

Motion Inspired Unsupervised Perception and Prediction in Autonomous Driving

Mahyar Najibi*, Jingwei Ji*, Yin Zhou*[†], Charles R. Qi, Xinchun Yan, Scott Ettinger, and Dragomir Anguelov

Waymo LLC

{najibi,jingweij,yinzhou,rqi,xcyan,settinger,dragomir}@waymo.com

Abstract. Learning-based perception and prediction modules in modern autonomous driving systems typically rely on expensive human annotation and are designed to perceive only a handful of predefined object categories. This closed-set paradigm is insufficient for the safety-critical autonomous driving task, where the autonomous vehicle needs to process arbitrarily many types of traffic participants and their motion behaviors in a highly dynamic world. To address this difficulty, this paper pioneers a novel and challenging direction, *i.e.*, training perception and prediction models to understand *open-set* moving objects, with no human supervision. Our proposed framework uses self-learned flow to trigger an automated meta labeling pipeline to achieve automatic supervision. 3D detection experiments on the Waymo Open Dataset show that our method significantly outperforms classical unsupervised approaches and is even competitive to the counterpart with supervised scene flow. We further show that our approach generates highly promising results in open-set 3D detection and trajectory prediction, confirming its potential in closing the safety gap of fully supervised systems.

Keywords: Autonomous driving, unsupervised learning, generalization, detection, motion prediction, scene understanding

1 Introduction

Modern 3D object detection [68,61,102,112] and trajectory prediction models [10,32,51,104] are often designed to handle a predefined set of object types and rely on costly human annotated datasets for their training. While such paradigm has achieved great success in pushing the capability of autonomy systems, it has difficulty in generalizing to the *open-set* environment that includes a long-tail distribution of object types far beyond the predefined taxonomy. Towards solving the 3D object detection and behavior prediction of those open-set objects, an alternative and potentially more scalable approach to supervised training is unsupervised perception and prediction.

* Equal contribution

[†] Corresponding author

One central problem in autonomous driving is perceiving the amodal shape of moving objects in space and forecasting their future trajectories, such that the planner and control systems can maneuver safely. As motion estimation (also known as the scene flow problem) is a fundamental task agnostic to the scene semantics [50], it provides an opportunity to address the problem of perception and prediction of open-set moving objects, without any human labels. This leads to our motion-inspired unsupervised perception and prediction system.

Using only LiDAR, our system decomposes the unsupervised, open-set learning task to two steps, as shown in Fig. 1: (1) Auto Meta Labeling (AML) assisted by scene flow estimation and temporal aggregation, which generates pseudo labels of any moving objects in the scene; (2) Training detection and trajectory prediction models based on the auto meta labels. Realizing such an automatic supervision, we transform the challenging open-set learning task to a known, well-studied task of supervised detection or behavior prediction model training.

To derive high-quality auto meta labels, we propose two key technologies: an unsupervised scene flow estimation model and a flow-based object proposal and concept construction approach. Most prior works on unsupervised scene flow estimation [96,45,55,59] optimize for the overall flow quality without specifically focusing on the moving objects or considering the usage of scene flow for onboard perception and prediction tasks. For example, the recently proposed Neural Scene Flow Prior (NSFP) [45] achieved state-of-the-art performance in overall scene flow metrics by learning to estimate scene flow through run-time optimization, without any labels. However, there are too many false positive flows generated for the background, which makes it not directly useful for flow-based object discovery. To tackle its limitations, we extend NSFP to a novel, more accurate and scalable version termed NSFP++. Based on the estimated flow, we propose an automatic pipeline to generate proposals for all moving objects and reconstruct the object shapes (represented as amodal 3D bounding boxes) through tracking, shape registration and refinement. The end product of the process is a set of 3D bounding boxes and tracklets. Given the auto labels, we can train top-performing 3D detection models to localize the open-set moving objects and train behavior prediction models to forecast their trajectories.

Evaluated on the Waymo Open Dataset [75], we show that our unsupervised and data-driven method significantly outperforms non-parametric clustering based approaches and is even competitive to supervised counterparts (using ground truth scene flow). More importantly, our method substantially extends the capability of handling open-set moving objects for 3D detection and trajectory prediction models, leading to a safety improved autonomy system.

2 Related Works

LiDAR-based 3D Object Detection. Fully supervised 3D detection based on point clouds has been extensively studied in the literature. Based on their input representation, these detectors can be categorized as those operating directly on the points [68,61,102,69,54,46], on a voxelized space [21,87,73,56,100,70,112,43,89,103,109], a perspective projection of the

scene [53,5,27], or a combination of these representations [76,12,111,34,67]. Semi-supervised 3D detection with a smaller labeled training set or under the annotator-in-the-loop setting have also been considered in [62,7,99]. However, unsupervised 3D detection has been mostly unexplored due to the inherent problem complexity. More recently, Tian *et al.* [78] proposed to use 3D point cloud clues to perform unsupervised 2D detection in images. In contrast, in our paper, we propose a novel method which performs 3D detection of moving objects in an unsupervised manner.

Scene Flow Estimation. Most previous learning-based works for 3D point cloud scene flow estimation were supervised [47,91,60,33]. More recently, the unsupervised setting has been also studied. [55] used self-supervised cycle consistency and nearest-neighbour losses to train a flow prediction network. In contrast, [45] took an inference-time optimization approach and trained a network per scene. We follow [45] to build our scene flow module given its unsupervised nature and relatively better performance. However, our analysis reveals the limitations of this method in handling complex scenes, making its direct adaptation for proposing high-quality auto labels challenging. In our paper, we noticeably improve the performance of this method by proposing novel techniques to better capture the locality constraints of the scene and to reduce its false predictions.

Unsupervised Object Detection. Existing efforts have been concentrated in the image and video domain, mostly evaluated on object-centric datasets or datasets containing only a handful of object instances per frame. These include statistic-based methods [71,65], visual similarity-based clustering methods [29,26,40], linkage analysis [42] with appearance and geometric consistency [15,84,85,86], visual saliency [105,39], and unsupervised feature learning using deep generative models [44,72,63,3]. In contrast, unsupervised object detection from LiDAR sequences is fairly under-explored [18,94,78,48]. [18,57] proposed to sequentially update the detections and perform tracking based on motion cues from 3D LiDAR scans. Cen *et al.* [9] used predictions of a supervised detector to yield proposals of unknown categories. However, this approach is inapplicable to unsupervised settings and works for unknown categories with close semantics to the known ones. Wong *et al.* [94] introduced a bottom-up approach to segment both known and unknown object instances by clustering and aggregating LiDAR points in a single frame based on their embedding similarities. In comparison, our work leverages both motion cues and point locations for clustering, which puts more emphasis on detecting motion coherent objects and is able to generate amodal bounding boxes.

Shape Registration. Shape registration has been an important topic in vision and graphics community for decades, spanning from classical methods including Iterative Closest Point (ICP) [4,13,64,30] and Structure-from-Motion (SfM) [79,1,38,66] to their deep learning variants [90,82,110,77,92,98,97,81,80,113,37,101]. These methods usually work under the assumption that the object or scene to register is mostly static or at least

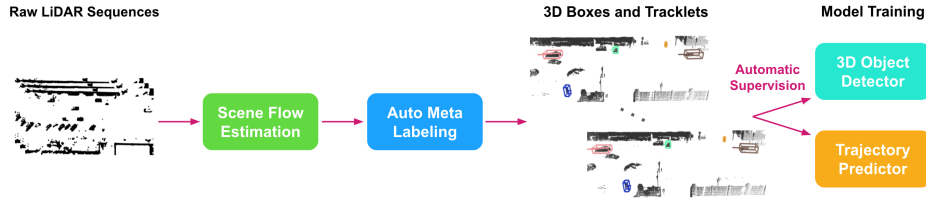


Fig. 1. Proposed framework. Taking as input LiDAR sequences (after ground removal), our approach first reasons about per point motion status (static or dynamic) and predicts accurate scene flow. Based on the motion signal, Auto Meta Labeling clusters points into semantic concepts, connects them across frames and estimates object amodal shapes (3D bounding boxes). The derived amodal boxes and tracklets will serve as automatic supervision to train 3D detection and trajectory prediction models.

non-deformable. In autonomous driving, shape registration has gained increasing attentions where offline processing is required [22,23,56,74,31,88,20]. The shape registration outcome can further support downstream applications such as off-board auto labeling [62,107,99], and perception simulation [52,14]. In this work, we use sequential ICP with motion-inspired initializations to aggregate partial views of objects and produce the auto-labeled bounding boxes.

Trajectory Prediction. The recent introduction of the large-scale trajectory prediction datasets [25,6,11,36], helped deep learning based methods to demonstrate new state-of-the-art performance. From a problem formulation standpoint, these methods can be categorized into uni-modal and multi-modal. Uni-modal approaches [8,19,51,28] predict a single trajectory per agent. Multi-modal methods [10,16,35,2,108,58,104,49,106] take into account the possibility of having multiple plausible trajectories per agent. However, all these methods rely on fully labeled datasets. Unsupervised or open-set settings, although practically important for autonomous driving, have so far remained unexplored. Our method enables existing behavior prediction models to generalize to all moving objects, without the need for predefining an object taxonomy.

3 Method

Fig. 1 illustrates an overview of our proposed method, which primarily relies on motion cues for recognizing moving objects in an unsupervised manner. The pipeline has two main modules: unsupervised scene flow estimation (Sec. 3.1) and Auto Meta Labeling (Sec. 3.2).

3.1 Neural Scene Flow Prior++

Background. Many prior works [41,95,33,47] on scene-flow estimation only considered the supervised scenario where human annotations are available for training. However, these methods cannot generalize well to new environments or to newly seen categories [45]. Recently, Li *et al.* [45] propose neural scene flow prior (NSFP), which can learn point-wise 3D flow vectors by solving an optimization problem at run-time without the need of human annotation. Thanks to its



Fig. 2. Proposed NSFP++. Taking as input raw LiDAR sequences (after ground removal), our approach first reasons about the motion status of each point, decomposes the scene into connected components and predicts local flows accurately for each semantically meaningful component.

unsupervised nature, NSFP can generalize to new environments. It also achieved state-of-the-art performance in 3D scene flow estimation. Still, our study shows that it has notable limitations in handling complex scenes when a mixture of low and high speed objects are present. For example, as illustrated in Fig. 3, NSFP suffers from underestimating the velocity of moving objects, *i.e.*, false negative flows over pedestrians and inaccurate estimation of fast-moving vehicles. It also introduces excessive false positive flows over static objects (e.g., buildings). We hypothesize that such issues are due to the fact that NSFP applies global optimization to the entire point cloud and the highly diverse velocities of different objects set contradictory learning targets for the network to learn properly.

Overview. Our goal is to realize an unsupervised 3D scene flow estimation algorithm that can adapt to various driving scenarios. Here, we present our neural scene flow prior++ (NSFP++) method. As illustrated in Fig. 2, our method features three key innovations: 1) robustly identifying static points; 2) divide-and-conquer strategy to handle different objects by decomposing a scene into semantically meaningful connected components and targetedly estimating local flow for each of the them; 3) flow consistency among points in each component.

Problem Formulation. Let $\mathbf{S}_t \in \mathbb{R}^{N_1 \times 3}$ and $\mathbf{S}_{t+1} \in \mathbb{R}^{N_2 \times 3}$ be two sets of points captured by the LiDAR sensor of an autonomous vehicle at time t and time $t+1$, where N_1 and N_2 denote the number of points in each set. We denote $\mathbf{F}_t \in \mathbb{R}^{N_1 \times 3}$ as the scene flow, a set of flow vectors corresponding to each point in \mathbf{S}_t . Given a point $\mathbf{p} \in \mathbf{S}_t$, we define $\mathbf{f} \in \mathbf{F}_t$ be the corresponding flow vector such that $\hat{\mathbf{p}} = \mathbf{p} + \mathbf{f}$ represents the future position of \mathbf{p} at $t+1$. Typically, points in \mathbf{S}_t and \mathbf{S}_{t+1} have no correspondence and N_1 differs from N_2 .

As in Li *et al.* [45], we model the flow vector $\mathbf{f} = h(\mathbf{p}; \Theta)$ as the output of a neural network h , containing a set of learnable parameters as Θ . To estimate \mathbf{F}_t , we solve for Θ by minimizing the following objective function:

$$\Theta^*, \Theta_{\text{bwd}}^* = \arg \min_{\Theta, \Theta_{\text{bwd}}} \sum_{\mathbf{p} \in \mathbf{S}_t} \mathcal{L}(\mathbf{p} + \mathbf{f}, \mathbf{S}_{t+1}) + \sum_{\hat{\mathbf{p}} \in \hat{\mathbf{S}}_t} \mathcal{L}(\hat{\mathbf{p}} + \mathbf{f}_{\text{bwd}}, \mathbf{S}_t) \quad (1)$$

where $\mathbf{f} = h(\mathbf{p}; \Theta)$ is the forward flow, $\mathbf{f}_{\text{bwd}} = h(\hat{\mathbf{p}}; \Theta_{\text{bwd}})$ is the backward flow, $\hat{\mathbf{S}}_t$ is the set of predicted future positions for points in \mathbf{S}_t and \mathcal{L} is Chamfer distance function. Here we have the forward and backward flow models share the same network architecture but parameterized by Θ and Θ_{bwd} respectively.

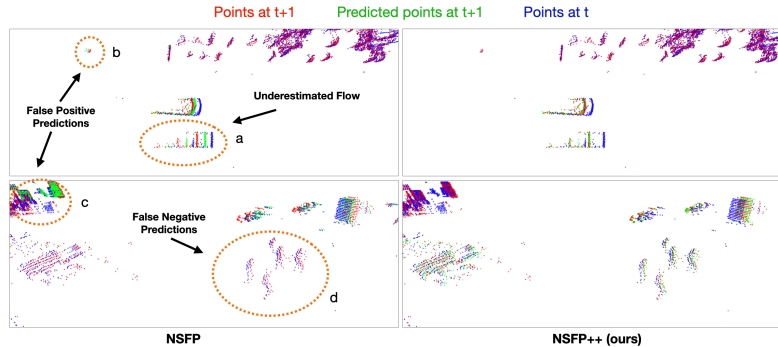


Fig. 3. Flow quality comparison between NSFP [45] and our NSFP++ over the Waymo Open Dataset. Dashed circles in orange color highlight the major shortcomings suffered by NSFP, *i.e.*, (a) underestimated flow for a fast-moving vehicle, (b)(c) false positive predictions at the background and (d) false negative predictions at pedestrians with subtle motion. In contrast, our NSFP++ generates accurate predictions in all these cases.

The model parameters, Θ and Θ_{bwd} are initialized and optimized for each time stamp t . Although we only take the forward flow into the next-step processing, learning the flows bidirectionally help improve the scene flow quality [45,47].

Identifying Static Points. Since our focus is moving objects, we start by strategically removing static points to reduce computational complexity and benefit scene flow estimation. In autonomous driving datasets, one large body of static points is ground. Ground is usually captured as a flat surface for which predicting local motion is not possible due to the aperture problem. We follow [45,47] and remove ground points prior to motion estimation. This is achieved by a RANSAC-based algorithm in which a parameterized close-to-horizontal plane is fitted to the points and points in its vicinity are marked as static. However, ground is not the only static part of the scene and unsupervised flow predictions in these static regions (*e.g.* walls, buildings, trees, *etc.*) introduce noise, reducing the quality of our final auto labels. As a result, we further propose to identify more static regions in the scene prior to scene flow estimation. This is achieved by comparing the Chamfer distance between the points in the current frame with those in earlier frames. We mark points as static if the computed Chamfer distance is less than a threshold. We set a small threshold to have a high precision in this step (*i.e.* 20 cm/s in our experiments).

Estimate Local Flow via Scene Decomposition. Inspired by the fact that objects in outdoor scenes are often well-separated after detecting and isolating the ground points, we propose to further decompose the dynamic part of the scene into connected components. This strategy allows us to solve for local flows for each cluster targetedly, which can greatly improve the accuracy of flow estimation for various traffic participants, *e.g.*, vehicles, pedestrians, cyclists, travelling at highly different velocities. Fig. 2 gives an overview of our method.

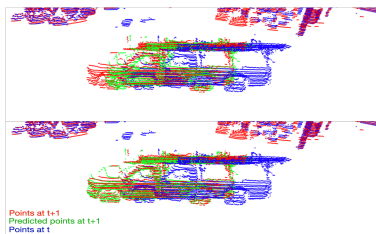


Fig. 4. Illustration of the effectiveness of box query with expansion in more accurately estimating flow over the object shape. Top and bottom are without and with expansion.

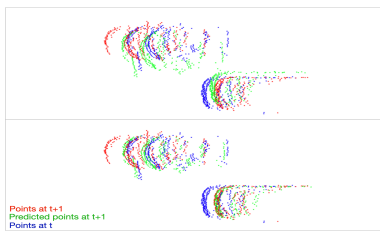


Fig. 5. Illustration of the effectiveness of box query followed by pruning in preserving accurately the local flow for nearby objects. Top and bottom are without and with pruning.

More precisely, given the identified static points, we split the point sets as $\mathbf{S}_t = \mathbf{S}_t^s \cup \mathbf{S}_t^d$ and $\mathbf{S}_{t+1} = \mathbf{S}_{t+1}^s \cup \mathbf{S}_{t+1}^d$, where \mathbf{S}_t^s and \mathbf{S}_{t+1}^s contain static points while \mathbf{S}_t^d and \mathbf{S}_{t+1}^d store dynamic points. This separation, not only helps decompose the scene into semantically meaningful connected components, but also substantially reduces false positive flow predictions on static objects. We then further break down the dynamic points into $\mathbf{S}_t^d = \bigcup_{i=1}^K \mathbf{C}_t^i$, where $\mathbf{C}_t^i \in \mathbb{R}^{m_i \times 3}$ is one disjoint cluster of m_i points (the number of clusters K can vary as the scene changes). In the rest of this section, we omit index i for brevity and let \mathbf{C}_t to represent one of the clusters. For every $\mathbf{C}_t \subseteq \mathbf{S}_t^d$ at time t , we solve for model parameters to derive local flows, by minimizing the objective function as:

$$\begin{aligned} \Theta^*, \Theta_{\text{bwd}}^* = \arg \min_{\Theta, \Theta_{\text{bwd}}} & \sum_{\mathbf{p} \in \mathbf{C}_t \subseteq \mathbf{S}_t^d} \mathcal{L}(\mathbf{p} + \mathbf{f}, \mathbf{C}_{t+1}) + \sum_{\hat{\mathbf{p}} \in \mathbf{C}_t \subseteq \hat{\mathbf{S}}_t^d} \mathcal{L}(\hat{\mathbf{p}} + \mathbf{f}_{\text{bwd}}, \mathbf{C}_t) \\ & + \frac{\alpha}{|\mathbf{C}_t|} \sum_{\substack{\mathbf{f}_i, \mathbf{f}_j \in \mathbf{F}_{\mathbf{C}_t} \\ i \neq j}} \|\mathbf{f}_i - \mathbf{f}_j\|_2^2 \end{aligned} \quad (2)$$

where the last term is the newly introduced local consistency regularizer with α set to 0.1, $\mathbf{F}_{\mathbf{C}_t}$ consists of flow vectors for each point in \mathbf{C}_t , $\hat{\mathbf{S}}_t^d$ contains predicted future positions of all points residing in \mathbf{S}_t^d , $\hat{\mathbf{C}}_t$ is a subset of $\hat{\mathbf{S}}_t^d$ only storing future positions of points in $\mathbf{C}_t \subseteq \mathbf{S}_t^d$ and \mathbf{C}_{t+1} is a subset of \mathbf{S}_{t+1}^d , derived based on box query within a neighborhood of \mathbf{C}_t . Next we will present our box query strategy: expansion with pruning.

Box Query Strategy. Considering that some objects (vehicles) may move at a high speed, we need to expand the field of view to find match points in the next frame. Given a cluster \mathbf{C}_t , we find the axis-aligned (along X and Y axes) bounding box tightly covering \mathbf{C}_t , in the bird’s eye view (BEV). The box is represented as $\mathbf{b} = [x_{\min}, y_{\min}, x_{\max}, y_{\max}]$. Note that fast-moving objects, *e.g.*, vehicles, can travel multiple meters between two LiDAR scans. To satisfactorily capture the points of such objects at time $t + 1$, we propose to expand the box query with axis-aligned buffer distances δ_x , δ_y and use $\mathbf{b}' = [x_{\min} - \delta_x, y_{\min} - \delta_y, x_{\max} + \delta_x, y_{\max} + \delta_y]$ to retrieve points from \mathbf{S}_{t+1}^d , resulting in \mathbf{C}_{t+1} . We set the buffer distances according to the aspect ratio of

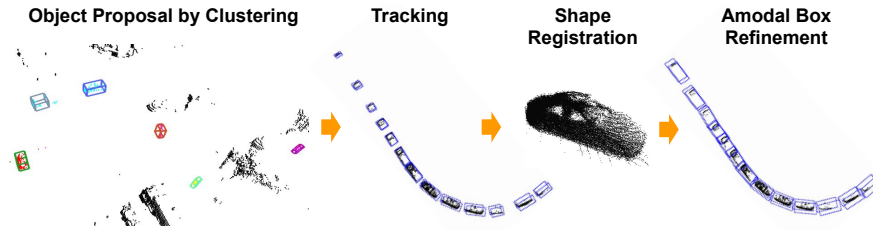


Fig. 6. Auto Meta Labeling pipeline. Given point locations and scene flows on each scene, our Auto Meta Labeling pipeline first proposes objects by spatio-temporal clustering, connects visible bounding boxes of proposals into tracks, then performs shape registration on each track to obtain 3D amodal bounding boxes on each scene.

the box \mathbf{b} , *i.e.*, $\frac{\delta_y}{\delta_x} = \frac{y_{\max} - y_{\min}}{x_{\max} - x_{\min}}$. We empirically set $\max\{\delta_x, \delta_y\} = 2.5\text{m}$. Fig. 4 illustrates that expanding box query captures the full shape of a fast-moving truck, resulting in accurate prediction of the future position of the entire object point cloud (*i.e.*, predicted future positions align nicely with the next frame).

In crowded areas of the scene, retrieved points with \mathbf{b}' may include irrelevant points into the optimization process, causing flow to drift erroneously. See Fig. 5 as an example, where two vehicles are moving fast and close to each other. Box query with \mathbf{b}' can include points from the other vehicle and lead to flow drifting. To address this challenge, we propose to prune retrieved points based on the statistics of \mathbf{C}_t . Formally, let Ω be the set of retrieved points by \mathbf{b}' from \mathbf{S}_{t+1}^d . We select $n = \min\{|\Omega|, |\mathbf{C}_t|\}$ nearest points from Ω with respect to the first moment of \mathbf{C}_t and store them in set $\mathbf{C}_{t+1} \in \mathbb{R}^{n \times 3}$. The effectiveness of pruning in keeping relevant points and thus preserving local flow is shown in Fig. 5.

3.2 Auto Meta Labeling

With the motion signals provided by the unsupervised scene flow module, we are able to generate 3D proposals for moving objects without any manual labels. We propose an Auto Meta Labeling pipeline, which takes point clouds and scene flows as inputs and generates high quality 3D auto labels (Fig. 6). The Auto Meta Labeling pipeline has four components: (a) object proposal by clustering, which leverages spatio-temporal information to cluster points into visible boxes (tight boxes covering visible points), forming the concept of objects in each scene; (b) tracking, which connects visible boxes of objects across frames into tracklets; (c) shape registration, which aggregates points of each track to complete the shape for the object; (d) amodal box refinement, which transforms visible boxes into amodal boxes. See supplementary materials for implementation details.

Object Proposal by Clustering. On each scene, given the point cloud locations $\mathbf{S} = \{\mathbf{p}_n \mid \mathbf{p}_n \in \mathbb{R}^3\}_{n=1}^N$ and the corresponding point-wise scene flows $\mathbf{F} = \{\mathbf{f}_n \mid \mathbf{f}_n \in \mathbb{R}^3\}_{n=1}^N$, the clustering module segments points into subsets where each subset represents an object proposal. We further compute a bounding box of each subset as an object representation. Traditional clustering methods on point cloud often consider 3D point locations \mathbf{S} as the only feature. In the autonomous driving data, with a large portion of points belonging to the background, such

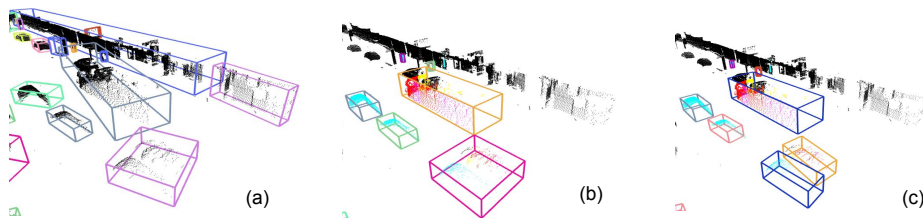


Fig. 7. Comparison between object proposals by different clustering approaches. Points are colored by scene flow magnitudes and directions. Dark for static points. (a) Clustering by location only. (b) Filter by flow magnitude and then cluster based on location (c) Filter by flow and cluster based on both location and motion (ours).

Algorithm 1 Object proposal by spatio-temporal clustering on each scene.

Input: point locations $\mathbf{S} = \{\mathbf{p}_n\}_{n=1}^N$; point-wise scene flows $\mathbf{F} = \{\mathbf{f}_n\}_{n=1}^N$
Hyperparams: neighborhood thresholds ϵ_p, ϵ_f ; minimum flow magnitude $|\mathbf{f}|_{min}$
Output: 3D bounding boxes $\mathbf{B}_{vis} = \{\mathbf{b}_k\}_{k=1}^{M_{pf}}$ of the visible parts of proposed objects

```

function FLOWBASEDCLUSTERING( $\mathbf{S}, \mathbf{F}; \epsilon_p, \epsilon_f, |\mathbf{f}|_{min}$ ):
   $\mathbf{S}', \mathbf{F}' \leftarrow \text{FilterByFlowMagnitude}(\mathbf{S}, \mathbf{F}; |\mathbf{f}|_{min})$ 
   $\mathbf{C}_p \leftarrow \text{DBSCAN}(\mathbf{S}'; \epsilon_p)$        $\triangleright \mathbf{C}_p = \{\mathbf{c}_i\}_{i=1}^{M_p}$ , point sets clustered by locations
   $\mathbf{C}_f \leftarrow \text{DBSCAN}(\mathbf{F}'; \epsilon_f)$      $\triangleright \mathbf{C}_f = \{\mathbf{c}_j\}_{j=1}^{M_f}$ , point sets clustered by flows
  for  $\mathbf{c}_i$  in  $\mathbf{C}_p$  do
    for  $\mathbf{c}_j$  in  $\mathbf{C}_f$  do
       $\mathbf{c}_k \leftarrow \mathbf{c}_i \cap \mathbf{c}_j$ 
       $\bar{\mathbf{f}}_k \leftarrow \text{Average}(\{\mathbf{f}_l \mid \forall l : \mathbf{p}_l \in \mathbf{c}_k\})$ 
       $\mathbf{b}_k \leftarrow \text{MinAreaBBBoxAlongDirection}(\mathbf{c}_k, \bar{\mathbf{f}}_k)$ 
  return  $\{\mathbf{b}_k\}_{k=1}^{M_{pf}}$ 

```

methods generate many irrelevant clusters (Fig. 7a). As we focus on moving objects, we leverage the motion signals to reduce false positives. Hence, a clustering method based on both point locations and scene flows is desired.

One simple yet effective strategy can be filtering point cloud by scene flows before object proposal: we only keep points with a flow magnitude larger than a threshold. We then apply the DBSCAN [24] clustering algorithm on the filtered point sets. This filtering can largely reduce the false positives (Fig. 7b).

However, there is still a common case where the aforementioned approach cannot handle well: close-by objects tend to be under-segmented into a single cluster. To solve this issue, we propose clustering by both spatial locations and scene flows (Algorithm 1). After removing points with flow magnitudes smaller than a threshold $|\mathbf{f}|_{min}$, we obtain the filtered point locations \mathbf{S}' and point-wise scene flows \mathbf{F}' . Then we apply DBSCAN to \mathbf{S}' and \mathbf{F}' separately, resulting in two sets of clusters. Based on its location and motion, a point may fall into different subsets based on these two clusterings. We then intersect the subsets obtained by the location-based and the flow-based clusterings to formulate the final clusters. In this way, two points are clustered together only if they are close with respect to both their location and motion (Fig. 7c).

Algorithm 2 Sequential shape registration and box refinement.

Input: An object track with point locations $\{\mathbf{X}_l\}_{l=1}^L$, bounding boxes $\{\mathbf{b}_l\}_{l=1}^L$, headings $\{\theta_l\}_{l=1}^L$. All in world coordinate system.

Output: Refined boxes $\{\mathbf{b}'_l\}_{l=1}^L$.

```

function SHAPEREGISTRATIONANDBOXREFINEMENT( $\{\mathbf{X}_l\}_{l=1}^L, \{\mathbf{b}_l\}_{l=1}^L, \{\theta_l\}_{l=1}^L$ ):
   $\mathbf{X}'_l = \mathbf{X}_l - \bar{\mathbf{X}}_l, \forall l \in \{1, \dots, L\}$   $\triangleright$  Normalize points to object-centered
   $\mathbf{X}'_{tgt} \leftarrow \mathbf{X}'_{\hat{i}} : \hat{i} = \operatorname{argmax}_i |\mathbf{X}'_i|$   $\triangleright$  Init target as the most dense point cloud
   $I = \{\hat{i} + 1, \hat{i} + 2, \dots, L, \hat{i} - 1, \hat{i} - 2, \dots, 1\}$   $\triangleright$  Shape registration ordering
  for  $i$  in  $I$  do
    for  $\mathbf{T}_j$  in SearchGrid( $\mathbf{b}_t$ ) do
       $\mathcal{T}_{\text{init}} \leftarrow [\mathbf{R}_{\theta_{tgt} - \theta_i} \mid \mathbf{T}_j]$ 
       $\mathbf{X}'_{tgt,j}, \mathcal{T}_{i \rightarrow tgt,j}, \epsilon_j \leftarrow \text{ICP}(\mathbf{X}'_i, \mathbf{X}'_{tgt}, \mathcal{T}_{\text{init}})$ 
       $\mathbf{X}'_{tgt}, \mathcal{T}_{i \rightarrow tgt} \leftarrow \mathbf{X}'_{tgt,j}, \mathcal{T}_{i \rightarrow tgt,j} : \hat{j} = \operatorname{argmin}_j \epsilon_j$   $\triangleright$  Registration w/ least error
     $\mathbf{b}'_{tgt} = \text{MinAreaBBoxAlongDirection}(\mathbf{X}'_{tgt} + \bar{\mathbf{X}}_{tgt}, \theta_{tgt})$ 
  for  $i$  in  $I$  do
     $\mathbf{b}'_i = \text{Transform}(\mathbf{b}'_{tgt}, \mathcal{T}_{i \rightarrow tgt}^{-1})$ 
  return  $\{\mathbf{b}'_l\}_{l=1}^L$ 

```

Having the cluster label for each point, we form the concept of an object via a bounding box covering each cluster. Given the partial observation of objects within a single frame, we only generate boxes tightly covering the visible part in this stage, $\mathbf{B}_{vis} = \{\mathbf{b}_k\}$. Without object semantics, we use motion information to decide the heading of each box. We compute the average flow $\bar{\mathbf{f}}_k$ of each cluster \mathbf{c}_k . Then we find the 7 DoF bounding box \mathbf{b}_k surrounding \mathbf{c}_k which has the minimum area on the xy -plane along the chosen heading direction parallel to $\bar{\mathbf{f}}_k$.

Multi-Object Tracking. The tracking module connects visible boxes \mathbf{B}_{vis} into object tracks. Following the tracking-by-detection paradigm [93,62], we use \mathbf{B}_{vis} for data associations and Kalman filter for state updates. However, rather than relying on the Kalman filter to estimate object speeds, our tracking module leverages our estimated scene flows in the associations. In each step of the association, we advance previously tracked boxes using scene flows and match the advanced boxes with those in the next frame.

Shape Registration and Amodal Box Refinement. In the unsupervised setting, human annotations of object shapes are unavailable. It is hard to infer the amodal shapes of occluded objects purely based on sensor data from one timestamp. However, the observed views of an object often change across time as the autonomous driving car or the object moves. This enables temporal data aggregation to achieve more complete amodal perception of each object.

For temporal aggregation, we propose a shape registration method built upon sequentially applying ICP [4,13,64] (Algorithm 2). ICP performance is sensitive to the transformation initialization. In clustering, we have obtained the headings $\{\theta_l\}_{l=1}^L$ of all visible boxes in each track. The difference in headings of each source and target point set constructs a rotation initialization $R_{\theta_{tgt} - \theta_{src}}$ for ICP.

In autonomous driving scenarios, shape registration among a sequence of observations poses special challenges: (a) objects are moving with large displacements in the world coordinate system; (b) many observations of objects are very sparse due to their far distance from the sensor and/or heavy occlusions. These two challenges make it hard to register points from different frames. To tackle this problem, we search in a grid to obtain the best translation for aligning the source (from frame A) and target (from frame B) point sets. The grid, or the search range, is defined by the size of the target frame bounding box. We initialize the translation \mathbf{T}_j corresponding to different grid points and find the best registration results out of them.

Sequentially, partial views of an object in a track are aggregated into a more complete point set, whose size is often close to amodal boxes. We then compute a bounding box around the target point set similar to the last step in object proposal. During registration, we have estimated the transformation from each source point set to the target, and we can propagate the target bounding box back to each scene by inverting each transformation matrix. Finally, we obtain 3D amodal bounding boxes of detected objects.

4 Experiments

We evaluate our framework using the challenging Waymo Open Dataset (WOD) [75], as it provides a large collection of LiDAR sequences with 3D labels for each frame (we only use labels for evaluation unless noted otherwise). In our experiments, objects with speed $> 1\text{m/s}$ are regarded moving. Hyperparameters and ablation studies are presented in the supplementary material.

4.1 Scene Flow

Metrics. We employ the widely adopted metrics as [45,96], which are 3D end-point error (EPE3D) computed as the mean L2 distance between the prediction and the ground truth for all points; Acc_5 denoting the percentage of points with EPE3D $< 5\text{cm}$ or relative error $< 5\%$; Acc_{10} denoting the percentage of points with EPE3D $< 10\text{cm}$ or relative error $< 10\%$; and θ , the mean angle error between predictions and ground truths. In addition, we evaluate our approach based on fine grained speed breakdowns. We assign each point to one speed class (*e.g.*, 0 - 3m/s, 3 - 6m/s, *etc.*) and employ the Intersection-over-Union (IoU) metric to measure the performance in terms of class-wise IoU and mean IoU. IoU is computed as $\frac{\text{TP}}{\text{TP}+\text{FP}+\text{FN}}$, same as in 3D semantic segmentation [6].

Results. We evaluate our NSFP++ over all frames of the WOD [75] validation set and compare it with the previous state-of-the-art scene flow estimator, NSFP [45]. Following [41], we use the provided vehicle pose to compensate for the ego motion, such that our metrics is independent from the autonomous vehicle motion and can better reflect the flow quality on the moving objects. Fig. 3 visualizes the improvement of the proposed NSFP++ compared to NSFP. Our approach accurately predicts flows for both high- and low-speed objects (a, d). In addition, NSFP++ not only is highly reliable in detecting the subtle motion

Table 1. Comparison of scene flow methods on the WOD validation set.

Method	EPE3D (m) ↓	Acc ₅ (%) ↑	Acc ₁₀ (%) ↑	θ (rad) ↓
NSFP [45]	0.455	23.65	43.06	0.9190
NSFP++ (ours)	0.017	95.05	96.45	0.4737

Table 2. Comparison of scene flow methods on the WOD validation set, with speed breakdowns.

Method	IoU per Speed Breakdown (m/s)						mIOU
	0 - 3	3 - 6	6 - 9	9 - 12	12 - 15	15+	
NSFP [45]	0.657	0.152	0.216	0.166	0.130	0.140	0.244
NSFP++ (ours)	0.989	0.474	0.522	0.479	0.442	0.608	0.586

of vulnerable road users (d) but can also robustly distinguish all moving objects from the static background (b, c). Finally, our approach outperforms NSFP substantially across all quantitative metrics, as listed in Tab. 1 and Tab. 2.

4.2 Unsupervised 3D Object Detection

Our method aims at generating auto labels for training downstream autonomous driving tasks in a fully unsupervised manner. 3D object detection is a core component in autonomous driving systems. In this section, we evaluate the effectiveness of our unsupervised AML pseudo labels by training a 3D object detector. We adopt the PointPillars [43] detector for our experiments. All models are trained and evaluated on WOD [75] training and validation sets. Since there is no category information during training, we use a single-class detector to detect any moving objects. We train and evaluate the detectors on a 100m x 40m rectangular region around the ego vehicle to reflect the egocentric importance of the predictions [17]. We set a 3D IoU of 0.4 during evaluation to count for the large variation in size of the class-agnostic moving objects, e.g., vehicles, pedestrians, cyclists. We employ a top-performing flow model [41] as the supervised counterpart to our unsupervised flow model NSFP++.

Tab. 3 compares performance of detectors trained with auto labels generated by our pipelines and several baselines. The first two rows show detection results when a fully supervised flow model [41] (flow supervision derived from human box labels) is deployed for generating the auto labels. The first row represents a baseline where our hybrid clustering method is used to form the auto labels based on motion cues [18]. The second row shows the performance when the same supervised flow predictions are used in combination with our AML pipeline. Clearly, our AML pipeline greatly outperforms the clustering baseline, verifying the high-quality auto labels generated by our method. The last four rows consider the unsupervised setting. *No flow + Clustering* is a baseline where DBSCAN is applied to the point locations to form the auto labels. *No flow + AML* is our pipeline when purely relying on a regular tracker without using any flow information. *Unsup Flow + Clustering* uses our proposed hybrid clustering technique on the outputs of our NSFP++ scene flow estimator without connecting with our AML. *Unsup Flow + AML* is our full unsupervised pipeline. Notably, not only does it outperforms other unsupervised baselines by a large margin,

Table 3. Comparisons between 3D detectors trained with autolabels generated by AML with supervised flow and unsupervised flow.

Method	Supervision	3D mAP		2D mAP	
		L1	L2	L1	L2
Sup Flow [41] + Clustering	Supervised	30.8	29.7	42.7	41.2
Sup Flow [41] + AML		49.9	48.0	56.8	54.8
No flow + Clustering	Unsupervised	4.7	4.5	5.8	5.6
No flow + AML		9.6	9.4	11.0	10.8
Unsup Flow + Clustering		30.4	29.2	36.7	35.3
Unsup Flow + AML		42.1	40.4	49.1	47.4

but it also achieves a comparable performance with the supervised *Sup Flow + AML* counterpart. Moreover, comparing it with other unsupervised baselines by removing parts of our pipeline validates the importance of all components in our design (please see the supplementary for more ablations). Most importantly, our approach is a fully unsupervised 3D pipeline, capable of detecting moving objects in the open-set environment. This new feature is cost efficient and safety critical for the autonomous vehicle to reliably detect arbitrary moving objects, removing the need of human annotation and the constraint of predefined taxonomy.

4.3 Open-set 3D Object Detection

In this section, we turn our attention to the open-set setting where only a subset of categories are annotated. Since there is no public 3D dataset designed for this purpose, we perform experiments in a leave-one-out manner on WOD [75]. WOD has three categories, namely vehicle, pedestrian, and cyclist. Considering the similar appearances and safety requirements, we combine pedestrian and cyclist into a larger category called VRU (vulnerable road user), resulting in a data size comparable with the vehicle category. We then assume to only have access to human annotations for one of the two categories, leaving the other one out for our auto meta label pipeline to pseudo label.

Tab. 4 shows the results. The first two rows show the performance of a fully supervised point pillars detector. As expected, when the detector is trained on one of the categories, it can not generalize to the other. In the last two rows, when human annotations are not available, we rely on our auto labels to fill in for the unknown category. When no vehicle label is available, our pipeline helps the detector to generalize and consequently improves the mAP from 48.8 to 77.1. Although generalizing to VRUs without any human labels is a more challenging scenario, our pipeline still improves the mAP by a noticeable margin, showing its effectiveness in the open-set settings.

Table 4. Open-set 3D detection results.

	Human Labeled		Vehicle 3D AP	VRU 3D AP	3D mAP
	Vehicle	VRU			
Supervised Method	✓		97.5	0.0	48.8
		✓	0.0	88.7	44.4
Ours (Supervised + AML)	✓		97.5	20.8	59.2
		✓	65.4	88.7	77.1

4.4 Open-set Trajectory Prediction

For trajectory prediction, we have extracted road graph information for a subset of WOD (consisting of 625 training and 172 validation sequences). We use those WOD run segments with road graph information for our trajectory prediction experiments. Following [25], a trajectory prediction model is required to forecast the future positions for surrounding agents for 8 seconds into the future, based on the observation of 1 second history. We use the MultiPath++ [83] model for our study. The model predicts 6 different trajectories for each object and a probability for each trajectory. To evaluate the impact of open-set moving objects on the behavior prediction task, we train models using perception labels derived via different strategies as the ground truth data and then evaluate the behavior prediction metrics of the trained models on a manually labeled validation set. We use the minADE and minFDE metrics as described in [25].

Tab. 5 reports the trajectory prediction results. While the supervised method achieves a reasonable result when the vehicle class is labeled, its performance is poor when trained only on the VRU class. This is expected, as the motion learned from slow vehicles can be generalized to VRUs to some extent, but predicting the trajectory of the fast moving vehicles is out of reach for a model trained on only VRUs. The last two rows show the performance of the same model when AML is deployed for auto-labeling the missing category. Consistent with our observation in 3D detection, our method can bridge the gap in the open-set setting. Namely, our approach significantly remedies the generalization problem from VRUs to vehicles and achieves the best performance when combining human labels of the vehicle class with our auto labels for VRUs.

Table 5. Open-set trajectory prediction results.

	Human Labeled		minADE	minFDE
	Vehicle	VRU		
Supervised Method	✓		2.12	5.39
		✓	9.53	22.31
Ours (Supervised + AML)	✓		1.89	4.79
		✓	2.15	5.55

5 Conclusion

In this paper, we proposed a novel unsupervised framework for training onboard 3D detection and prediction models to understand open-set moving objects. Extensive experiments show that our unsupervised approach is competitive in regular detection tasks to the counterpart which uses supervised scene flow. With promising results, it demonstrates great potential in enabling perception and prediction systems to handle open-set moving objects. We hope our findings encourage more research toward solving autonomy in an open-set environment.

References

1. Agarwal, S., Snavely, N., Seitz, S.M., Szeliski, R.: Bundle adjustment in the large. In: ECCV (2010) [3](#)
2. Bansal, M., Krizhevsky, A., Ogale, A.: Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. arXiv preprint arXiv:1812.03079 (2018) [4](#)
3. Bau, D., Zhu, J.Y., Strobel, H., Zhou, B., Tenenbaum, J.B., Freeman, W.T., Torralba, A.: Gan dissection: Visualizing and understanding generative adversarial networks. In: ICLR (2019) [3](#)
4. Besl, P.J., McKay, N.D.: Method for registration of 3-d shapes. In: Sensor fusion IV: control paradigms and data structures. vol. 1611, pp. 586–606. Spie (1992) [3](#), [10](#)
5. Bewley, A., Sun, P., Mensink, T., Anguelov, D., Sminchisescu, C.: Range conditioned dilated convolutions for scale invariant 3d object detection (2020) [3](#)
6. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nusenes: A multimodal dataset for autonomous driving. In: CVPR (2020) [4](#), [11](#)
7. Caine, B., Roelofs, R., Vasudevan, V., Ngiam, J., Chai, Y., Chen, Z., Shlens, J.: Pseudo-labeling for scalable 3d object detection. arXiv preprint arXiv:2103.02093 (2021) [3](#)
8. Casas, S., Luo, W., Urtasun, R.: Intentnet: Learning to predict intention from raw sensor data. In: CoRL (2018) [4](#)
9. Cen, J., Yun, P., Cai, J., Wang, M.Y., Liu, M.: Open-set 3d object detection. In: 3DV (2021) [3](#)
10. Chai, Y., Sapp, B., Bansal, M., Anguelov, D.: Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In: CoRL (2019) [1](#), [4](#)
11. Chang, M.F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., et al.: Argoverse: 3d tracking and forecasting with rich maps. In: CVPR (2019) [4](#)
12. Chen, Y., Liu, S., Shen, X., Jia, J.: Fast point r-cnn. In: ICCV (2019) [3](#)
13. Chen, Y., Medioni, G.: Object modelling by registration of multiple range images. *Image and vision computing* **10**(3), 145–155 (1992) [3](#), [10](#)
14. Chen, Y., Rong, F., Duggal, S., Wang, S., Yan, X., Manivasagam, S., Xue, S., Yumer, E., Urtasun, R.: Geosim: Realistic video simulation via geometry-aware composition for self-driving. In: CVPR (2021) [4](#)
15. Cho, M., Kwak, S., Schmid, C., Ponce, J.: Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In: CVPR (2015) [3](#)
16. Cui, H., Radosavljevic, V., Chou, F.C., Lin, T.H., Nguyen, T., Huang, T.K., Schneider, J., Djuric, N.: Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In: ICRA (2019) [4](#)
17. Deng, B., Qi, C.R., Najibi, M., Funkhouser, T., Zhou, Y., Anguelov, D.: Revisiting 3d object detection from an egocentric perspective. NeurIPS (2021) [12](#)
18. Dewan, A., Caselitz, T., Tipaldi, G.D., Burgard, W.: Motion-based detection and tracking in 3d lidar scans. In: ICRA (2016) [3](#), [12](#)
19. Djuric, N., Radosavljevic, V., Cui, H., Nguyen, T., Chou, F.C., Lin, T.H., Schneider, J.: Short-term motion prediction of traffic actors for autonomous driving using deep convolutional networks (2018) [4](#)

20. Duggal, S., Wang, Z., Ma, W.C., Manivasagam, S., Liang, J., Wang, S., Urtasun, R.: Mending neural implicit modeling for 3d vehicle reconstruction in the wild. In: WACV (2022) [4](#)
21. Engelcke, M., Rao, D., Wang, D.Z., Tong, C.H., Posner, I.: Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In: ICRA (2017) [2](#)
22. Engelmann, F., Stückler, J., Leibe, B.: Joint object pose estimation and shape reconstruction in urban street scenes using 3d shape priors. In: German Conference on Pattern Recognition. pp. 219–230. Springer (2016) [4](#)
23. Engelmann, F., Stückler, J., Leibe, B.: Samp: shape and motion priors for 4d vehicle reconstruction. In: WACV (2017) [4](#)
24. Ester, M., Kriegel, H.P., Sander, J., Xu, X., et al.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD (1996) [9](#)
25. Ettinger, S., Cheng, S., Caine, B., Liu, C., Zhao, H., Pradhan, S., Chai, Y., Sapp, B., Qi, C.R., Zhou, Y., et al.: Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In: ICCV (2021) [4](#), [14](#)
26. Faktor, A., Irani, M.: “clustering by composition”—unsupervised discovery of image categories. In: ECCV (2012) [3](#)
27. Fan, L., Xiong, X., Wang, F., Wang, N., Zhang, Z.: Rangedet: In defense of range view for lidar-based 3d object detection. In: ICCV (2021) [3](#)
28. Gao, J., Sun, C., Zhao, H., Shen, Y., Anguelov, D., Li, C., Schmid, C.: Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In: CVPR (2020) [4](#)
29. Grauman, K., Darrell, T.: Unsupervised learning of categories from sets of partially matching image features. In: CVPR (2006) [3](#)
30. Groß, J., Ošep, A., Leibe, B.: Alignnet-3d: Fast point cloud registration of partially observed objects. In: 3DV (2019) [3](#), [21](#), [23](#)
31. Gu, J., Ma, W.C., Manivasagam, S., Zeng, W., Wang, Z., Xiong, Y., Su, H., Urtasun, R.: Weakly-supervised 3d shape completion in the wild. In: ECCV (2020) [4](#)
32. Gu, J., Sun, C., Zhao, H.: Densetnt: End-to-end trajectory prediction from dense goal sets. In: ICCV (2021) [1](#)
33. Gu, X., Wang, Y., Wu, C., Lee, Y.J., Wang, P.: Hplflownet: Hierarchical permutohedral lattice flownet for scene flow estimation on large-scale point clouds. In: CVPR (2019) [3](#), [4](#)
34. He, C., Zeng, H., Huang, J., Hua, X.S., Zhang, L.: Structure aware single-stage 3d object detection from point cloud. In: CVPR (June 2020) [3](#)
35. Hong, J., Sapp, B., Philbin, J.: Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions. In: CVPR (2019) [4](#)
36. Houston, J., Zuidhof, G., Bergamini, L., Ye, Y., Chen, L., Jain, A., Omari, S., Igloukov, V., Ondruska, P.: One thousand and one hours: Self-driving motion prediction dataset. arXiv preprint arXiv:2006.14480 (2020) [4](#)
37. Insafutdinov, E., Dosovitskiy, A.: Unsupervised learning of shape and pose with differentiable point clouds. In: NeurIPS (2018) [3](#)
38. Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., et al.: Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In: Proceedings of the 24th annual ACM symposium on User interface software and technology. pp. 559–568 (2011) [3](#)
39. Jerripothula, K.R., Cai, J., Yuan, J.: Cats: Co-saliency activated tracklet selection for video co-localization. In: ECCV (2016) [3](#)

40. Joulin, A., Bach, F., Ponce, J.: Discriminative clustering for image co-segmentation. In: CVPR (2010) [3](#)
41. Jund, P., Sweeney, C., Abdo, N., Chen, Z., Shlens, J.: Scalable scene flow from point clouds in the real world. *IEEE Robotics and Automation Letters* **7**(2), 1589–1596 (2022). <https://doi.org/10.1109/LRA.2021.3139542> [4](#), [11](#), [12](#), [13](#), [21](#)
42. Kim, G., Torralba, A.: Unsupervised detection of regions of interest using iterative link analysis. In: NIPS (2009) [3](#)
43. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: CVPR (2019) [2](#), [12](#)
44. Lee, H., Grosse, R., Ranganath, R., Ng, A.Y.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: ICML (2009) [3](#)
45. Li, X., Pontes, J.K., Lucey, S.: Neural scene flow prior. In: NeurIPS (2021) [2](#), [3](#), [4](#), [5](#), [6](#), [11](#), [12](#)
46. Li, Z., Wang, F., Wang, N.: Lidar r-cnn: An efficient and universal 3d object detector. In: CVPR (2021) [2](#)
47. Liu, X., Qi, C.R., Guibas, L.J.: Flownet3d: Learning scene flow in 3d point clouds. In: CVPR (2019) [3](#), [4](#), [6](#)
48. Liu, Y., Zufikar, I.E., Luiten, J., Dave, A., Ošep, A., Ramanan, D., Leibe, B., Leal-Taixé, L.: Opening up open-world tracking. In: CVPR (2022) [3](#)
49. Liu, Y., Zhang, J., Fang, L., Jiang, Q., Zhou, B.: Multimodal motion prediction with stacked transformers. In: CVPR (2021) [4](#)
50. Luo, C., Yang, X., Yuille, A.: Self-supervised pillar motion learning for autonomous driving. In: CVPR (2021) [2](#)
51. 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. In: CVPR (2018) [1](#), [4](#)
52. 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: CVPR (2020) [4](#)
53. Meyer, G.P., Laddha, A., Kee, E., Vallespi-Gonzalez, C., Wellington, C.K.: Laser-net: An efficient probabilistic 3d object detector for autonomous driving. In: CVPR (2019) [3](#)
54. Misra, I., Girdhar, R., Joulin, A.: An end-to-end transformer model for 3d object detection. In: ICCV (2021) [2](#)
55. Mittal, H., Okorn, B., Held, D.: Just go with the flow: Self-supervised scene flow estimation. In: CVPR (2020) [2](#), [3](#)
56. Najibi, M., Lai, G., Kundu, A., Lu, Z., Rathod, V., Funkhouser, T., Pantofaru, C., Ross, D., Davis, L.S., Fathi, A.: Dops: Learning to detect 3d objects and predict their 3d shapes. In: CVPR (2020) [2](#), [4](#)
57. Pang, Z., Li, Z., Wang, N.: Model-free vehicle tracking and state estimation in point cloud sequences. In: IROS (2021) [3](#)
58. Phan-Minh, T., Grigore, E.C., Boulton, F.A., Beijbom, O., Wolff, E.M.: Covernet: Multimodal behavior prediction using trajectory sets. In: CVPR (2020) [4](#)
59. Pontes, J.K., Hays, J., Lucey, S.: Scene flow from point clouds with or without learning. In: 2020 International Conference on 3D Vision (3DV). pp. 261–270 (2020). <https://doi.org/10.1109/3DV50981.2020.00036> [2](#)
60. Puy, G., Boulch, A., Marlet, R.: Flot: Scene flow on point clouds guided by optimal transport. In: ECCV (2020) [3](#)
61. Qi, C.R., Litany, O., He, K., Guibas, L.J.: Deep hough voting for 3d object detection in point clouds. In: ICCV (2019) [1](#), [2](#)

62. Qi, C.R., Zhou, Y., Najibi, M., Sun, P., Vo, K., Deng, B., Anguelov, D.: Offboard 3d object detection from point cloud sequences. In: CVPR (2021) [3](#), [4](#), [10](#), [21](#)
63. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. In: ICLR (2015) [3](#)
64. Rusinkiewicz, S., Levoy, M.: Efficient variants of the icp algorithm. In: Proceedings third international conference on 3-D digital imaging and modeling. pp. 145–152. IEEE (2001) [3](#), [10](#)
65. Russell, B.C., Freeman, W.T., Efros, A.A., Sivic, J., Zisserman, A.: Using multiple segmentations to discover objects and their extent in image collections. In: CVPR (2006) [3](#)
66. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: CVPR (2016) [3](#)
67. Shi, S., Guo, C., Jiang, L., Wang, Z., Shi, J., Wang, X., Li, H.: Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In: CVPR (2020) [3](#)
68. Shi, S., Wang, X., Li, H.: Pointcnn: 3d object proposal generation and detection from point cloud. In: CVPR (2019) [1](#), [2](#)
69. Shi, W., Rajkumar, R.R.: Point-gnn: Graph neural network for 3d object detection in a point cloud. In: CVPR (2020) [2](#)
70. Simony, M., Milzy, S., Amendey, K., Gross, H.M.: Complex-yolo: an euler-region-proposal for real-time 3d object detection on point clouds. In: ECCV (2018) [2](#)
71. Sivic, J., Russell, B.C., Efros, A.A., Zisserman, A., Freeman, W.T.: Discovering object categories in image collections. In: ICCV (2005) [3](#)
72. Sohn, K., Zhou, G., Lee, C., Lee, H.: Learning and selecting features jointly with point-wise gated boltzmann machines. In: ICML (2013) [3](#)
73. Song, S., Xiao, J.: Deep sliding shapes for amodal 3d object detection in rgb-d images. In: CVPR (2016) [2](#)
74. Stutz, D., Geiger, A.: Learning 3d shape completion from laser scan data with weak supervision. In: CVPR (2018) [4](#)
75. Sun, P., Kretschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., Vasudevan, V., Han, W., Ngiam, J., Zhao, H., Timofeev, A., Ettinger, S., Krivokon, M., Gao, A., Joshi, A., Zhang, Y., Shlens, J., Chen, Z., Anguelov, D.: Scalability in perception for autonomous driving: Waymo open dataset. In: CVPR (2020) [2](#), [11](#), [12](#), [13](#), [21](#)
76. Sun, P., Wang, W., Chai, Y., Elsayed, G., Bewley, A., Zhang, X., Sminchisescu, C., Anguelov, D.: Rsn: Range sparse net for efficient, accurate lidar 3d object detection. In: CVPR. pp. 5725–5734 (2021) [3](#)
77. Tang, C., Tan, P.: Ba-net: Dense bundle adjustment network. In: ICLR (2019) [3](#)
78. Tian, H., Chen, Y., Dai, J., Zhang, Z., Zhu, X.: Unsupervised object detection with lidar clues. In: CVPR (2021) [3](#)
79. Triggs, B., McLauchlan, P.F., Hartley, R.I., Fitzgibbon, A.W.: Bundle adjustment—a modern synthesis. In: International workshop on vision algorithms. pp. 298–372. Springer (1999) [3](#)
80. Tulsiani, S., Efros, A.A., Malik, J.: Multi-view consistency as supervisory signal for learning shape and pose prediction. In: CVPR (2018) [3](#)
81. Tulsiani, S., Zhou, T., Efros, A.A., Malik, J.: Multi-view supervision for single-view reconstruction via differentiable ray consistency. In: CVPR (2017) [3](#)
82. Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., Brox, T.: Demon: Depth and motion network for learning monocular stereo. In: CVPR (2017) [3](#)

83. Varadarajan, B., Hefny, A., Srivastava, A., Refaat, K.S., Nayakanti, N., Cornman, A., Chen, K., Douillard, B., Lam, C., Anguelov, D., Sapp, B.: Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. CoRR **abs/2111.14973** (2021), <https://arxiv.org/abs/2111.14973> 14
84. Vo, H.V., Bach, F., Cho, M., Han, K., LeCun, Y., Pérez, P., Ponce, J.: Unsupervised image matching and object discovery as optimization. In: CVPR (2019) 3
85. Vo, H.V., Pérez, P., Ponce, J.: Toward unsupervised, multi-object discovery in large-scale image collections. In: ECCV (2020) 3
86. Vo, V.H., Sizikova, E., Schmid, C., Pérez, P., Ponce, J.: Large-scale unsupervised object discovery. In: NeurIPS (2021) 3
87. Wang, D.Z., Posner, I.: Voting for voting in online point cloud object detection. In: Proceedings of Robotics: Science and Systems. Rome, Italy (July 2015) 2
88. Wang, R., Yang, N., Stückler, J., Cremers, D.: Directshape: Direct photometric alignment of shape priors for visual vehicle pose and shape estimation. In: ICRA (2020) 4
89. Wang, Y., Fathi, A., Kundu, A., Ross, D., Pantofaru, C., Funkhouser, T., Solomon, J.: Pillar-based object detection for autonomous driving. In: ECCV (2020) 2
90. Wang, Y., Solomon, J.M.: Deep closest point: Learning representations for point cloud registration. In: ICCV (2019) 3
91. Wang, Z., Li, S., Howard-Jenkins, H., Prisacariu, V., Chen, M.: Flownet3d++: Geometric losses for deep scene flow estimation. In: WACV (2020) 3
92. Wei, X., Zhang, Y., Li, Z., Fu, Y., Xue, X.: Deepsfm: Structure from motion via deep bundle adjustment. In: ECCV (2020) 3
93. Weng, X., Kitani, K.: A baseline for 3d multi-object tracking. arXiv preprint arXiv:1907.03961 1(2), 6 (2019) 10
94. Wong, K., Wang, S., Ren, M., Liang, M., Urtasun, R.: Identifying unknown instances for autonomous driving. In: CoRL. PMLR (2020) 3
95. Wu, P., Chen, S., Metaxas, D.N.: Motionnet: Joint perception and motion prediction for autonomous driving based on bird’s eye view maps. In: CVPR (2020) 4
96. Wu, W., Wang, Z.Y., Li, Z., Liu, W., Fuxin, L.: Pointpwc-net: Cost volume on point clouds for (self-) supervised scene flow estimation. In: ECCV (2020) 2, 11
97. Yan, X., Hsu, J., Khansari, M., Bai, Y., Pathak, A., Gupta, A., Davidson, J., Lee, H.: Learning 6-dof grasping interaction via deep geometry-aware 3d representations. In: ICRA (2018) 3
98. Yan, X., Yang, J., Yumer, E., Guo, Y., Lee, H.: Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In: NIPS (2016) 3
99. Yang, B., Bai, M., Liang, M., Zeng, W., Urtasun, R.: Auto4d: Learning to label 4d objects from sequential point clouds. arXiv preprint arXiv:2101.06586 (2021) 3, 4
100. Yang, B., Luo, W., Urtasun, R.: Pixor: Real-time 3d object detection from point clouds. In: CVPR (2018) 2
101. Yang, H., Shi, J., Carlone, L.: Teaser: Fast and certifiable point cloud registration. IEEE Transactions on Robotics 37(2), 314–333 (2020) 3
102. Yang, Z., Sun, Y., Liu, S., Jia, J.: 3dssd: Point-based 3d single stage object detector. In: CVPR (2020) 1, 2
103. Ye, M., Xu, S., Cao, T.: Hynet: Hybrid voxel network for lidar based 3d object detection. In: CVPR (2020) 2

104. Ye, M., Cao, T., Chen, Q.: Tpcn: Temporal point cloud networks for motion forecasting. In: CVPR (2021) [1](#), [4](#)
105. Yuan, J., Liu, Z., Wu, Y.: Discriminative subvolume search for efficient action detection. In: CVPR (2009) [3](#)
106. Yuan, Y., Weng, X., Ou, Y., Kitani, K.M.: Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In: ICCV (2021) [4](#)
107. Zakharov, S., Kehl, W., Bhargava, A., Gaidon, A.: Autolabeling 3d objects with differentiable rendering of sdf shape priors. In: CVPR (2020) [4](#)
108. Zeng, W., Luo, W., Suo, S., Sadat, A., Yang, B., Casas, S., Urtasun, R.: End-to-end interpretable neural motion planner. In: CVPR (2019) [4](#)
109. Zheng, W., Tang, W., Jiang, L., Fu, C.W.: Se-ssd: Self-ensembling single-stage object detector from point cloud. In: CVPR (2021) [2](#)
110. Zhou, T., Brown, M., Snavely, N., Lowe, D.G.: Unsupervised learning of depth and ego-motion from video. In: CVPR (2017) [3](#)
111. Zhou, Y., Sun, P., Zhang, Y., Anguelov, D., Gao, J., Ouyang, T., Guo, J., Ngiam, J., Vasudevan, V.: End-to-end multi-view fusion for 3d object detection in lidar point clouds. In: CoRL (2020) [3](#)
112. Zhou, Y., Tuzel, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: CVPR (2018) [1](#), [2](#)
113. Zhu, R., Kiani Galoogahi, H., Wang, C., Lucey, S.: Rethinking reprojection: Closing the loop for pose-aware shape reconstruction from a single image. In: ICCV (2017) [3](#)

Appendix

A Implementation Details of Auto Meta Labeling

The Auto Meta Labeling pipeline has four components: object proposal by clustering, multi-object tracking and amodal box refinement based on shape registration. In the object proposal step, we use DBSCAN for both clustering by point locations and by scene flows. Both clustering methods use Euclidean distance as the distance metric. The neighborhood thresholds ϵ_p and ϵ_f are set to be 1.0 and 0.1, respectively. The minimum flow magnitude $|\mathbf{f}|_{min}$ is set to 1m/s, so as to include meaningful motions without introducing too much background noise. Our tracker follows the implementation as in [62]. We use bird’s eye view (BEV) boxes for data association and use Hungarian matching with an IoU threshold of 0.1. In shape registration, we use a constrained ICP [30] which limits the rotation to be only round z -axis. We have compared the effect of constrained and unconstrained ICP in AML ablation study. The search grid for translation initialization is decided by the target box dimensions on the xy -plane, *i.e.* the length $l_{\mathbf{b}_{tgt}}$ and the width $w_{\mathbf{b}_{tgt}}$ of the target bounding box. We enumerate translation initialization \mathbf{T}_j in a 5×5 grid covering the target bounding box region with a list \mathcal{T}_x of strides as $[-l_{\mathbf{b}_{tgt}}/2, -l_{\mathbf{b}_{tgt}}/4, 0, l_{\mathbf{b}_{tgt}}/4, l_{\mathbf{b}_{tgt}}/2]$ and a list \mathcal{T}_y of strides $[-w_{\mathbf{b}_{tgt}}/2, -w_{\mathbf{b}_{tgt}}/4, 0, w_{\mathbf{b}_{tgt}}/4, w_{\mathbf{b}_{tgt}}/2]$. Each computation of ICP outputs an error ϵ_j , which is defined as the mean of the Euclidean distances among matched points between the source and the target point sets.

B Ablation Study on Unsupervised Flow Estimation

In this section we provide additional ablation studies focusing on our unsupervised flow estimation method, NSFP++.

Static point removal As mentioned in Sec. 3.1, we apply static point removal prior to unsupervised flow estimation. This step is designed to achieve a high precision to avoid removing dynamic points in the early stages of our pipeline. Here, we compute the precision/recall of this step on the WOD [75] validation set. We define ground-truth dynamic/static labels based on the available ground-truth bounding boxes [41]. Dynamic points are defined as those with a ground-truth flow magnitude larger than $|\mathbf{f}|_{min}$, and the remaining points belonging to any ground-truth box are assigned to the static class. Our static point removal step has a precision of 97.2%, and a recall of 62.2%, validating the high precision of this step in determining the static points.

Local flow estimation We also conduct ablation study to validate the effectiveness of the proposed components in the local flow estimation step, *i.e.*, box query with expansion followed by pruning and local consistency loss. As illustrated in Table 6, box query with expansion (second row) effectively boosts mIoU

Table 6. Ablation study on different components in the proposed local flow estimation. BQ stands for the proposed box query strategy, which contains two steps, the first being expansion and the second being pruning. Local consistency represents the local consistency loss among flow predictions within each point cluster.

Variants of NSFP++			EPE3D ↓	θ (rad) ↓	mIoU ↑
BQ w. Expansion	BQ w. Pruning	Local Consistency			
			0.020	0.515	0.404
✓			0.023	0.560	0.552
✓	✓		0.018	0.504	0.571
✓	✓	✓	0.017	0.474	0.586

Table 7. Flow comparison with the fully supervised model.

Method	EPE3D (m) ↓	θ (rad) ↓	mIoU ↑
Fully Supervised Network	0.005	0.062	0.826
Unsupervised NSFP++ (ours)	0.017	0.474	0.586

from 0.404 to 0.552 but suffers from higher 3D end-point error (EPE3D) and mean angle error (θ), compared to the method without using box query (first row). This is due to the fact that the expanded query region can capture more matching points but at the cost of including irrelevant points. With the proposed pruning scheme (third row), all metrics are significantly improved compared to the previous two rows. Finally, by adding local consistency loss (fourth row), we obtain the best performance across the board.

Comparison with the fully supervised model In this subsection, we compare our unsupervised flow estimation method with the fully supervised scene flow model used in Sec. 4. Table 7 shows the comparison. As expected, the supervised model outperforms our unsupervised NSFP++ method which does not use any human annotations. However, as shown in Tab. 3, the AML pipeline can robustly use our unsupervised NSFP++ predictions and eventually achieves comparable results to the counterpart using a supervised flow model on downstream tasks (e.g., L1 mAP of 42.1 for unsupervised *v.s.* 49.9 for supervised in the object detection task).

C Ablation Study on Auto Meta Labeling

To examine the design choices in the AML pipeline, we compute the detection metrics on the auto labels generated by our full AML pipeline and several baselines (Table 8). Note that the numbers reported in Table 8 are from evaluation on auto labels, rather than on the predictions by trained detectors. *Filtered by flow + Clustering by position* is a baseline where we generate auto labels only using this clustering method. Compared to our spatial-temporal clustering

Table 8. Comparisons of different variants of components in the AML pipeline. All methods are evaluated on the WOD validation set.

AML Variants	3D mAP		2D mAP	
	L1	L2	L1	L2
Filtered by flow + Clustering by position	25.5	24.6	32.4	31.2
Spatio-temporal clustering	30.4	29.2	36.7	35.3
Regis. w/o init.	32.2	31.0	36.6	35.3
Regis. w/ R init. by flow heading	33.2	31.9	37.4	36.0
Regis. w/ T init. by grid search	35.2	33.8	39.3	37.9
Regis. w/ Unconstrained ICP	34.3	33.0	38.5	37.1
Regis. w/ RT init. & constrained ICP [30] (Full AML)	36.9	35.5	40.5	39.0

method described in Algorithm 1, this baseline does not perform clustering on the estimated flows and as a result it leads to under-segmentation and lower performance.

We also carry out experiments on variants of shape registration. *Regis. w/o init.* is a baseline where we have no initialization when performing constrained ICP. Adding either rotation initialization by flow heading (*Regis. w/ R init. by flow heading*) or translation initialization by grid search (*Regis. w/ T init. by grid search*) improves the quality of auto labels. Another baseline, *Regis. w/ Unconstrained ICP*, is applying both **R** and **T** initializations but uses an unconstrained ICP such that 3D rotations are allowed when aligning the source and the target point sets. We find that limiting the rotation to be only around z -axis generates auto labels with a higher quality. Finally, our full AML (*Regis. w/ RT init. & constrained ICP*) outperforms all other variants. Compared to the 3D detection results in the main paper (see Tab. 3), we find that the object detector achieves higher mAP than the auto labels it is trained on. The reason is that auto labels by design pursue high recall while contain some false positives in the background due to inaccurate flow or noise in the environment. As these false positive labels do not form a consistent data pattern, the object detector learns to focus only on auto labels with common patterns, such as vehicles and VRUs, and assign high confidence scores to these objects at inference time.

D Qualitative Analysis

D.1 Auto Meta Labeling and Unsupervised Object Detection

Fig. 8 shows four examples from the WOD validation set comparing ground truth, auto labels and unsupervised object detection results. In our unsupervised setting, both the auto labels and object detectors localize objects in a class-agnostic manner and are not limited by certain categories. In example (a) we show that auto labels and object detectors capture both pedestrians and vehicles.

In example (b), we demonstrate that even though there is *false positive* non-zero flow estimation, in AML we filter out many of these clusters during tracking and post-processing where very short tracks are dropped. The resulting detector

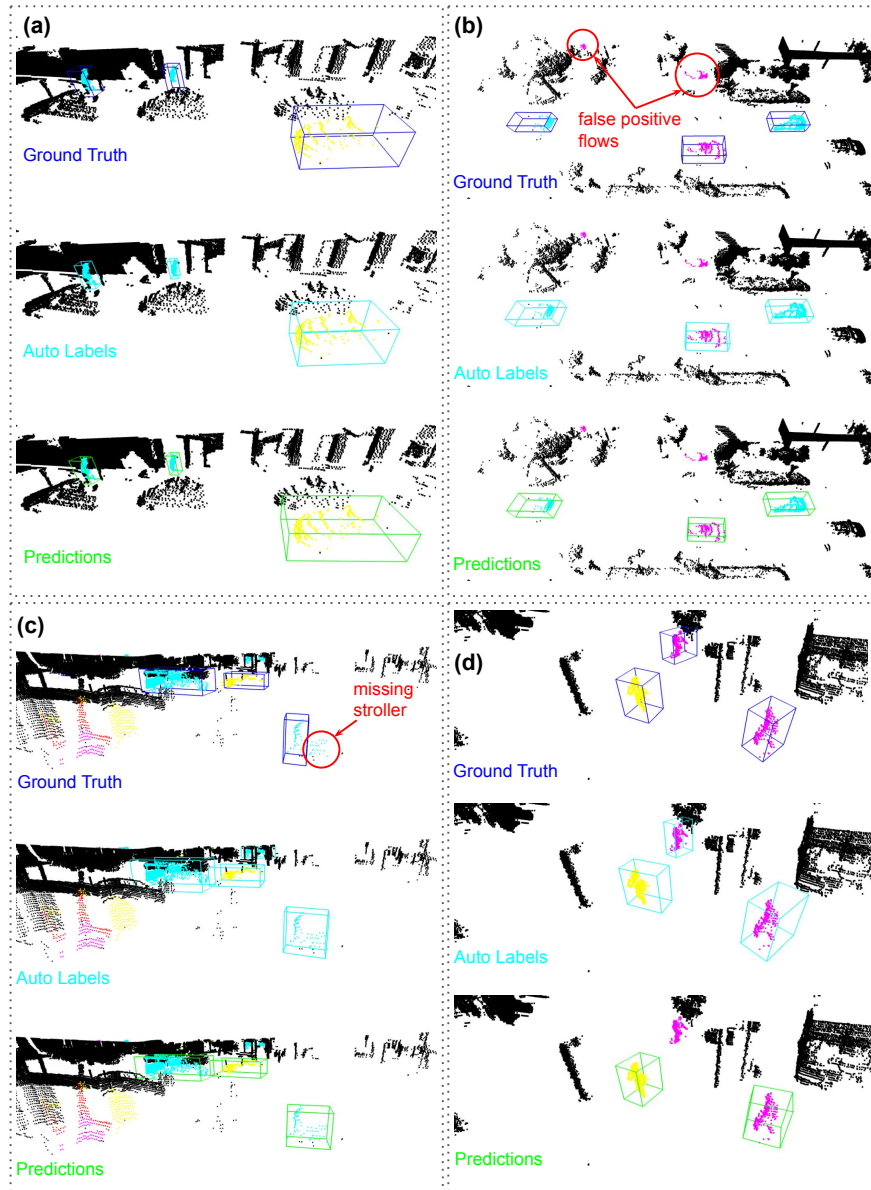


Fig. 8. Visualization of auto labels and detection predictions compared with the ground truth of moving objects. Points are colored by flow magnitudes and directions. Dark points are static. (a) The class-agnostic auto labels and unsupervised object detectors capture objects of multiple categories. (b) Although false positive flows occur, AML filters out many of them if they are inconsistent, and the detector learns to ignore these false positive flows. (c) Although the ground truth does not cover categories beyond vehicle, pedestrian, and cyclist, auto labels and our detector can capture open-set moving objects, such as the stroller. (d) An failure case that the detector may not be confident on objects with limited data amount, such as cyclists.

has also learned to ignore clusters of false positive flows. This example also shows that both auto labels and object detectors can infer the amodal boxes of some objects with only partial views.

Sometimes the unsupervised flow estimation captures *true positive* motion on points that are beyond the predefined categories in the ground truth. In example (c), a pedestrian is walking with a stroller while *stroller* is not a class included in the ground truth labels and therefore no bounding box is annotated around the stroller. NSFP++ has estimated the flow on the stroller, enabling AML and detectors to localize it. Since the stroller is held by the pedestrian with a similar speed, the clustering by design does not separate them apart. Clearly, it is safety-critical for autonomous vehicles to understand such moving objects in the open-set environment.

Example (d) shows a failure case where the detector could not confidently detect a cyclist. Although the auto labels have captured it, cyclists are less common than pedestrians and vehicles in the training set, which leads to inferior performance. We encourage future work to tackle the data imbalance issue under the unsupervised setting. Another failure pattern is that bounding boxes in auto labels tend to be larger than the actual size, due to the fact that temporal aggregation can include noise points. More advanced shape registration methods may help reduce noise and we leave it for future work.

D.2 Open-set Trajectory Prediction

Fig. 9 and 10 show behavior prediction qualitative results on the validation set of our newly created Anonymized Dataset. For each example scenario, we show the trajectory predictions of two models, *i.e.*, one trained only with a human-labeled category (the first column) and the other one trained with the combination of available human-labels and our AML auto labels for all other moving objects (the second column). The red and magenta trajectories represent the ground-truth routes taken by the autonomous vehicle and by an agent of interest, respectively. The blue and yellow trajectories are the possible predictions for the agent of interest and other agents in the scene. Fig. 9 shows three examples where human labels are available for the VRU category. As can be seen in all three examples, without using our unsupervised auto labels, the model tends to erroneously underestimate the speed (*e.g.* the first row), have difficulty in predicting trajectories consistent with the underlying roadgraph (*e.g.* the second row), and generating dangerous pedestrian-like trajectories along the pedestrian crosswalk (*e.g.* the third row). Fig. 10 shows the results when human labels are available only for the vehicle category. Similarly, when the model is only trained on the human labels (the first column), it cannot generalize well to the VRU class, predicting fast speeds and vehicle-like trajectories for VRUs. However, in both scenarios, adding auto labels (the second columns in Fig. 9 and 10) satisfactorily overcomes these errors, showing the effectiveness of our auto labels for training behavior prediction models in the open-set environment.

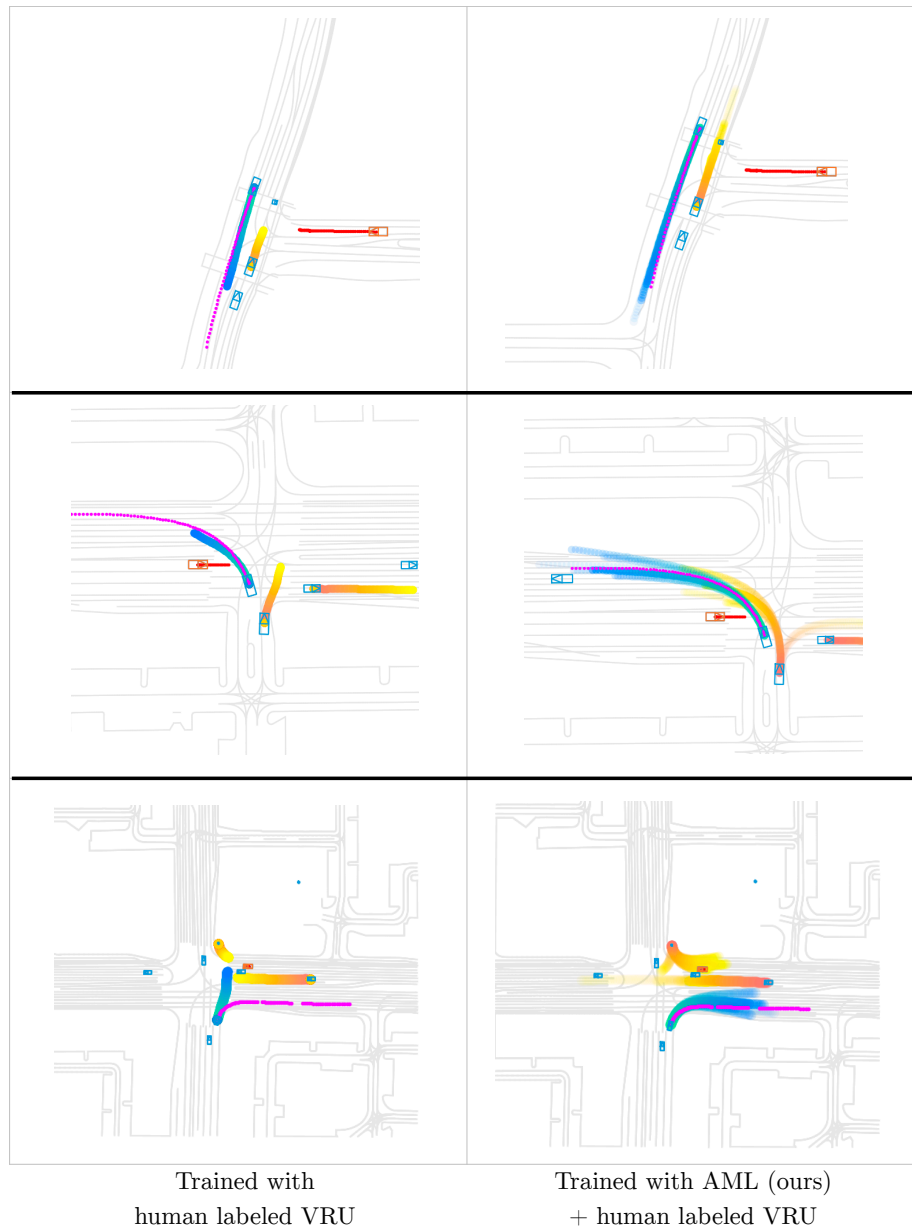


Fig. 9. Behavior prediction qualitative analysis. Trajectory predictions on three example scenarios for a model trained with human labeled VRUs *v.s.* a model trained with a combination of human labeled VRUs and our generated autolabels. Red and magenta dotted trajectories represent the ground-truth routes of the autonomous vehicle and agents, respectively. Blue and yellow trajectories are the predictions for the agent of interest and other agents, respectively.

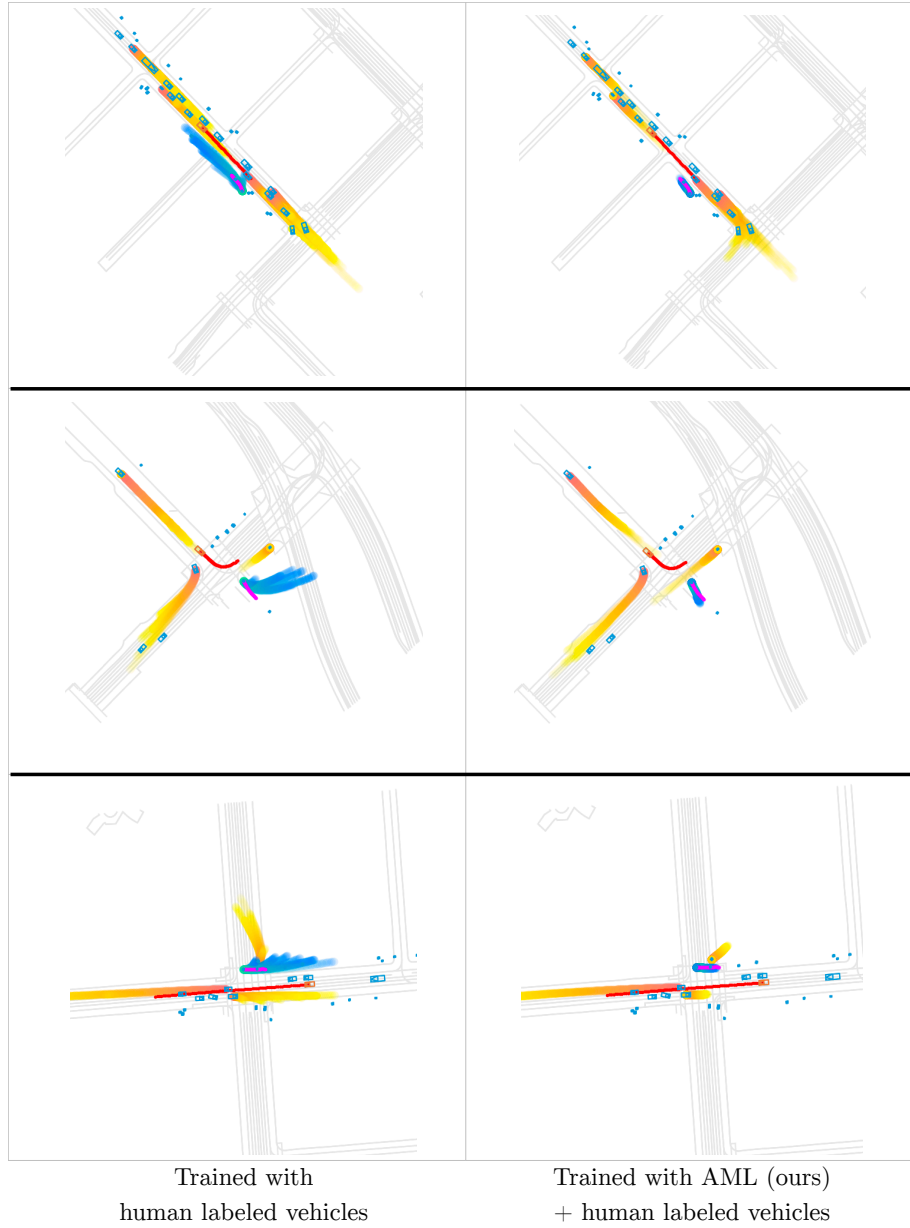


Fig. 10. Behavior prediction qualitative analysis. Trajectory predictions on three example scenarios for a model trained with human labeled vehicles *v.s.* a model trained with a combination of human labeled vehicles and our generated autolabels.

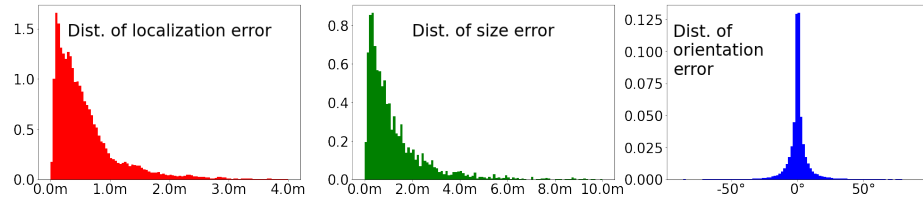


Fig. 11. Error distributions. y axis is probability density.

E Failure Analysis

In this section, we analyze the factors causing failure cases. Under threshold $\text{IoU}=0.4$, the precision/recall of our auto meta labels is 0.69/0.50. Part of the failure cases come from (1) false positive predictions that do not match any ground truth boxes; (2) false negatives where ground truth boxes are entirely missed. Moreover, there are predicted boxes overlapping with ground truth boxes while their IoUs are lower than the threshold. To have a better understanding, we breakdown 3D bounding box dimensions into three groups: localization (box center x, y, z), size (box length l , width w , height h), and orientation (BEV box heading r). Then, we summarize the distributions of localization, size, and orientation errors of the generated bounding boxes which overlap with at least one ground truth box (Fig. 11). The errors are computed between each pair of a predicted box and the ground truth box that has the highest IoU with the predicted box.

Localization. The localization error is defined as

$$\epsilon_{localization} = \sqrt{(x_{pr} - x_{gt})^2 + (y_{pr} - y_{gt})^2 + (z_{pr} - z_{gt})^2}. \quad (3)$$

As shown in Fig. 11, most of the localization errors are within 1.0 meter.

Size. The size error is defined as

$$\epsilon_{size} = \max\{|l_{pr} - l_{gt}| + |w_{pr} - w_{gt}| + |h_{pr} - h_{gt}|\}. \quad (4)$$

Many predictions have relatively high size errors. This is often caused by inclusion of noisy points in the registration step or missing parts of an object if the parts are always invisible throughout the object track.

Orientation. The orientation error is defined as

$$\epsilon_{orientation} = r_{pr} - r_{gt} \quad (5)$$

The orientation errors are generally small, as the orientation of each object is determined by the direction of the scene flows averaged over all points within

Table 9. Comparison between an oracle with GT box coordinates and baselines switching localization/size/orientation coordinates into AML predictions in turn. The performance drops show that the localization and size errors are dominant.

	3D mAPH@IoU=0.4
(Oracle) GT localization + GT size + GT orientation	46.1
Predicted localization + GT size + GT orientation	39.7 (-6.4)
GT localization + Predicted size + GT orientation	39.7 (-6.4)
GT localization + GT size + Predicted orientation	44.5 (-1.6)

the object bounding box. This error distribution verifies the quality of the unsupervised scene flows.

To find out the dominant factors leading to wrong auto meta labels, we construct several baselines by modifying the predictions and measure their label quality. The baselines are as follows:

1. (Oracle) GT localization + GT size + GT orientation: we replace the 7D values (x, y, z, l, w, h, r) of each predicted box with the values of its best matched ground truth box if any;
2. Predicted localization + GT size + GT orientation: we replace the (l, w, h, r) of each predicted box with the ground truth values. Comparison with the oracle will show the impact of localization errors;
3. GT localization + Predicted size + GT orientation: we replace the (x, y, z, r) of each predicted box with the ground truth values. Comparison with the oracle will show the impact of size errors;
4. GT localization + GT size + Predicted orientation: we replace the (x, y, z, l, w, h) of each predicted box with the ground truth values. Comparison with the oracle will show the impact of orientation errors.

We report the 3D mAPH@IoU=0.4 on the above baselines as mAPH additionally reflect the quality of heading prediction. We found that localization and size errors are dominant factors and future work may focus on improving the quality of auto labels on these fronts.