Uncertainty-Aware Estimation of Vehicle Orientation for Self-Driving Applications

Henggang Cui, Fang-Chieh Chou, Jake Charland, Carlos Vallespi-Gonzalez, Nemanja Djuric

***Abstract*— Object detection is a critical component of a self-driving system, tasked with inferring the current states of the surrounding traffic actors. While there exist a number of studies on the problem of inferring the position and shape of vehicle actors, understanding actors' orientation remains a challenge for existing state-of-the-art detectors. Orientation is an important property for downstream modules of an autonomous system, particularly relevant for motion prediction of stationary or reversing actors where current approaches struggle. We focus on this task and present a general method that allows us to more accurately infer vehicle orientations, which we apply to the state-of-the-art models that perform joint object detection and motion prediction. In addition, the approach is able to quantify prediction uncertainty, outputting the probability that the inferred orientation is flipped, which allows for improved motion prediction and safer autonomous operations. Empirical results show the benefits of the approach, obtaining state-of-the-art performance on the open-sourced nuScenes data set.**

I. Introduction

In order to operate safely and efficiently in the real world, a self-driving vehicle (SDV) needs to be able to infer the current state of its surroundings, as well as to predict how this state would change in the near future. This task is addressed by object detection and motion prediction modules, two critical components of an autonomous system [1], [2], [3]. The traditional approach to implement these modules is in a sequential manner [1], where they are trained and run separately. In particular, a detection model processes raw sensor inputs to infer object detections and their states (such as position, dimensions, and velocity) [2], which are in turn used as an input to a motion prediction model that outputs objects' future trajectories as well as the motion uncertainty [4]. Going beyond the sequential approach, researchers recently proposed to combine the two components into a unified, end-to-end model, shown to achieve exemplary performance on both tasks [2], [5]. We use these state-of-the-art models to showcase the benefits of our current work.

Due to its importance for SDV operations, the task of object detection has sparked a lot of interest from the research community [6], leading to significant improvements in the detection performance over the past years. However, when it comes to vehicle detections most of the recent research was focused on estimating a limited set of object states, namely positions and bounding boxes. Nevertheless, object orientation (defined as a direction of the front of the vehicle) is an important piece of information that the autonomous system requires to improve safety and efficiency, allowing better and more accurate motion prediction. Moreover, an important aspect of orientation estimation is modeling of its uncertainty [7], providing a more complete view of the stochastic SDV environment. Understanding the full orientation state is particularly critical in the case of vehicles that are static, slow-moving, or reversing, where behavior uncertainty is high and the common heuristic [8] that computes the orientation from the direction of inferred future trajectories breaks down. This leads to inaccurate state estimation for the surrounding vehicles and may impact the behavior and safety of the SDV. More advanced methods are needed to address these issues, and although the problem of orientation prediction has received some attention recently, accuracy of the current state-of-the-art models remains suboptimal.

The existing work on orientation estimation of an object's bounding box can be split into two categories of methods. The first category, which we refer to as *full-range* methods, estimates the orientation in the $(-180^\circ, 180^\circ]$ range [7], [13], allowing estimation of the exact actor state. The second category, which we refer to as *half-range* methods, only estimates the vehicle orientation in the limited $(-90^\circ, 90^\circ]$ range [5], [9], [10], [11], [12], and can be used in applications where a full understanding of object orientation is not that relevant (such as in pure detection applications). It was shown that the half-range methods achieve better detection performance than the full-range methods. However, by design, they are not able to distinguish the front and back of a bounding box, which is very important for the motion prediction task and the SDV safety. In the experimental section we revisit these two types of methods and provide a detailed evaluation of the pros and cons of both approaches.

In this paper we take the above-mentioned considerations into account and propose a novel uncertainty-aware full-range method for orientation estimation. The approach is able to achieve detection performance that is comparable to or better than the half-range methods, while being able to estimate the actor's full-range orientation as well as the probability that the estimation is flipped from the ground-truth by 180°. We summarize our contributions below:

- we describe a novel method to estimate full-range orientation without losing detection accuracy, as well as to quantify the flipped uncertainty of that estimate;
- we conduct extensive experiments showing that the proposed method outperforms the existing state-of-the-art, and present ablation and uncertainty studies.

Work done while at Uber Advanced Technologies Group (ATG), 50 33rd Street, Pittsburgh, PA 15201; Emails: {hcui2, fchou, jakec, cvallespi, ndjuric}@uber.com

TABLE I: Overview of various orientation estimation methods

Method	References	Full-range	Model output	Loss
`Sin-cos-2x`	[5], [9]	✗	$\sin(2\hat{\theta})$, $\cos(2\hat{\theta})$	$\ell_1\big(\sin(2\hat{\theta})-\sin(2\theta)\big)+\ell_1\big(\cos(2\hat{\theta})-\cos(2\theta)\big)$
`L1-sin`	[10], [11], [12]	✗	$\hat{\theta}$	$\ell_1\big(\sin(\hat{\theta}-\theta)\big)$
`Sin-cos`	[13]	✓	$\sin(\hat{\theta})$, $\cos(\hat{\theta})$	$\ell_1\big(\sin(\hat{\theta})-\sin(\theta)\big)+\ell_1\big(\cos(\hat{\theta}_t)-\cos(\theta)\big)$
`MultiBin`	[7]	✓	$\{\Delta\hat{\theta}_i,\hat{p}_i\}_{i=1}^{n}$	$\mathcal{L}_{\text{multibin}}$ [7]
`Flip-aware`	ours	✓	$\sin(\hat{\theta})$, $\cos(\hat{\theta})$, $\hat{p}_f$	$\mathcal{L}_{\text{final}}$ from equation (5)

II. Related work

A. Object detection for autonomous driving

Detecting objects in SDV's surroundings is a critical task of autonomous systems, required for safe road operations. LiDAR-based approaches have proven to be the workhorse within the research community [6], with a large number of methods proposed recently to address the detection task. Deep neural networks are the most popular choice shown to achieve state-of-the-art performance, and various deep methods differ by how the LiDAR data is presented to the model. The authors of [14] proposed to encode the LiDAR points into a bird's-eye view (BEV) grid, and following a number of convolutional layers directly regress an object's existence probability and its state for each grid cell. If available, additional information can be fused into the BEV grid in a straightforward manner, such as a high-definition map [15] or radar [16]. An alternative approach is to project the inputs into the range view (RV) grid, encoding the sensor data in LiDAR's native representation [17]. The authors of [18] extended the RV-based method with fused camera data, which is also natively captured in a front-view frame.

Recently, detection methods were extended to also solve the prediction task, giving rise to unified, end-to-end approaches operating in the BEV frame [13]. Authors of [2] proposed to also infer detected actor's high-level intents, while authors of MultiXNet [5] introduced a two-stage architecture that achieved state-of-the-art performance in both detection and prediction metrics. In our current work we take MultiXNet as the baseline, and propose a loss formulation that leads to significantly improved orientation accuracy.

B. Orientation estimation

The *full-range* orientation estimation methods attempt to estimate actors' orientations in the full $(-180^\circ, 180^\circ]$ range, as exemplified by the `Sin-cos` approach [13]. This method maps each orientation value θ into two independent targets, $\sin(\theta)$ and $\cos(\theta)$, which are independently trained using two smooth-L1 loss terms, denoted as ℓ_1 (note that the two output values are not guaranteed to be normalized as they are modeled separately). Then, during inference, the final orientation can be computed as $\arctan\big(\sin(\theta),\cos(\theta)\big)$. However, with this setup a flipped orientation prediction (i.e., error of 180°) is penalized by a large loss, and causes the two target outputs to move across the unit circle instead of along the circle as the outputs are not normalized. Indeed, our experiments indicate that this characteristic of the loss hurts the overall performance, leading to suboptimal detection and prediction accuracies. Moreover, as in this paper we consider a LiDAR-based model, estimating the front and back of a bounding box is a challenging problem because the front and back of a vehicle may not be easily distinguishable from the LiDAR point cloud, especially for objects that are further away from the sensor and having just a few LiDAR returns.

To address this issue, several detection models took a step back and proposed the *half-range* methods that estimate orientation in the limited $(-90^\circ, 90^\circ]$ range. Examples of such orientation estimation methods include `Sin-cos-2x` and `L1-sin` approaches. MultiXNet [5] and HDNet [9] used the `Sin-cos-2x` method that represents each orientation value as $\sin(2\theta)$ and $\cos(2\theta)$. Similarly to above-mentioned `Sin-cos`, the two targets are trained independently with the ℓ_1 loss, and the final orientation is computed as $0.5\arctan\big(\sin(2\theta),\cos(2\theta)\big)$. On the other hand, SECOND [10], PointPillars [11], and Zhou et al. [12] used the `L1-sin` method that directly regresses the orientation values trained using the loss $\ell_1\big(\sin(\hat{\theta}-\theta)\big)$. As we show in our evaluation results, the half-range methods achieve better detection accuracies than full-range methods in terms of average precision, yet by design they are not able to accurately estimate the front and back of the vehicles, which is critical for the autonomous driving task.

C. Uncertainty-aware orientation estimation

Beyond predicting the orientation itself, understanding its uncertainty is another important task that allows for safer autonomous operations. Mousavian et al. [7] proposed the `MultiBin` method for multimodal full-range orientation estimation. They proposed to bin the orientations into n overlapping bins, and for each bin to output two values, a probability that the object orientation lies within the bin and the residual angle correction relative to the bin's central angle. They trained the bin probabilities with the cross-entropy loss and the residual angle corrections of the matching bins with the cosine distance loss. In such a way, the model can produce multiple orientation estimations for an actor, along with their probabilities. The method, however, requires one to tune the number of bins as well as their placement, whereas our method does not require any extra hyper-parameters. Moreover, our work estimates a full-range orientation along with its uncertainty without the discretization step, thus simplifying the learning problem. The experimental results will show that our proposed method outperforms the state-of-the-art `MultiBin` method. Table I summarizes the discussed state-of-the-art methods for orientation estimation.

III. METHODOLOGY

While the proposed approach is generic and can be applied to any model architecture, we implemented and evaluated it on top of MultiXNet [5], a state-of-the-art joint object detection and motion prediction model that employs a two-stage architecture. The model takes as input a total of T current and historical LiDAR sweeps along with a high-definition map of SDV's surroundings, which are rasterized onto a BEV grid. The method then applies a multi-scale convolutional network on the resulting raster, outputting existence probability and bounding box for each grid cell which completes the first stage. As a part of the second stage, the feature maps corresponding to detected objects are cropped and further processed by a sequence of convolutional layers, eventually outputting each actor's future trajectories for a total of H time steps. The actors' bounding boxes are parameterized by center position, width and height dimensions, and orientation, where for the orientation the model predicts the yaw component for all future time steps, denoted as $\{\hat{\theta}_t\}_{t=1}^{H}$. For more detailed discussion we refer the reader to [5], omitted here due to space limitations.

The original MultiXNet model uses the `Sin-cos-2x` orientation estimation method that represents the orientations as $2H$ independent targets $\{\sin(2\hat{\theta}_t), \cos(2\hat{\theta}_t)\}_{t=1}^{H}$, which are trained with a *half-range loss* given as

$$\begin{aligned}\mathcal{L}_{\text{half}} = \sum_{t=1}^{H} &\ell_1\big(\sin(2\hat{\theta}_t) - \sin(2\theta_t)\big) \\ &+ \ell_1\big(\cos(2\hat{\theta}_t) - \cos(2\theta_t)\big),\end{aligned} \tag{1}$$

where $\{\theta_t\}_{t=1}^{H}$ represent actor's ground-truth orientations. Output values of $\sin(2\hat{\theta}_t)$ and $\cos(2\hat{\theta}_t)$ targets are not guaranteed to be normalized, and the final orientation can be computed during inference as $\hat{\theta}_t = 0.5 \arctan\big(\sin(2\hat{\theta}_t), \cos(2\hat{\theta}_t)\big)$, given in the $(-90^\circ, 90^\circ]$ range.

A. Combining half-range and full-range losses

In order to estimate the full range orientations, our proposed method represents the orientations as $\{\sin(\hat{\theta}_t), \cos(\hat{\theta}_t)\}_{t=1}^{H}$. Then, similarly to the `Sin-cos` method [13], we define a *full-range loss* computed as

$$\begin{aligned}\mathcal{L}_{\text{full}} = \sum_{t=1}^{H} &\ell_1\big(\sin(\hat{\theta}_t) - \sin(\theta_t)\big) \\ &+ \ell_1\big(\cos(\hat{\theta}_t) - \cos(\theta_t)\big).\end{aligned} \tag{2}$$

To bridge the detection performance gap between full-range and half-range orientation estimation methods, we propose to extend (2) with an additional half-range loss term from (1), where the half-range representation parameters in (1) can be computed from the full-range parameters using the following trigonometric identities,

$$\begin{aligned}\sin(2\hat{\theta}_t) &= 2\sin(\hat{\theta}_t)\cos(\hat{\theta}_t), \\ \cos(2\hat{\theta}_t) &= \cos^2(\hat{\theta}_t) - \sin^2(\hat{\theta}_t).\end{aligned} \tag{3}$$

As will be shown in the evaluation results presented in Section IV-B, combining the losses leads to significant improvements of the model performance.

B. Flipping-aware orientation prediction

Even with the combined loss, the model may still incur a high penalty from the $\mathcal{L}_{\text{full}}$ component when outputting an orientation that is flipped by 180°. This is a particular problem for static or slow-moving objects, where historical LiDAR sweeps provide little evidence of their orientation. To mitigate this problem, we propose a novel formulation that renders the full-range loss *flipping-aware*.

In particular, in addition to the full-range orientation outputs $\{\sin(\hat{\theta}_t), \cos(\hat{\theta}_t)\}_{t=1}^{H}$, the model is trained to also predict a probability that the orientation is flipped by 180°, denoted by $\hat{p}_f$. In other words, $\hat{p}_f$ is an indicator of whether $\arctan\big(\sin(\hat{\theta}_t), \cos(\hat{\theta}_t)\big)$ or $\arctan\big(-\sin(\hat{\theta}_t), -\cos(\hat{\theta}_t)\big)$ is the true orientation of the bounding box. This value can then be used by the downstream SDV modules to account for the possibility that an actor would move in either direction, and to modulate SDV's behavior according to the inferred flipped probability to ensure safe vehicle operations.

In order to train such a model, we define a *flipped full-range loss* $\mathcal{L}_{\text{flipped}}$ as given here,

$$\begin{aligned}\mathcal{L}_{\text{flipped}} = \sum_{t=1}^{H} &\ell_1\big(-\sin(\hat{\theta}_t) - \sin(\theta_t)\big) \\ &+ \ell_1\big(-\cos(\hat{\theta}_t) - \cos(\theta_t)\big).\end{aligned} \tag{4}$$

Then, for each actor we compare the values of $\mathcal{L}_{\text{full}}$ and $\mathcal{L}_{\text{flipped}}$ losses and use $p_f = \mathbb{1}_{\mathcal{L}_{\text{full}} > \mathcal{L}_{\text{flipped}}}$ as the ground-truth classification label to train the flipped classification output $\hat{p}_f$, where $\mathbb{1}_{cond}$ is an indicator function equaling 1 if condition *cond* is true and 0 otherwise. Then, we define our final loss as follows,

$$\begin{aligned}\mathcal{L}_{\text{final}} = &\mathcal{L}_{\text{half}} + \min(\mathcal{L}_{\text{full}}, \mathcal{L}_{\text{flipped}}) \\ &+ \text{CrossEntropy}(\hat{p}_f, \mathbb{1}_{\mathcal{L}_{\text{full}} > \mathcal{L}_{\text{flipped}}}).\end{aligned} \tag{5}$$

Note that the loss penalizes only the minimum of $\mathcal{L}_{\text{full}}$ and $\mathcal{L}_{\text{flipped}}$. Thus, if the model predicts a flipped orientation for the bounding box, it will not be penalized by the orientation loss but by the flipped classification loss instead. As a result, when the model predicts a flipped orientation it will be encouraged to keep pushing the orientation closer to the flipped ground-truth orientation while moving the flipped probability closer to 1. Since the model only needs to predict one extra output value for each actor indicating flipped probability, our method has a negligible impact on the model inference speed. Lastly, following the training completion we implement a post-processing step to flip the orientations whose flipped probabilities are greater than 0.5, where we also update their flipped probabilities as $\hat{p}_f \leftarrow 1 - \hat{p}_f$.

C. Loss analysis

In Fig. 1 we provide an analysis of the loss landscapes of various orientation losses considered in this work, with the ground truth corresponding to $\big(\sin(\theta), \cos(\theta)\big) = (0, 1)$. For simplicity, the provided plots do not include the cross-entropy loss term introduced in (5).

In Fig. 1(a) we can see that $\mathcal{L}_{\text{full}}$ from (2) has only one minimum at the ground-truth point $(0, 1)$, and has a

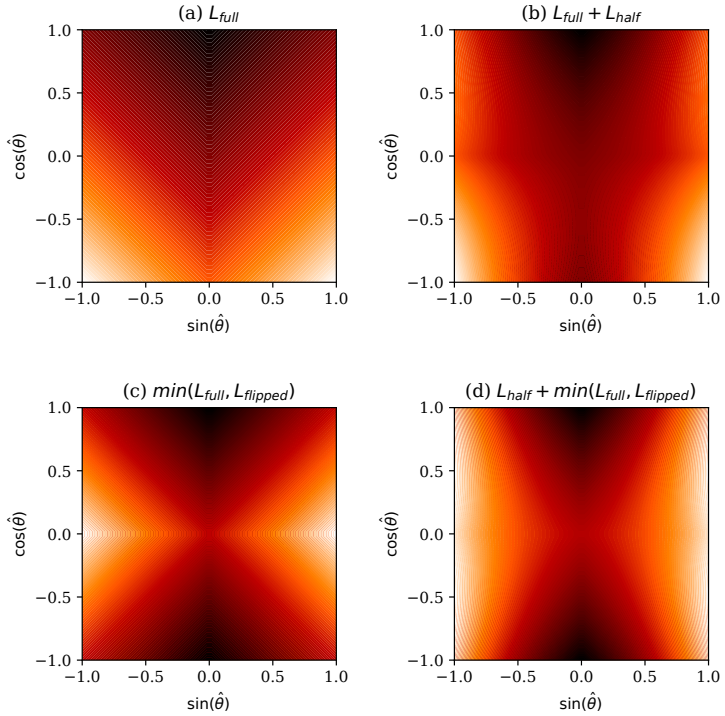

Fig. 1: Loss visualization, with darker color indicating lower loss and ground truth corresponding to $\big(\sin(\theta), \cos(\theta)\big) = (0, 1)$

high loss at the point $(0, -1)$ which corresponds to the 180°-flipped orientation. On the other hand, in Fig. 1(b) we illustrate $\mathcal{L}_{\text{full}} + \mathcal{L}_{\text{half}}$ introduced in Section III-A, which mitigates this issue with the addition of the half-range loss. It has two local minima, the ground truth at $(0, 1)$ and the flipped ground-truth point $(0, -1)$, where $(0, 1)$ is the global minimum. Note that orientation error of 180° at $(0, -1)$ has lower loss than lower-degree errors (e.g., consider $(1, 0)$ that corresponds to orientation error of 90°), which encourages the model to prioritize estimation of half-range over the full-range orientation in the case of errors larger than 90°.

When it comes to the losses $\min(\mathcal{L}_{\text{full}}, \mathcal{L}_{\text{flipped}})$ and $\min(\mathcal{L}_{\text{full}}, \mathcal{L}_{\text{flipped}}) + \mathcal{L}_{\text{half}}$ introduced in Section III-B and illustrated in Fig. 1(c) and Fig. 1(d), respectively, both have two global minima located at the ground-truth point $(0, 1)$ and the flipped ground-truth point $(0, -1)$. As a result, they incur no penalty for 180°-errors and instead rely on the cross-entropy term to encourage the model to distinguish the front and back of the bounding box.

IV. Evaluation

A. Experimental setup

Data set. We evaluated our method on the public nuScenes data set [19]. The data contains 1,000 scenes collected from public roads in Boston and Singapore, sampled into 390,000 frames at 20Hz, where we used the official partition for training and validation sets.

Implementation details. Our MultiXNet implementation used the same hyper-parameters as the original MultiXNet model [5]. The rasterized BEV images have shape $L = 100$m, $W = 100$m, $V = 8$m, with resolution $\Delta_L = 0.125$m, $\Delta_W = 0.125$m, $\Delta_V = 0.2$m. The model input is a history of 0.5s LiDAR sweeps ($T = 10$ frames at 20Hz) and the rasterized map, and the model outputs bounding boxes and predicts their trajectories for 3s into the future ($H = 30$ at 10Hz).

Evaluation metrics. We used the standard KITTI [20] object detection and prediction metrics, including average orientation similarity (AOS), average precision (AP), orientation errors, as well as trajectory ℓ_2 displacement errors [21]. While MultiXNet detects and predicts trajectories for all actor types, in this work we focus only on the vehicle actors. Nevertheless, we note that the metrics for other actor classes are mostly unchanged. Following [5] we used an intersection-over-union (IoU) threshold of 0.7 when computing the AP. However, note that AP is IoU-based and a flipped bounding box will still be counted as a true positive. For that reason, we also considered the AOS metric which weighs the precision by the average cosine distance of the orientations (normalized to a $[0, 1]$ range) at each recall point. As a result, the AP metric is by definition an upper bound of AOS, and a completely flipped bounding box will have no positive contribution towards AOS. We also measure the orientation error of the bounding boxes at 0s and the future trajectory ℓ_2 error at 3s of the true positive detections, with the operating point set at 80% recall using an IoU threshold of 0.5, as done in previous works [5], [22].

For the orientation error we measure both full-range (FOE) and half-range orientation error (HOE), defined as

$$\begin{aligned} \text{FOE} &= |(\theta_0 - \hat{\theta}_0) \mod 360^\circ|, \\ \text{HOE} &= |(\theta_0 - \hat{\theta}_0) \mod 180^\circ|. \end{aligned} \tag{6}$$

When evaluating the orientation and ℓ_2 errors, we also report

TABLE II: Quantitative comparison of the competing approaches; confidence intervals are computed over 3 runs

			Orientation error [degree] ↓			ℓ_2@3s [m] ↓	
			Half-range	Full-range			
Method	**AOS$_{0.7}$ ↑**	**AP$_{0.7}$ ↑**	**All**	**All**	**Moving**	**All**	**Moving**
`Sin-cos-2x`	40.7 ± 0.7	**60.8 ± 1.0**	**1.72 ± 0.04**	59.9 ± 1.4	4.7 ± 0.5	0.99 ± 0.02	2.80 ± 0.06
`L1-sin`	39.8 ± 0.1	59.5 ± 0.1	2.06 ± 0.10	60.2 ± 0.4	5.0 ± 0.1	**0.97 ± 0.01**	2.76 ± 0.03
`Sin-cos`	55.1 ± 0.5	57.1 ± 0.5	2.32 ± 0.01	**8.2 ± 0.2**	2.4 ± 0.2	1.01 ± 0.02	2.79 ± 0.06
`MultiBin-2`	55.0 ± 0.3	57.3 ± 0.6	2.54 ± 0.01	9.4 ± 0.7	3.0 ± 0.1	1.01 ± 0.01	**2.73 ± 0.05**
`MultiBin-4`	55.5 ± 0.3	58.0 ± 0.5	2.14 ± 0.07	9.8 ± 0.5	2.6 ± 0.3	1.01 ± 0.01	2.78 ± 0.04
`Flip-aware`	**57.9 ± 0.4**	**60.7 ± 0.2**	**1.71 ± 0.04**	9.6 ± 0.8	**2.2 ± 0.1**	0.99 ± 0.01	**2.75 ± 0.03**

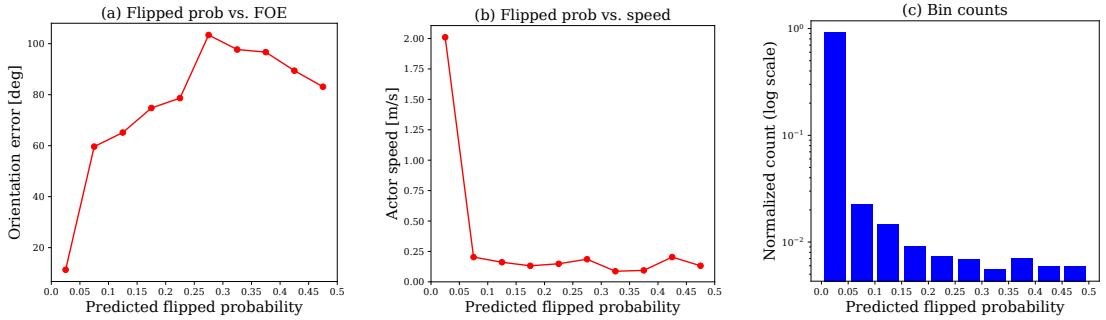

Fig. 2: Analysis of the orientation uncertainty outputs of the proposed `Flip-aware` method

the metrics computed only on moving actors. We define an actor as moving if its ground-truth speed is larger than 0.5m/s, computed as a difference between the ground-truth waypoints at 0.5s and 0s. Lastly, for methods that output multiple orientations, such as our method and `MultiBin` [7], we measure the errors of the highest-probability output.

Baselines. We compared the proposed method against the other state-of-the-art approaches listed in Table I, all added on top of the original MultiXNet losses. For `MultiBin` [7] we used $n = 2$ and $n = 4$ bins (denoted as `MultiBin-n`), found in the original paper to result in the best performance. The bins are centered at $\{0°, 180°\}$ for $n = 2$ and $\{-90°, 0°, 90°, 180°\}$ for $n = 4$. Lastly, in order to fairly evaluate FOE for `L1-sin` and `Sin-cos-2x`, we used a direction of the predicted trajectories to convert their half-range orientations to the full 360°-range.

B. Quantitative results

The comparison of different approaches on the nuScenes data is given in Table II, where the proposed method is denoted as `Flip-aware`. We trained all models three times and report the means and standard deviations of the metrics, marking the best-performing methods in bold.

We can see that `Sin-cos-2x` had the best AP performance, and `L1-sin` had the best ℓ_2 performance. However, they were not able to represent full-range orientations as seen by large orientation errors. Even when using the trajectory predictions to infer the orientations, their FOEs were still very high, especially when it comes to the non-moving actors. As a result, they exhibited very low AOS.

When it comes to the full-range methods, `Sin-cos` had the lowest FOE. However, it also had the worst AP performance among all the considered methods, trailing `Sin-cos-2x` by about 4%. This could be explained by the fact that the network incurs high losses when predicting a flipped bounding box, which causes the model to not spend enough learning capacity in accurately estimating the object detections. The `MultiBin-n` models had similar performance compared to `Sin-cos`. We hypothesize that MultiBin is not able to achieve better performance for our application as it was originally designed for 3D orientation estimation from camera images, while we estimate the object orientations from BEV voxels.

We can see that the proposed `Flip-aware` method achieved significant improvement in AP and HOE compared to the other full-range methods, and it achieved the best AOS, AP, HOE, and moving FOE and ℓ_2 metrics among all the methods. This demonstrates that our proposed loss design allows for very accurate full-range orientation estimation while achieving detection performance comparable to the best half-range methods. In Section IV-D we investigate the reasons behind such performance, where we show that both the addition of the $\mathcal{L}_{\text{half}}$ loss term and the flipping-aware orientation estimation contributed to the improvements. Moreover, our method is able to predict the probability that the orientation is actually flipped, helping the downstream modules to reason about scene uncertainties and allowing safer SDV operations.

C. Analysis of uncertainty outputs

In this section we present the analysis of uncertainty outputs of the proposed `Flip-aware` method. We binned all actors by their predicted flipped probabilities, and for each bin we report the average FOE metric, as well as average speed and actor counts, illustrated in Fig. 2. Note that, due

TABLE III: Ablation study of the proposed `Flip-aware` method

			Orientation error [degree] ↓				
			Half-range	Full-range		ℓ_2@3s [m] ↓	
Method	**AOS$_{0.7}$ ↑**	**AP$_{0.7}$ ↑**	**All**	**All**	**Moving**	**All**	**Moving**
`Flip-aware`	**57.9**	**60.7**	**1.71**	9.6	**2.2**	**0.99**	2.75
`*-no-half`	55.2	59.6	2.07	10.1	2.8	1.03	2.81
`*-no-flip`	56.8	58.9	1.79	**7.8**	2.3	**0.99**	**2.70**

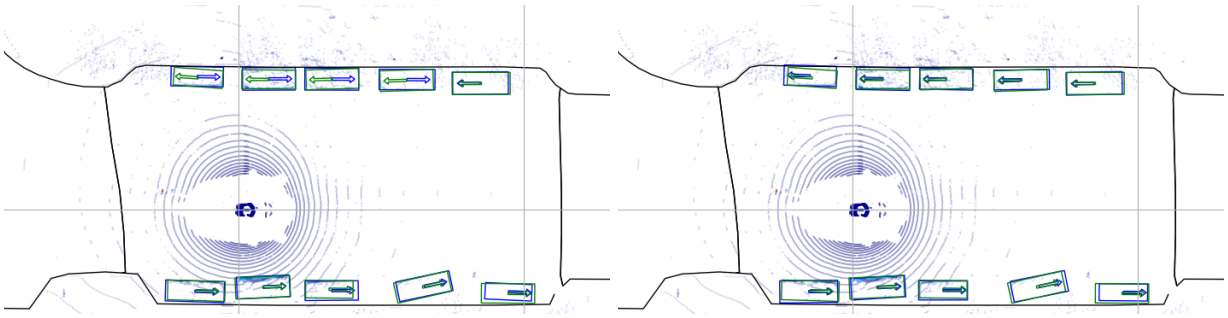

Fig. 3: Outputs of `Sin-cos-2x` and `Flip-aware`, respectively; ground truth shown in green and outputs shown in blue

to the post-processing step discussed in Section III-A, flipped probabilities of all actors are no greater than 0.5.

Results in Fig. 2(a) show that when the model predicts an actor to have a low flipped probability, its FOE is also expected to be significantly lower than for those actors that have higher probabilities. We can conclude that the predicted flipped probability is strongly correlated with the expected error, and is a reliable measure of our uncertainty in the actor's orientation. In addition, in Fig. 2(b) and Fig. 2(c) we present an average actor speed and normalized actor count for each bin, respectively. As expected, the model predicts lower flipped probabilities for moving actors than for non-moving actors, since the observed direction of motion is a strong indicator of actor orientation. Moreover, for most of the actors the model outputs low uncertainty, mirroring the distribution of the actor speeds in the data set.

D. Ablation study

To better understand the effectiveness of the two components of the proposed `Flip-aware` method, we performed an ablation study with two variants of the method, namely `Flip-aware-no-half` that does not include $\mathcal{L}_{\text{half}}$ loss term in (5), and `Flip-aware-no-flip` that does not model uncertainty and instead uses $\mathcal{L}_{\text{full}} + \mathcal{L}_{\text{half}}$ as its loss. The experimental results are given in Table III, showing that both models underperformed when considering the AOS, AP, and HOE metrics, as compared to `Flip-aware`. This demonstrates that both half-range loss and the uncertainty-aware component contributed to the detection performance improvements of our method. We can also see that `Flip-aware-no-flip` had better FOE performance. However, its AOS was lower and, importantly, the method does not provide the uncertainty outputs.

E. Qualitative results

In this section we present the qualitative comparison of the two best performing half-range and full-range methods, namely `Sin-cos-2x` and `Flip-aware`. As all methods performed reasonably well on moving actors, we focused on a scene with a large number of non-moving actors, shown in Fig. 3. As can be seen in the example, the `Sin-cos-2x` method estimated incorrect orientations for a large number of parked vehicles. This occurred because the method is trained to be indifferent to orientation flip through the design of its loss function, and moreover these parked vehicles have no moving trajectory predictions that could be used to reliably infer the orientations. On the other hand, the proposed `Flip-aware` model predicted correct orientations for all vehicles in the scene, which is a result consistent with the previously seen strong performance in the quantitative evaluation. We emphasize that correct orientation prediction for such parked and slow-moving vehicles is an important task for autonomous driving, leading to better behavioral modeling and improved safety during SDV operations.

V. Conclusion

We considered the problem of object detection and motion prediction in the context of the self-driving technology, focusing on an important task of orientation prediction for vehicle actors. This is critical for a full understanding of the actors' states and their motion prediction, especially for slow-moving and stopped vehicles, which directly impacts the behavior and safety of the SDV. In addition to improved orientation prediction, the proposed approach also quantifies the prediction uncertainty by inferring the flipped probability, which is useful information for downstream modules in the SDV system. Experiments on the real-world, open-source nuScenes data indicate the benefits of the proposed method.

REFERENCES

[1] C. Urmson and W. Whittaker, "Self-driving cars and the urban challenge," *IEEE Intelligent Systems*, vol. 23, no. 2, pp. 66–68, 2008.

[2] S. Casas, W. Luo, and R. Urtasun, "Intentnet: Learning to predict intention from raw sensor data," in *Conference on Robot Learning*, 2018, pp. 947–956.

[3] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "LaserNet: An efficient probabilistic 3D object detector for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[4] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, N. Singh, and J. Schneider, "Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020.

[5] N. Djuric, H. Cui, Z. Su, S. Wu, H. Wang, F.-C. Chou, L. S. Martin, S. Feng, R. Hu, Y. Xu, A. Dayan, S. Zhang, B. C. Becker, G. P. Meyer, C. Vallespi-Gonzalez, and C. K. Wellington, "Multixnet: Multiclass multistage multimodal motion prediction," *arXiv preprint arXiv:2006.02000*, 2020.

[6] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3d point clouds: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[7] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3d bounding box estimation using deep learning and geometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7074–7082.

[8] H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, J. Schneider, D. Bradley, and N. Djuric, "Deep kinematic models for kinematically feasible vehicle trajectory predictions," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 563–10 569.

[9] B. Yang, M. Liang, and R. Urtasun, "Hdnet: Exploiting hd maps for 3d object detection," in *Conference on Robot Learning*, 2018, pp. 146–155.

[10] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018.

[11] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 697–12 705.

[12] Y. Zhou, P. Sun, Y. Zhang, D. Anguelov, J. Gao, T. Ouyang, J. Guo, J. Ngiam, and V. Vasudevan, "End-to-end multi-view fusion for 3d object detection in lidar point clouds," in *Conference on Robot Learning*, 2020, pp. 923–932.

[13] W. Luo, B. Yang, and R. Urtasun, "Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 3569–3577.

[14] B. Yang, W. Luo, and R. Urtasun, "Pixor: Real-time 3d object detection from point clouds," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 7652–7660.

[15] B. Yang, M. Liang, and R. Urtasun, "Hdnet: Exploiting hd maps for 3d object detection," in *Conference on Robot Learning*, vol. 87, 2018, pp. 146–155.

[16] B. Yang, R. Guo, M. Liang, S. Casas, and R. Urtasun, "Radarnet: Exploiting radar for robust perception of dynamic objects," *arXiv preprint arXiv:2007.14366*, 2020.

[17] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "Lasernet: An efficient probabilistic 3d object detector for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 677–12 686.

[18] G. P. Meyer, J. Charland, D. Hegde, A. Laddha, and C. Vallespi-Gonzalez, "Sensor fusion for joint 3d object detection and semantic segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[19] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 621–11 631.

[20] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[21] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 2090–2096.

[22] S. Casas, C. Gulino, R. Liao, and R. Urtasun, "Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9491–9497.