	-	-	-	-
NMP [54]	0.01	0.09	0.78	0.35	0.77	2.99	0.31	1.09	2.35
DSDNet	**0.00**	**0.05**	**0.26**	**0.11**	**0.57**	**1.55**	0.29	0.97	2.02

Table 2: **Motion planning performance** on ATG4D. All metrics are computed in a cumulative manner across time, lower the better.

Metrics: Following previous works [1, 18, 25, 29, 54], we report **L2 Displacement Error** between our prediction (most likely) and the ground-truth at different future timestamps. We also report **Collision Rate**, defined as the percentage of actors that will collide with others if they follow the predictions. We argue that a socially consistent prediction model should achieve low collision rate, as avoiding collision is always one of the highest priorities for a human driver. On **CARLA**, we follow [39] and use **minMSD** as our metric, which is the minimal mean squared distance between the top 12 predictions and the ground-truth.

Quantitative Results: As shown in Table 1a, our method achieves the best results on both datasets. This is impressive as most baselines use ℓ_2 as training objective, and thus are directly favored by the ℓ_2 error metric, while our approaches uses cross-entropy loss to learn proper distributions and capture multi-modality. Note that multimodal techniques are thought to score worst in this metric (see e.g., [11]). Here, we show that it is possible to model multi-modality while achieving lower ℓ_2 error, as the model can better understand actors' behavior. Our approach also significantly reduces the collisions between the actors' predicted trajectories, which justifies the benefit of our multi-agent interaction modeling. We further evaluate our prediction performance when assuming ground-truth perception / history are known, instead of predicting using noisy detections from the model. We conduct this evaluation on CARLA where all previous methods use this settings. As shown in Table 1b, our method again significantly outperforms previous best results.

4.2   Motion Planning

Baselines: We implement multiple baselines for comparison, including both neural-based and classical motion planners: **Ego-motion** takes past 1 second positions of the ego-car and use a 4-layer MLP to predict the future locations, as the ego-motion usually providse strong cues of how the SDV will move in the future. **Imitation Learning (IL)** uses the same backbone network as our model but directly regresses an output trajectory for the SDV. We train such a regression model with ℓ_2 loss w.r.t. the ground-truth planning trajectory. **Manual**

nuScenes	Det AP @ meter				ATG4D	Det AP @ IoU			
Method	0.5	1.0	2.0	4.0	Method	0.5	0.6	0.7	0.8
Mapillary[32]	10.2	36.2	64.9	80.1	FaF [29]	89.8	82.5	68.1	35.8
PointPillars [24]	55.5	71.8	76.1	78.6	IntentNet [10]	94.4	89.4	78.8	43.5
NMP [54]	71.7	82.5	85.5	87.0	Pixor [52]	93.4	89.4	78.8	52.2
Megvii [59]	**72.9**	82.5	85.9	**87.7**	NMP [54]	**94.2**	**90.8**	**81.1**	53.7
DSDNet	72.1	**83.2**	**86.2**	87.4	**DSDNet**	92.5	90.2	**81.1**	**55.4**

Table 3: **Detection performance**: higher is better. Note that although our method uses single backbone for multiple challenging tasks, our detection module can achieve on-par performance with the state-of-the-art.

Cost is a classical sampling based motion planner based on a manually designed cost function encoding collision avoidance and route following. The planned trajectory is chosen by finding the trajectory sample with minimal cost. We also include previously published learnable motion planning methods: **Learnable-PLT** [41] and **Neural Motion Planner (NMP)** [54]. These two method utilize a similar max-margin planning loss as ours. However, **Learnable-PLT** only consider the most probable future prediction, while **NMP** assumes planning is independent of prediction given the features.

Metrics: We exploit three metrics to evaluate motion planning performance. **Collision Rate** and **Lane Violation rate** are the ratios of frames at which our planned trajectory either collides with other actors' ground-truth future behaviors, or touches / crosses a lane boundary, up to a specific future timestamp. Those are important safety metrics (lower is better). **L2 to expert path** is the average ℓ_2 distance between the planning trajectory and the expert driving path. Note that the expert driving path is just one among many possibilities, thus rendering this metric not perfect.

Quantitative Results: The planning results are shown in Table 2. We can observe that: **1)** our proposed method provides the safest plans, as we achieve much lower collision and lane violation rates compared to all other methods. **2)** Ego-motion and IL achieves the best ℓ_2 metric, as they employ the power of neural networks and directly optimize the ℓ_2 loss. However, they have high collision rate, which indicates directly mimicking expert demonstrations is still insufficient to learn a safety-aware self-driving stack. In contrast, by learning interpretable intermediate results (detection and prediction) and by incorporating prior knowledge (collision cost), our model can achieve much better results. The later point is further validated by comparing to NMP, which, despite learning detection and prediction, does not explicitly condition on them during planning. **3)** Manual-Cost and Learnable-PLT explicitly consider collision avoidance and traffic rules. However, unlike our approach, they only take the most likely motion forecast into consideration. Consequently, these methods have a higher collision rate than our approach.

Multi-modal	Interaction	Pred Col (‰)	Pred L2 (m)
		4.9	1.28
✓		2.4	**1.22**
✓	✓	**0.2**	**1.22**

(a) Prediction ablation studies on multi-modality and pairwise interaction modeling.

w/ Prediction (Type)	Plan Col (%)	Lane Vio (%)
N/A	0.60	1.64
most likely	0.45	1.60
multi-modal	**0.26**	**1.55**

(b) Motion planning ablation studies on incorporating different prediction results.

(c) Ablation Study for prediction and planning modules

4.3 Object Detection Results

We show our object detection results on **nuScenes** and **ATG4D**. Although we use a single backbone for all three challenging tasks, we show in Table 3 that our model can achieve similar or better results than state-of-the-art LiDAR detectors on both datasets. On **nuScenes**[8], we follow the official benchmark and use detection average precision (AP) at different distance (in meters) thresholds as our metric. Here Megvii [59] is the leading method on the leaderboard at the time of our submission. We use a smaller resolution (Megvii has 1000 pixels on each side while ours is 500) for faster online inference, yet achieve on-par performance. We also conduct experiments on ATG4D. Since our model uses the same backbone as Pixor [52], which only focuses on detection, we demonstrate that a multi-task formulation does not sacrifice detection performance.

4.4 Ablation Study and Qualitative Results

Table 4a compares different prediction modules with the same backbone network. We can see that explicitly modeling a multi-modal future significantly boosts the prediction performance in terms of both collision rate and ℓ_2 error, comparing to a deterministic (unimodal) prediction. The performance is further boosted if the prediction module explicitly models the future interaction between multiple actors, particularly in collision rate. Table 4c compares motion planners that consider different prediction results. We can see that explicitly incorporating future prediction, even only the most likely prediction, will boost the motion planning performance, especially the collision rate. Furthermore, if the motion planner takes multi-modal futures into consideration, it achieves the best performance among the three. This further justifies our model design.

---

[8] We conduct the comparison on the official validation split, as our model currently only focuses on vehicles while the testing benchmark is built for multi-class detection.
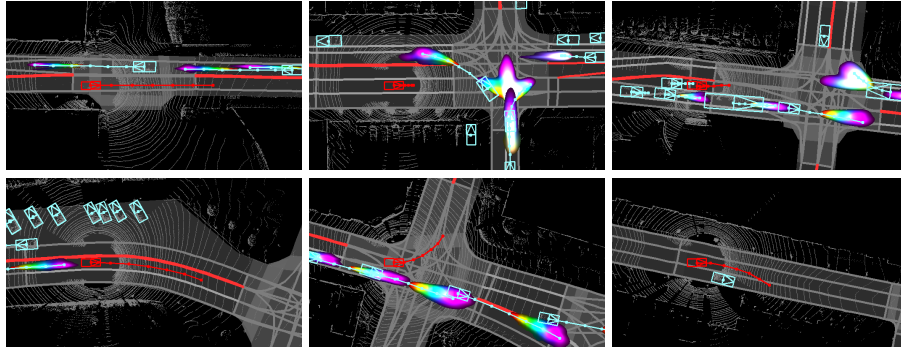
Fig. 4: Qualitative results on ATG4D. Our prediction module can capture multi-modalities when vehicles approach intersections, while being certain and accurate when vehicles drive along a single lane (top). Our model can produce smooth trajectories which follow the lane and are compliant with other actors (bottom). Cyan boxes: detection. Cyan trajectory: prediction. Red box: ego-car. Red trajectory: our planning. We overlay the predicted marginal distribution for different timestamps with different colors and only show high-probability regions.

We show qualitative results in Fig. 4, where we visualize our detections, predictions, motion planning trajectories, and the predicted uncertainties. We use different colors for different future timestamps to visualize high-probability actors' future positions estimated from our prediction module. Thus larger 'rainbow' areas mean more uncertain. On the first row, we can see the predictions are certain when vehicles drive along the lanes (left), while we see multi-modal predictions when vehicles approach an intersection (middle, right). On the second row, we can see our planning can nicely follow the lane (left), make a smooth left turn (middle), and take a nudge when an obstacle is blocking our path (right).

5   Conclusion

In this paper, we propose DSDNet, which is built on top of a single backbone network that takes as input raw sensor data and an HD map and performs perception, prediction, and planning under a unified framework. In particular, we build a deep energy based model to parameterize the joint distribution of future trajectories of all traffic participants. We resort to a sample based formulation, which enables efficient inference of the marginal distribution via belief propagation. We design a structured planning cost which encourages traffic-rule following and collision avoidance for the SDV. We show that our model outperforms the state-of-the-art on several challenging datasets. In the future, we plan to explore more types of structured energies and cost functions.

References

1. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social lstm: Human trajectory prediction in crowded spaces. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 961–971 (2016)
2. Bandyopadhyay, T., Won, K.S., Frazzoli, E., Hsu, D., Lee, W.S., Rus, D.: Intention-aware motion planning. In: Algorithmic foundations of robotics X, pp. 475–491. Springer (2013)
3. Belanger, D., McCallum, A.: Structured prediction energy networks. In: International Conference on Machine Learning. pp. 983–992 (2016)
4. Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J., et al.: End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316 (2016)
5. Buehler, M., Iagnemma, K., Singh, S.: The DARPA urban challenge: autonomous vehicles in city traffic, vol. 56. springer (2009)
6. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. arXiv preprint arXiv:1903.11027 (2019)
7. Casas, S., Gulino, C., Liao, R., Urtasun, R.: Spatially-aware graph neural networks for relational behavior forecasting from sensor data. arXiv preprint arXiv:1910.08233 (2019)
8. Casas, S., Gulino, C., Suo, S., Luo, K., Liao, R., Urtasun, R.: Implicit latent variable model for scene-consistent motion forecasting. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020)
9. Casas, S., Gulino, C., Suo, S., Urtasun, R.: The importance of prior knowledge in precise multimodal prediction. In: IROS (2020)
10. Casas, S., Luo, W., Urtasun, R.: Intentnet: Learning to predict intention from raw sensor data. In: Billard, A., Dragan, A., Peters, J., Morimoto, J. (eds.) Proceedings of The 2nd Conference on Robot Learning. Proceedings of Machine Learning Research, vol. 87, pp. 947–956. PMLR (29–31 Oct 2018), http://proceedings.mlr.press/v87/casas18a.html
11. Chai, Y., Sapp, B., Bansal, M., Anguelov, D.: Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. arXiv preprint arXiv:1910.05449 (2019)
12. Chen, L.C., Schwing, A., Yuille, A., Urtasun, R.: Learning deep structured models. In: International Conference on Machine Learning. pp. 1785–1794 (2015)
13. Codevilla, F., Miiller, M., López, A., Koltun, V., Dosovitskiy, A.: End-to-end driving via conditional imitation learning. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 1–9. IEEE (2018)
14. Deo, N., Trivedi, M.M.: Convolutional social pooling for vehicle trajectory prediction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 1468–1476 (2018)
15. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: Carla: An open urban driving simulator. arXiv preprint arXiv:1711.03938 (2017)
16. Fan, H., Zhu, F., Liu, C., Zhang, L., Zhuang, L., Li, D., Zhu, W., Hu, J., Li, H., Kong, Q.: Baidu apollo em motion planner. arXiv preprint arXiv:1807.08048 (2018)
17. Graber, C., Meshi, O., Schwing, A.: Deep structured prediction with nonlinear output transformations. In: Advances in Neural Information Processing Systems. pp. 6320–6331 (2018)

18. Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., Alahi, A.: Social gan: Socially acceptable trajectories with generative adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2255–2264 (2018)
19. Hardy, J., Campbell, M.: Contingency planning over probabilistic obstacle predictions for autonomous road vehicles. IEEE Transactions on Robotics **29**(4), 913–929 (2013)
20. Hong, J., Sapp, B., Philbin, J.: Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8454–8462 (2019)
21. Ihler, A., McAllester, D.: Particle belief propagation. In: Artificial Intelligence and Statistics. pp. 256–263 (2009)
22. Jain, A., Casas, S., Liao, R., Xiong, Y., Feng, S., Segal, S., Urtasun, R.: Discrete residual flow for probabilistic pedestrian behavior prediction. arXiv preprint arXiv:1910.08041 (2019)
23. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
24. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12697–12705 (2019)
25. Lee, N., Choi, W., Vernaza, P., Choy, C.B., Torr, P.H., Chandraker, M.: Desire: Distant future prediction in dynamic scenes with interacting agents. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 336–345 (2017)
26. Li, L., Yang, B., Liang, M., Zeng, W., Ren, M., Segal, S., Urtasun, R.: End-to-end contextual perception and prediction with interaction transformer. In: IROS (2020)
27. Liang, M., Yang, B., Hu, R., Chen, Y., Liao, R., Feng, S., Urtasun, R.: Learning lane graph representations for motion forecasting. In: ECCV (2020)
28. Liang, M., Yang, B., Zeng, W., Chen, Y., Hu, R., Casas, S., Urtasun, R.: Pnpnet: End-to-end perception and prediction with tracking in the loop. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11553–11562 (2020)
29. Luo, W., Yang, B., Urtasun, R.: Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net
30. Manivasagam, S., Wang, S., Wong, K., Zeng, W., Sazanovich, M., Tan, S., Yang, B., Ma, W.C., Urtasun, R.: Lidarsim: Realistic lidar simulation by leveraging the real world. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11167–11176 (2020)
31. Marcos, D., Tuia, D., Kellenberger, B., Zhang, L., Bai, M., Liao, R., Urtasun, R.: Learning deep structured active contours end-to-end. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8877–8885 (2018)
32. Min Choi, H., Kang, H., Hyun, Y.: Multi-view reprojection architecture for orientation estimation. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 0–0 (2019)
33. Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., Haehnel, D., Hilden, T., Hoffmann, G., Huhnke, B., et al.: Junior: The stanford entry in the urban challenge. Journal of field Robotics **25**(9), 569–597 (2008)
34. Müller, M., Dosovitskiy, A., Ghanem, B., Koltun, V.: Driving policy transfer via modularity and abstraction. arXiv preprint arXiv:1804.09364 (2018)

35. Murphy, K.P., Weiss, Y., Jordan, M.I.: Loopy belief propagation for approximate inference: An empirical study. In: Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence. pp. 467–475. Morgan Kaufmann Publishers Inc. (1999)
36. Phan-Minh, T., Grigore, E.C., Boulton, F.A., Beijbom, O., Wolff, E.M.: Covernet: Multimodal behavior prediction using trajectory sets. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14074–14083 (2020)
37. Pomerleau, D.A.: Alvinn: An autonomous land vehicle in a neural network. In: Advances in neural information processing systems. pp. 305–313 (1989)
38. Rhinehart, N., Kitani, K.M., Vernaza, P.: R2p2: A reparameterized pushforward policy for diverse, precise generative path forecasting. In: European Conference on Computer Vision. pp. 794–811. Springer, Cham (2018)
39. Rhinehart, N., McAllister, R., Kitani, K., Levine, S.: Precog: Prediction conditioned on goals in visual multi-agent settings. arXiv preprint arXiv:1905.01296 (2019)
40. Sadat, A., Casas, S., Ren, M., Wu, X., Dhawan, P., Urtasun, R.: Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020)
41. Sadat, A., Ren, M., Pokrovsky, A., Lin, Y.C., Yumer, E., Urtasun, R.: Jointly learnable behavior and trajectory planning for self-driving vehicles. arXiv preprint arXiv:1910.04586 (2019)
42. Sadeghian, A., Legros, F., Voisin, M., Vesel, R., Alahi, A., Savarese, S.: Car-net: Clairvoyant attentive recurrent network. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 151–167 (2018)
43. Schwing, A.G., Urtasun, R.: Fully connected deep structured networks. arXiv preprint arXiv:1503.02351 (2015)
44. Sohn, K., Lee, H., Yan, X.: Learning structured output representation using deep conditional generative models. In: Advances in neural information processing systems. pp. 3483–3491 (2015)
45. Sudderth, E.B., Ihler, A.T., Isard, M., Freeman, W.T., Willsky, A.S.: Nonparametric belief propagation. Communications of the ACM **53**(10), 95–103 (2010)
46. Tang, Y.C., Salakhutdinov, R.: Multiple futures prediction. arXiv preprint arXiv:1911.00997 (2019)
47. Wang, T.H., Manivasagam, S., Liang, M., Yang, B., Zeng, W., Raquel, U.: V2vnet: Vehicle-to-vehicle communication for joint perception and prediction. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020)
48. Weiss, Y., Pearl, J.: Belief propagation: technical perspective. Communications of the ACM **53**(10), 94–94 (2010)
49. Wulfmeier, M., Ondruska, P., Posner, I.: Maximum entropy deep inverse reinforcement learning. arXiv preprint arXiv:1507.04888 (2015)
50. Yamaguchi, K., Hazan, T., McAllester, D., Urtasun, R.: Continuous markov random fields for robust stereo estimation. In: European Conference on Computer Vision. pp. 45–58. Springer (2012)
51. Yamaguchi, K., McAllester, D., Urtasun, R.: Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In: European Conference on Computer Vision. pp. 756–771. Springer (2014)
52. Yang, B., Luo, W., Urtasun, R.: Pixor: Real-time 3d object detection from point clouds

53. Yedidia, J.S., Freeman, W.T., Weiss, Y.: Understanding belief propagation and its generalizations. Exploring artificial intelligence in the new millennium **8**, 236–239 (2003)
54. Zeng, W., Luo, W., Suo, S., Sadat, A., Yang, B., Casas, S., Urtasun, R.: End-to-end interpretable neural motion planner. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8660–8669 (2019)
55. Zhai, S., Cheng, Y., Lu, W., Zhang, Z.: Deep structured energy based models for anomaly detection. In: International Conference on Machine Learning. pp. 1100–1109 (2016)
56. Zhan, W., Liu, C., Chan, C.Y., Tomizuka, M.: A non-conservatively defensive strategy for urban autonomous driving. In: Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on. pp. 459–464. IEEE (2016)
57. Zhao, T., Xu, Y., Monfort, M., Choi, W., Baker, C., Zhao, Y., Wang, Y., Wu, Y.N.: Multi-agent tensor fusion for contextual trajectory prediction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12126–12134 (2019)
58. Zhou, Y., Tuzel, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4490–4499 (2018)
59. Zhu, B., Jiang, Z., Zhou, X., Li, Z., Yu, G.: Class-balanced grouping and sampling for point cloud 3d object detection. arXiv preprint arXiv:1908.09492 (2019)
60. Ziebart, B.D., Maas, A.L., Bagnell, J.A., Dey, A.K.: Maximum entropy inverse reinforcement learning. In: AAAI. vol. 8, pp. 1433–1438. Chicago, IL, USA (2008)
61. Ziegler, J., Bender, P., Dang, T., Stiller, C.: Trajectory planning for bertha—a local, continuous method. In: Intelligent Vehicles Symposium Proceedings, 2014 IEEE. pp. 450–457. IEEE (2014)