

I Can't Believe It's Not Scene Flow!

Ishan Khatri^{1,3*}, Kyle Vedder^{2*}, Neehar Peri³, Deva Ramanan³, James Hays⁴

¹Stack AV, ²University of Pennsylvania, ³CMU, ⁴Georgia Tech

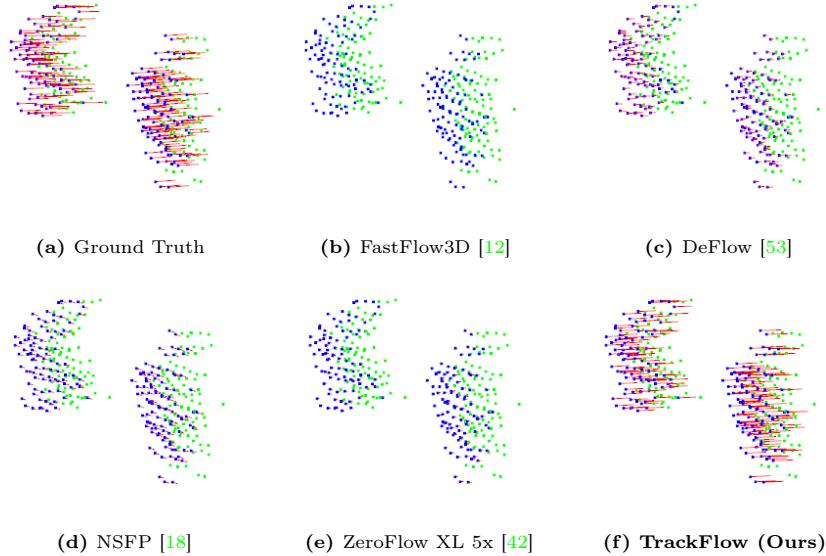


Fig. 1: We visualize an example of two pedestrians (walking from **left** to **right**), cherry-picked to have unusually high density lidar returns, making it particularly easy to estimate flow. We expect that state-of-the-art scene flow methods should work well in this case, but find that all prior art fails catastrophically. Notably, TrackFlow is the only method to estimate flow for these pedestrians.

Abstract. State-of-the-art scene flow methods broadly fail to describe the motion of small objects, and existing evaluation protocols hide this failure by averaging over many points. To address this limitation, we propose *Bucket Normalized EPE*, a new class-aware and speed-normalized evaluation protocol that better contextualizes error comparisons between object types that move at vastly different speeds. In addition, we propose *TrackFlow*, a frustratingly simple supervised scene flow baseline that combines a high-quality 3D object detector (trained using standard class re-balancing techniques) with a simple Kalman filter-based tracker. Notably, *TrackFlow* achieves state-of-the-art performance on existing metrics and shows large improvements over prior work on our proposed metric. Our results highlight that scene flow evaluation must be class and speed aware, and supervised scene flow methods must address point-level class imbalances. Our evaluation toolkit and code is available on [GitHub](#).

Keywords: LiDAR Scene Flow, Autonomous Vehicles

1 Introduction

Scene flow estimation is the task of describing a 3D motion field between temporally successive point clouds [2, 7, 12, 25, 42, 43, 52]. In theory, high quality flow estimators can provide a valuable signal about scene-level dynamics [12, 42] for both online [52] and offline [32] processing. Do state-of-the-art scene flow methods actually work well in practice?

Status Quo. Standard scene flow metrics suggest that existing methods can estimate motion to centimeter-level accuracy. For example, ZeroFlow XL 5x [42] achieves an average Threeway EPE [5] of only 4.9 centimeters (1.9 inches) and a Dynamic EPE (averaged over points moving faster than 0.5 m/s) of 11.7 centimeters (4.6 inches). Notably, these errors are relatively small compared to the scale of cars and pedestrians, implying that current scene flow methods produce high quality flow. On the scale of cars and people, these *feel* like tiny errors and seem to imply that current scene flow methods are high quality.

Bucket Normalized EPE. We visualize flow predictions from several state-of-the-art supervised (FastFlow3D [12], DeFlow [53]) and unsupervised (NSFP [18], ZeroFlow [42]) approaches and find that all methods underestimate flow for small objects (Fig. 1) with fewer lidar points (e.g. pedestrians and bicyclists). Surprisingly, existing scene flow metrics do not highlight such failure cases on these safety-critical categories because small objects only make up a tiny fraction of the dynamic points in a scene (Fig. 2). To address this limitation, we propose *Bucket Normalized EPE*, a new evaluation protocol that allows us to directly measure performance disparities across classes of different sizes and speed profiles. Specifically, *Bucket Normalized EPE* evaluates the *percentage* of described motion, allowing us to normalize comparisons between objects moving at different speeds. Our proposed evaluation metric takes inspiration from *mean Average Precision* (mAP), a metric commonly used to evaluate object detectors. Notably, unlike existing scene flow metrics, mAP equally weights the performance of large common objects like cars and small rare objects like strollers. Therefore, state-of-the-art 3D object detectors use data augmentation and class re-balancing techniques [56] to perform well on both common and rare classes.

TrackFlow. Based on this observation, we propose TrackFlow, a frustratingly simple baseline that generates scene flow estimates using rigid transformations to describe point-level motion within a 3D object track. Specifically, we run a state-of-the-art 3D object detector [46] followed by a simple 3D Kalman filter-based tracker [47] to generate object trajectories. Despite its simplicity, *TrackFlow* achieves state-of-the-art performance on Threeway EPE and significantly outperforms prior art on our Bucket Normalized EPE metric, capturing an additional 10% of total motion in general and an additional 20% of total motion on pedestrians (a 1.5 \times improvement). Importantly, our simple baseline’s state-of-the-art performance is an indictment of existing supervised scene flow methods. We argue that utilizing (well established) class re-balancing techniques can improve performance on rare safety-critical categories in real-world datasets, and evaluating scene flow methods using class and speed-aware metrics more closely reflects real-world performance.

Contributions. We present three primary contributions.

1. We highlight the qualitative failure of state-of-the-art scene flow methods on safety-critical categories like pedestrians and bicycles.
2. We introduce *Bucket Normalized EPE*, a new evaluation protocol that allows us to quantify this qualitative failure on small objects.
3. We propose *TrackFlow*, a frustratingly simple baseline that achieves state-of-the-art performance on standard metrics and significantly outperforms prior art on our class-aware *Bucket Normalized EPE* metric.

2 Related Work

2.1 Scene Flow Datasets and Ground Truth

Unlike next token prediction in language [38] or next frame prediction in vision [48], scene flow is not naïvely self-supervised: future observations do not provide ground truth scene flow. Therefore, ground truth motion descriptions must be provided by an oracle, typically from human annotators for real data [4, 29, 30, 40, 49] or a data generator for synthetic datasets [28, 55]. For real world datasets (typically from the autonomous vehicle domain) human annotations are provided in the form of 3D bounding boxes and tracks for every object in the scene [5]. Consequently, the generated ground truth flow is assumed to be rigid, even in the case of non-rigid motion like pedestrian gaits.

2.2 Scene Flow Estimation

Given point clouds P_t and P_{t+1} , scene flow estimators predict $\hat{F}_{t,t+1}$, a 3D vector per point in P_t that describes its motion from t to $t + 1$ [6]. Performance is typically measured using Average Endpoint Error (EPE) which is the L_2 norm between the predicted ($\hat{F}_{t,t+1}$) and ground truth flow ($F_{t,t+1}^*$), as in Equation 1.

$$\text{Average EPE}(P_t) = \frac{1}{\|P_t\|} \sum_{p \in P_t} \left\| \hat{F}_{t,t+1}(p) - F_{t,t+1}^*(p) \right\|_2. \quad (1)$$

Current state-of-the-art methods for scene flow estimation broadly fall into one of two categories: supervised and unsupervised.

Supervised Scene Flow methods train feedforward networks to perform flow vector regression based on ground truth annotations [1, 3, 10, 12, 15, 17, 21, 25, 37, 41, 45, 50, 53]. Many of these networks utilize custom point operations such as point-based convolutions [10, 15, 21, 25], making them intractable to train on large point clouds. In contrast, FastFlow3D [12] uses a feedforward architecture based on PointPillars [16], an efficient lidar detector architecture, to train and predict flow on real-world large-scale point clouds. FastFlow3D’s speed and quality make it a popular base architecture for both unsupervised and supervised methods like ZeroFlow [42] and DeFlow [53], respectively.

Unsupervised Scene Flow methods tend to use online optimization against surrogate objectives such as Chamfer distance [18], cycle-consistency [31], distance transforms [19], or other hand-designed heuristics [5, 9, 36]. For example, Neural Scene Flow Prior (NSFP) [18] provides high quality scene flow estimates by optimizing a small ReLU MLP at test time to minimize Chamfer distance and maintain cycle-consistency. Other unsupervised methods like ZeroFlow [42] indirectly leverage online optimization. Vedder et. al [42] introduces *Scene Flow via Distillation*, a framework that uses a slow optimization-based method to pseudolabel unlabeled point cloud pairs and trains a fast feedforward network with these pseudolabels.

2.3 Scene Flow Evaluation Metrics

In real-world scenes, most points belong to the static background. Consequently, simply computing Average EPE (Equation 1) over all points is dominated by background points. In order to separately measure non-ego dynamics, Chodosh et al. [5] introduces *Threeway EPE*, which computes a mean over the Average EPE for three disjoint classes of points: *Foreground Dynamic* (points inside bounding box labels moving greater than 0.5m/s), *Foreground Static* (points inside bounding box labels moving less than 0.5m/s), and *Background Static*. We extend Threeway EPE to consider different class and speed profiles.

2.4 3D Object Detection and Tracking

Object detectors have advanced techniques for training with imbalanced datasets. Notably, modern object detectors use carefully designed losses to mitigate foreground-vs-background imbalances in proposal generation, and data augmentation strategies to train with long-tailed taxonomies. Existing methods address imbalanced foreground-vs-background region proposals using Focal Loss [22] to upweight the importance of foreground regions. More recently, 3D object detectors use class-balanced sampling [56] and copy-paste augmentation to upsample and rebalance the distribution of examples per class. In addition, state-of-the-art 3D object detectors take advantage of multi-modal data to improve detection [27, 33, 44] of small and rare categories. Since many state-of-the-art tracking algorithms [47] follow the tracking-by-detection paradigm, improving detection quality also significantly improves tracking performance.

3 Bucket Normalized EPE: Small Objects (Should) Matter in Scene Flow

As shown in Fig. 1 (and further in Fig. 8), existing scene flow methods consistently struggle to describe the motion of safety-critical objects like pedestrians. However, these failures are not captured by Threeway EPE because these objects are small and have few points. Specifically, Threeway EPE’s *Foreground Dynamic* category is dominated by large, common objects with many points like

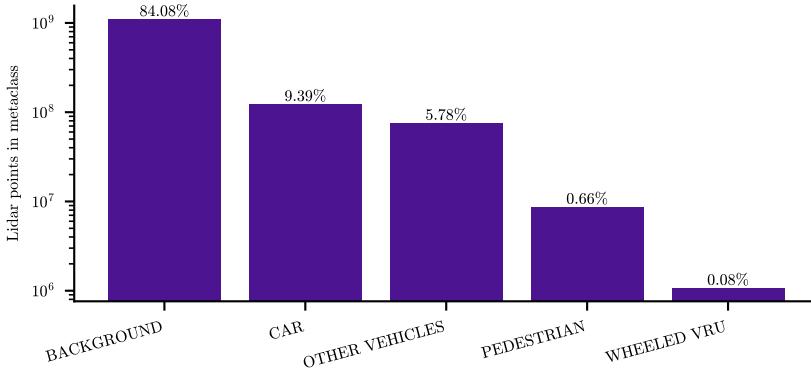


Fig. 2: Number of points from each semantic meta-class for Argoverse 2’s *val* split. Although **PEDESTRIAN** instances are common, they contribute less than 1% of the total number of points owing to their small instance size relative to **CAR** and **OTHER VEHICLES**. Number of points (Y axis) shown on a log scale.

cars and other vehicles. As shown in Fig. 2, 15% of all points are from cars or other vehicles (dominating *Foreground Dynamic*’s Average EPE), while fewer than 1% of points are from pedestrians and other vulnerable road users (VRUs).

Additionally, Threeway EPE fails to account for large differences in speed across objects. For example, a 0.5m/s estimation error on a car moving 20 m/s is negligible (<2.5%), while a 0.5m/s estimation error on a pedestrian moving 0.5m/s fails to describe 100% of the pedestrian’s motion. However, Threeway EPE treats both estimation errors equally.

We address these two limitations with our *Bucket Normalized EPE* metric. First, our proposed metric breaks down the object distribution using a taxonomy that human labelers have deemed important (similar to *mean Average Precision* [23], see Appendix C for discussion on semantics-free evaluation). Second, our proposed metric allows us to contextualize the percentage of object motion being described by normalizing for the speed of the object, allowing us to directly compare performance across object categories.

We implement our class-aware and speed-normalized metric by accumulating every point into a class-speed matrix (e.g. Appendix B, Table 4) based on its ground truth speed and class, recording an Average EPE as well as a per-bucket average speed. To summarize these results, we report two numbers per class:

- *Static EPE*, taken directly from the Average EPE of the first speed bucket for that class (i.e. the first column of Appendix B, Table 4)
- *Dynamic Normalized EPE*, computed from a mean over the Normalized EPE ($\frac{\text{Average EPE}}{\text{average speed}}$) of each non-empty speed bucket (i.e. an average across the Normalized EPEs of the second column onwards in Appendix B, Table 4)

Dynamic Normalized EPE measures the fraction of motion *not* described by the estimated flow vectors across the entire speed spectrum. A method that only

Class	Static (Avg EPE)	Dynamic (Norm EPE)
BACKGROUND	0.002402	-
CAR	0.018442	0.182092
OTHER VEHICLES	0.081475	0.312882
PEDESTRIAN	0.052842	0.396849
WHEELED VRU	0.062573	0.257647

Table 1: TrackFlow’s Bucket Normalized EPE on the Argoverse 2 *test* split. Similar to Threeway EPE, we breakdown our evaluation into static and dynamic buckets. However, we also further breakdown performance by meta-categories and normalize by speed to compare performance disparities on safety-critical categories. TrackFlow is able to capture most dynamic car motion (lower is better), but performs considerably worse on other vehicles and pedestrian.

predicts ego motion (e.g. $\vec{0}$ after ego-motion compensation) will achieve 1.0 Dynamic Normalized EPE, and a method that perfectly describes all motion will have 0.0 Dynamic Normalized EPE. Methods may achieve errors greater than 1.0 by predicting errors with a magnitude greater than the average speed. For example, a method that describes the negative vector of true motion will get exactly 2.0 Dynamic Normalized EPE (every bucket’s Average EPE will be exactly $2 \times$ the magnitude of the average speed). The range of Dynamic Normalized EPE is between 0 (perfect) and ∞ , and is undefined for buckets without any points. After normalization, Dynamic Normalized EPE can be directly compared across classes.

We provide an example per-class performance breakdown in Table 1 for TrackFlow (Section 4). Results can be further summarized into a single tuple of *mean Static EPE* and *mean Dynamic Normalized EPE* by taking a mean across classes (similar to *mean Average Precision* [23]). TrackFlow has a mean Static EPE of 0.076277 and a mean Dyannmic Normalized EPE of 0.287368. We rank methods according to their mean Dynamic Normalized EPE.

4 TrackFlow: Scene Flow via Tracking

To highlight the failure of current supervised scene flow methods on smaller objects, we propose *Scene Flow via Tracking*, a simple framework that uses bounding box track motion from off-the-shelf 3D detectors and trackers to generate scene flow estimates (Fig. 3). We instantiate *Scene Flow via Tracking* with LE3DE2E [46]¹, a state-of-the-art 3D detector, and AB3DMOT [47], a Kalman filter-based 3D tracker. As shown in Section 5, *TrackFlow* achieves state-of-the-art performance on Threeway EPE and beats all prior art by a large margin on Bucket Normalized EPE.

¹ LE3DE2E [46] is the winning method from the *Argoverse 2 2023 3D Detection, Tracking and Forecasting challenge* [33–35].

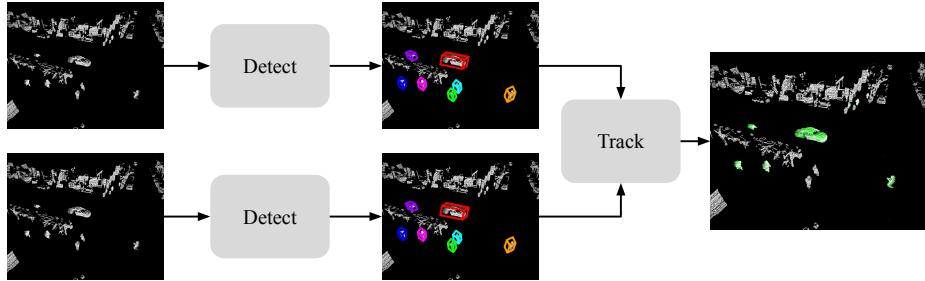


Fig. 3: Overview of the *Scene Flow via Tracking* framework. Our proposed framework generates scene flow estimates using rigid transformations to describe point-level motion within a 3D object track.

Scene Flow via Tracking works well in practice because it mimics the ground truth flow annotation process. Specifically, ground truth flow is generated using rigid transforms to describe point-level motion within ground truth 3D object tracks (Section 2.1). Therefore, a perfect 3D detector and tracker will achieve perfect flow. However, the power of TrackFlow isn't just derived from its use of bounding boxes; it also greatly benefits from recent advances in class-imbalanced learning [56]. As discussed in Section 2.4, modern detectors are trained with a variety of data augmentation techniques to achieve high precision and recall on all semantic class. TrackFlow leverages the strength of modern 3D detectors to significantly outperform prior art on pedestrians and other small objects.

Interestingly, we find that the Scene Flow via Tracking framework performs best when using detectors tuned to a low confidence threshold. Typically, detectors are optimized to only predict high confidence boxes (0.7 - 0.9) to minimize the number of false positives. However, our method works best when setting the confidence threshold lower (0.2 for TrackFlow) to increase recall. Specifically, we find that detectors with higher recall and more accurate heading estimation are better suited for Scene Flow via Tracking. We explore detector choice and ablate the impact of confidence thresholds further in Section 5.3.

5 Experiments

In this section, we compare TrackFlow against state-of-the-art supervised and unsupervised scene flow methods like FastFlow3D [12], DeFlow [53], NSFP [18], and ZeroFlow [42] on the Argoverse 2 benchmark [49]².

5.1 TrackFlow Achieves SOTA Performance on Threeway EPE

TrackFlow is state-of-the-art on Threeway EPE (Fig. 4a) on the Argoverse 2 benchmark [49], achieving an overall reduction of 0.0015m (0.15cm, or 1.5mm)

² All evaluations are performed with a maximum radius of 35m from the ego vehicle to maintain consistency with Chodosh et al. [5].

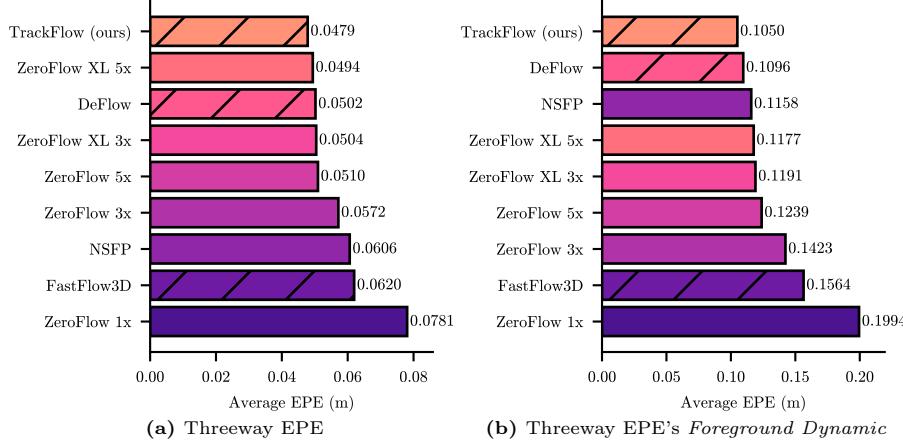


Fig. 4: *Threeway EPE* and *Threeway EPE’s Foreground Dynamic* performance of recent supervised and unsupervised scene flow methods on Argoverse 2’s *test* split. Supervised methods shown with hatching. Lower is better. Method color is consistent between plots. We find that all recent methods achieve 5cm error on *Threeway EPE*, suggesting that these approaches work well in-the-wild. However, this number hides the failure of these methods to describe small object motion.

over the next best method, ZeroFlow XL 5x. Notably, this improvement can be attributed to significantly better *Threeway EPE’s Dynamic Foreground* (Fig. 4b). Is this performance difference meaningful?

Based on our reduction of 1.5mm on *Threeway EPE* (about $4\times$ the thickness of a human fingernail), it would seem that TrackFlow is only an incremental improvement over prior art. However, TrackFlow qualitatively outperforms prior work on important small objects such as pedestrians (Fig. 1, Fig. 8). As shown in the next section, our proposed evaluation protocol *Bucket Normalized EPE* makes it quantitatively clear that TrackFlow performs significantly better on safety-critical categories like pedestrians and VRUs.

5.2 Bucket Normalized EPE Highlights Failures on Small Objects

Evaluating existing state-of-the-art methods on our class-aware, speed-normalized evaluation, Bucket Normalized EPE, makes it clear that TrackFlow meaningfully outperforms prior art (Fig. 6) — TrackFlow correctly describes almost 10% additional total motion across meta-classes compared to DeFlow [53]. This difference in dynamic performance becomes even more clear when broken down by meta-class: Fig. 5 shows that TrackFlow is the only method able to describe more than 50% of pedestrian motion, beating DeFlow [53] by more than 20% (Fig. 5c), a $1.5\times$ improvement. Similarly, other state-of-the-art methods like NSFP [18] and ZeroFlow XL 5x [42] describe less than 30% and 20% of pedestrian motion, respectively.

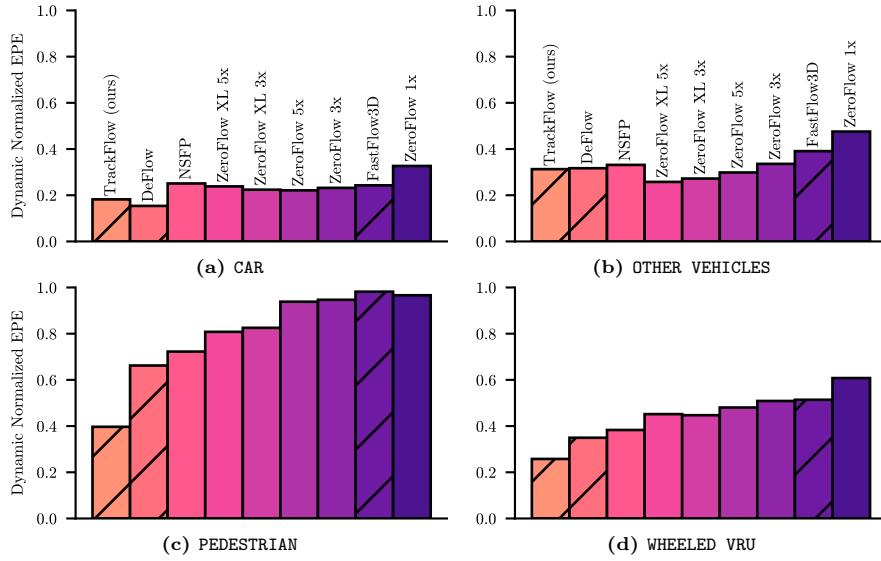


Fig. 5: Per meta-class Dynamic Normalized EPE of recent supervised and unsupervised scene flow estimation methods on Argoverse 2’s *test* split. Supervised methods shown with hatching. Lower is better. Method color and position is consistent between plots. TrackFlow significantly improves over prior work on both pedestrian and wheeled VRUs. Notably, Bucket Normalized EPE quantitatively demonstrates significant method performance differences not highlighted in Threeway EPE.

Bucket Normalized EPE allows practitioners to effectively compare performance between methods that were almost indistinguishable under Threeway EPE. For example, if you only care about flow performance on cars, DeFlow out-performs all other methods including TrackFlow (Fig. 5a), while ZeroFlow XL 5x out-performs all other methods on larger vehicles (Fig. 5b).

5.3 What Makes a Good Detector for TrackFlow?

As discussed in Section 4, we tune *Scene Flow via Tracking* with a low confidence threshold to maximize recall. What makes a good detector for TrackFlow?

We ablate the impact of detector quality on TrackFlow by replacing LE3DE2E [46] with BEVFusion [26]. We call this new approach *TrackFlowBEVF*. BEVFusion only has 2% lower mAP than LE3DE2E on the AV2 detection leaderboard³, but we find that TrackFlowBEVF performs significantly worse than TrackFlow, with 10% to 22% drops in performance on Dynamic Normalized EPE (Table 2).

This significant degradation is the result of BEVFusion’s poor recall at low confidence thresholds (Table 3). In contrast, LE3DE2E has very high recall at low thresholds (Fig. 7), producing many candidate boxes for pedestrians in the

³ BEVFusion [26] was second on the *Argoverse 2 2023 3D Detection, Tracking and Forecasting challenge* [33–35].

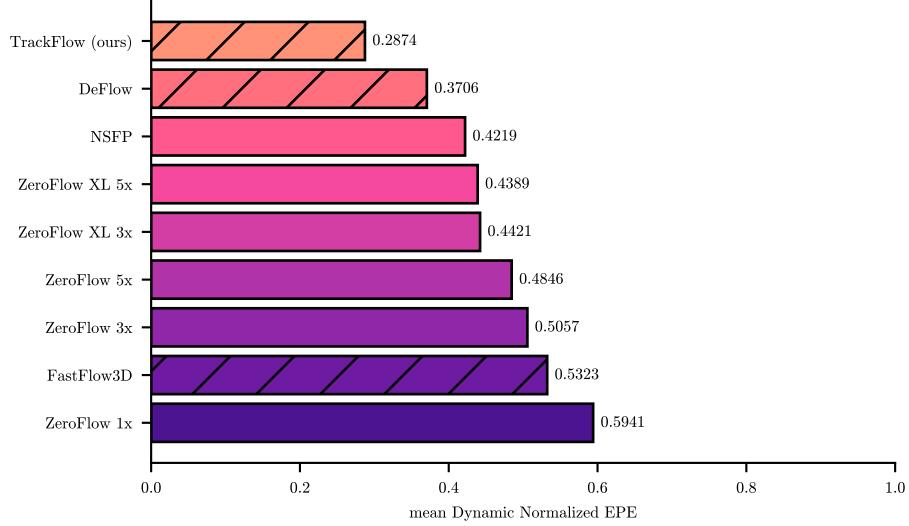


Fig. 6: Average Dynamic Normalized EPE of recent supervised and unsupervised scene flow estimation methods on Argoverse 2’s *test* split. Supervised methods shown with hatching. Lower is better. Our simple baseline achieves state-of-the-art performance, suggesting that supervised scene flow methods should embrace point-level class rebalancing.

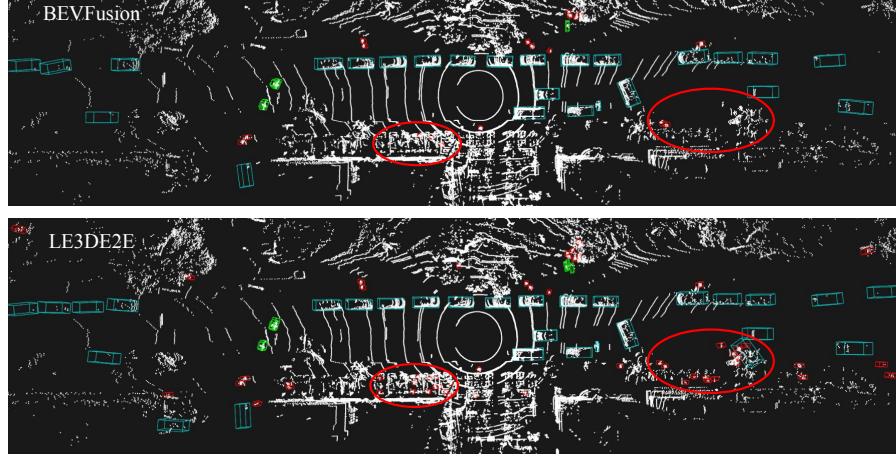


Fig. 7: A qualitative comparison of the recall of BEVFusion and LE3DE2E. LE3DE2E has much higher recall, allowing it to pick out pedestrians BEVFusion missed (circled in red), and better quality box heading estimates. Both detectors are using a confidence threshold of 0.2.

scene. BEVFusion’s false negatives are extremely costly to TrackFlowBEVF, as they result in $\vec{0}$ flow estimates that miss 100% of each false positive pedestrian’s motion.

Class	Static (Avg EPE)	Dynamic (Norm EPE)
BACKGROUND	-0.000228	-
CAR	+0.039049	+0.117944
OTHER VEHICLES	+0.009013	+0.224830
PEDESTRIAN	+0.007187	+0.224250
WHEELED VRU	-0.025889	+0.151373

Table 2: Relative Bucket Normalized EPE performance of TrackFlowBEVF compared to TrackFlow, on the Argoverse 2’s *test* split. Increases in error (worse) are shown with a + in red, and decreases in error (better) are shown with a - in green. TrackFlow’s absolute results are shown in Table 1.BEVFusion only has 2% lower mAP than LE3DE2E on the AV2 detection leaderboard, but performs significantly worse than TrackFlow on Dynamic Normalized EPE.

Confidence	Mean Dynamic Norm EPE
0.1	0.4816
0.2	0.4643
0.3	0.6008
0.4	0.8176

Table 3: Mean Dynamic Bucketed EPE values for TrackFlowBEVF using various confidence thresholds for the detector. Lower confidences with higher recall significantly improve Dynamic Norm EPE performance.

More broadly, a good detector for Scene Flow via Tracking isn’t necessarily one with a high mAP; it’s one with very high recall and accurate heading estimates. Notably, these error characteristics enable the tracker to reject false positives. We believe this interaction between the detector and tracker is an important yet subtle point — two detectors may have the same mAP, but dramatically different performance in our Scene Flow via Tracking framework.

6 Conclusion

In this work, we highlight that current scene flow methods consistently fail to describe motion on pedestrians and other small objects. We demonstrate that current standard evaluation metrics hide this failure and present Bucket Normalized EPE, a new class-aware, speed normalized evaluation protocol, to quantify this failure. In addition, we present TrackFlow, a frustratingly simple supervised scene flow baseline that achieves state-of-the-art on Threeway EPE and Bucket Normalized EPE. We argue that current evaluation protocols fail to reveal performance across the distribution of safety-critical objects, and do not contextualize absolute errors in the context of an object’s speed. Moreover, we highlight that class and speed aware evaluation is important *even if a method has zero human*

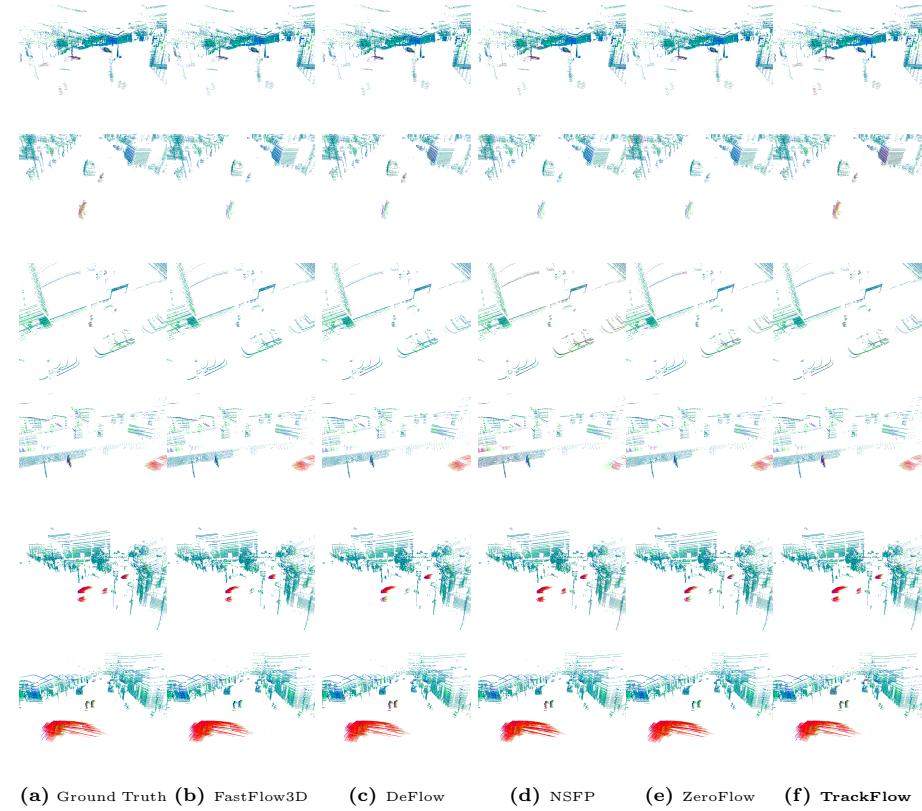


Fig. 8: Visualizations of different methods on diverse scenes in Argoverse 2. Each method is estimating flow from the **blue** to the **green** point cloud.

- Row 1: Two pedestrians in left foreground with cars moving in the background. TrackFlow is the only method able to describe the pedestrian motion.
- Row 2: Three pedestrians walking across an intersection in front of a stationary car. DeFlow is able to capture the furthest pedestrian, but only TrackFlow is able to capture the motion of all three. TrackFlow also falsely estimates motion of the moving box truck in the background.
- Row 3: Top view of pedestrians walking down the sidewalk between a building and several cars parked in the street. TrackFlow is the only method able to describe the pedestrian motion.
- Row 4: Pedestrians walking down the sidewalk next to a moving car. TrackFlow is the only method able to describe the pedestrian motion.
- Row 5: Two bicyclists riding across an intersection next to driving cars. Most methods are able to capture the training bicyclists and the moving cars, but only NSFP and TrackFlow are able to capture the lead bicyclist.
- Row 6: Two pedestrians walk across an intersection while a car drives parallel to them. All methods capture the car motion, but only DeFlow, NSFP, and TrackFlow capture most of the pedestrian motion. TrackFlow also falsely estimates motion of one of the parked cars far down the street in the background.

supervision. Importantly, we cannot expect any method to meaningfully generalize to the long tail of unknown objects if it cannot provide high quality motion descriptions on a known set of objects. Lastly, TrackFlow outperforms prior art by a wide margin because it leverages recent advances in class imbalanced learning. Our approach highlights that supervised scene flow methods should adopt many of the lessons learned by the detection community to properly address class and point imbalances.

6.1 Limitations

TrackFlow only predicts rigid flow for objects within LE3DE2E’s fixed taxonomy because it uses a closed-world bounding box based detector. However, as discussed in Appendix D, these limitations can be addressed with a different detector architectures, and is not a fundamental limitations of the Scene Flow via Tracking framework.

Acknowledgements: This work was supported in part by funding from the NSF GRFP (Grant No. DGE2140739). This work was in part supported by the Army Research Office under MURI award W911NF20-1-0080. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the Army or the US government.

References

1. Battarawty, R., Schuster, R., Mahani, M.A.N., Stricker, D.: RMS-FlowNet: Efficient and Robust Multi-Scale Scene Flow Estimation for Large-Scale Point Clouds. In: Int. Conf. Rob. Aut. pp. 883–889. IEEE (2022) [3](#)
2. Baur, S.A., Emmerichs, D.J., Moosmann, F., Pinggera, P., Ommer, B., Geiger, A.: SLIM: Self-supervised LiDAR scene flow and motion segmentation. In: Int. Conf. Comput. Vis. pp. 13126–13136 (2021) [2](#)
3. Behl, A., Paschalidou, D., Donné, S., Geiger, A.: Pointflownet: Learning representations for rigid motion estimation from point clouds. In: Int. Conf. Comput. Vis. pp. 7962–7971 (2019) [3](#)
4. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuScenes: A multimodal dataset for autonomous driving. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 11621–11631 (2020) [3](#)
5. Chodosh, N., Ramanan, D., Lucey, S.: Re-Evaluating LiDAR Scene Flow for Autonomous Driving. arXiv preprint (2023) [2, 3, 4, 7](#)
6. Dewan, A., Caselitz, T., Tipaldi, G.D., Burgard, W.: Rigid scene flow for 3d lidar scans. In: Int. Conf. Intel. Rob. Sys. pp. 1765–1770. IEEE (2016) [3](#)
7. Erçelik, E., Yurtsever, E., Liu, M., Yang, Z., Zhang, H., Topçam, P., Listl, M., Çaylı, Y.K., Knoll, A.: 3D Object Detection with a Self-supervised Lidar Scene Flow Backbone. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022. pp. 247–265. Springer Nature Switzerland, Cham (2022) [2](#)
8. Girshick, R.: Fast R-CNN. In: 2015 IEEE International Conference on Computer Vision (ICCV). pp. 1440–1448 (2015) [19](#)

9. Gojcic, Z., Litany, O., Wieser, A., Guibas, L.J., Birdal, T.: Weakly supervised learning of rigid 3d scene flow. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5692–5703 (2021) [4](#)
10. 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: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3254–3263 (2019) [3](#)
11. Huang, X., Wang, Y., Guizilini, V.C., Ambrus, R.A., Gaidon, A., Solomon, J.: Representation Learning for Object Detection from Unlabeled Point Cloud Sequences. In: Liu, K., Kulic, D., Ichnowski, J. (eds.) Proceedings of The 6th Conference on Robot Learning (CoRL). Proceedings of Machine Learning Research, vol. 205, pp. 1277–1288 (2023) [19](#)
12. 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 (12 2021) [1](#), [2](#), [3](#), [7](#), [17](#)
13. Kim, D., Lin, T.Y., Angelova, A., Kweon, I.S., Kuo, W.: Learning open-world object proposals without learning to classify. IEEE Robotics and Automation Letters (RA-L) (2022) [19](#)
14. Kim, J., Woo, J., Shin, U., Oh, J., Im, S.: Flow4D: Leveraging 4D Voxel Network for LiDAR Scene Flow Estimation (2024), <https://arxiv.org/abs/2407.07995> [17](#)
15. Kittenplon, Y., Eldar, Y.C., Raviv, D.: Flowstep3d: Model unrolling for self-supervised scene flow estimation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4114–4123 (2021) [3](#)
16. Lang, A., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: PointPillars: Fast Encoders for Object Detection From Point Clouds. In: Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12689–12697 (2019) [3](#)
17. Li, R., Lin, G., He, T., Liu, F., Shen, C.: HCRF-Flow: Scene flow from point clouds with continuous high-order CRFs and position-aware flow embedding. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 364–373 (2021) [3](#)
18. Li, X., Pontes, J.K., Lucey, S.: Neural Scene Flow Prior. Advances in Neural Information Processing Systems **34** (2021) [1](#), [2](#), [4](#), [7](#), [8](#)
19. Li, X., Zheng, J., Ferroni, F., Pontes, J.K., Lucey, S.: Fast neural scene flow. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9878–9890 (October 2023) [4](#)
20. Li, X., Zheng, J., Ferroni, F., Pontes, J.K., Lucey, S.: Fast Neural Scene Flow. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9878–9890 (October 2023) [17](#)
21. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: Pointcnn: Convolution on x-transformed points. Adv. Neural Inform. Process. Syst. **31** (2018) [3](#)
22. Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal Loss for Dense Object Detection. In: ICCV 2017. pp. 2999–3007 (2017) [4](#)
23. Lin, T., Maire, M., Belongie, S.J., Bourdev, L.D., Girshick, R.B., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common Objects in Context. Corr (2014) [5](#), [6](#)
24. Lin, Y., Caesar, H.: ICP-Flow: LiDAR Scene Flow Estimation with ICP (2024) [17](#)
25. Liu, X., Qi, C.R., Guibas, L.J.: FlowNet3D: Learning Scene Flow in 3D Point Clouds. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019) [2](#), [3](#)

26. Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D., Han, S.: Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In: IEEE International Conference on Robotics and Automation (ICRA) (2023) [9](#)
27. Ma, Y., Peri, N., Wei, S., Hua, W., Ramanan, D., Li, Y., Kong, S.: Long-tailed 3d detection via 2d late fusion. arXiv preprint arXiv:2312.10986 (2023) [4](#)
28. Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2016) [3](#)
29. Menze, M., Heipke, C., Geiger, A.: Joint 3D Estimation of Vehicles and Scene Flow. In: ISPRS Workshop on Image Sequence Analysis (ISA) (2015) [3](#)
30. Menze, M., Heipke, C., Geiger, A.: Object Scene Flow. ISPRS Journal of Photogrammetry and Remote Sensing (JPRS) (2018) [3](#)
31. Mittal, H., Okorn, B., Held, D.: Just Go With the Flow: Self-Supervised Scene Flow Estimation. In: IEEE Conf. Comput. Vis. Pattern Recog. (June 2020) [4](#)
32. Najibi, M., Ji, J., Zhou, Y., Qi, C.R., Yan, X., Ettinger, S., Anguelov, D.: Motion Inspired Unsupervised Perception and Prediction in Autonomous Driving. European Conference on Computer Vision (ECCV) (2022) [2](#)
33. Peri, N., Dave, A., Ramanan, D., Kong, S.: Towards Long Tailed 3D Detection. CoRL (2022) [4, 6, 9](#)
34. Peri, N., Li, M., Wilson, B., Wang, Y.X., Hays, J., Ramanan, D.: An empirical analysis of range for 3d object detection. arXiv preprint arXiv:2308.04054 (2023) [6, 9](#)
35. Peri, N., Luiten, J., Li, M., Ošep, A., Leal-Taixé, L., Ramanan, D.: Forecasting from lidar via future object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 17202–17211 (June 2022) [6, 9](#)
36. Pontes, J.K., Hays, J., Lucey, S.: Scene flow from point clouds with or without learning. In: Int. Conf. 3D Vis. pp. 261–270. IEEE (2020) [4](#)
37. Puy, G., Boulch, A., Marlet, R.: Flot: Scene flow on point clouds guided by optimal transport. In: Eur. Conf. Comput. Vis. pp. 527–544. Springer (2020) [3](#)
38. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018) [3](#)
39. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(6), 1137–1149 (2017) [19](#)
40. Sun, P., Kretzschmar, 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: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020) [3](#)
41. Tishchenko, I., Lombardi, S., Oswald, M.R., Pollefeys, M.: Self-supervised learning of non-rigid residual flow and ego-motion. In: Int. Conf. 3D Vis. pp. 150–159. IEEE (2020) [3](#)
42. Vedder, K., Peri, N., Chodosh, N., Khatri, I., Eaton, E., Jayaraman, D., Liu, Y., Ramanan, D., Hays, J.: ZeroFlow: Scalable Scene Flow via Distillation. In: Twelfth International Conference on Learning Representations (ICLR) (2024) [1, 2, 3, 4, 7, 8, 17](#)
43. Vedula, S., Baker, S., Rander, P., Collins, R., Kanade, T.: Three-dimensional scene flow. In: Int. Conf. Comput. Vis. vol. 2, pp. 722–729. IEEE (1999) [2](#)

44. Vora, S., Lang, A.H., Helou, B., Bejbom, O.: Pointpainting: Sequential fusion for 3d object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4604–4612 (2020) [4](#)
45. Wang, J., Li, X., Sullivan, A., Abbott, L., Chen, S.: PointMotionNet: Point-Wise Motion Learning for Large-Scale LiDAR Point Clouds Sequences. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 4418–4427 (2022) [3](#)
46. Wang, Z., Chen, F., Lertniphonphan, K., Chen, S., Bao, J., Zheng, P., Zhang, J., Huang, K., Zhang, T.: Technical report for argoverse challenges on unified sensor-based detection, tracking, and forecasting (2023) [2](#), [6](#), [9](#)
47. Weng, X., Wang, J., Held, D., Kitani, K.: 3D Multi-Object Tracking: A Baseline and New Evaluation Metrics. IROS (2020) [2](#), [4](#), [6](#)
48. Weng, X., Wang, J., Levine, S., Kitani, K., Rhinehart, N.: Inverting the pose forecasting pipeline with spf2: Sequential pointcloud forecasting for sequential pose forecasting. In: Conference on robot learning. pp. 11–20. PMLR (2021) [3](#)
49. Wilson, B., Qi, W., Agarwal, T., Lambert, J., Singh, J., Khandelwal, S., Pan, B., Kumar, R., Hartnett, A., Pontes, J.K., Ramanan, D., Carr, P., Hays, J.: Argoverse 2: Next Generation Datasets for Self-driving Perception and Forecasting. In: Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021) (2021) [3](#), [7](#)
50. 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: Eur. Conf. Comput. Vis. pp. 88–107. Springer (2020) [3](#)
51. Yang, J., Zeng, A., Zhang, R., Zhang, L.: UniPose: Detection Any Keypoints. arXiv preprint arXiv:2310.08530 (2023) [19](#)
52. Zhai, G., Kong, X., Cui, J., Liu, Y., Yang, Z.: FlowMOT: 3D Multi-Object Tracking by Scene Flow Association. ArXiv **abs/2012.07541** (2020) [2](#)
53. Zhang, Q., Yang, Y., Fang, H., Geng, R., Jensfelt, P.: DeFlow: Decoder of Scene Flow Network in Autonomous Driving. ICRA (2024) [1](#), [2](#), [3](#), [7](#), [8](#), [17](#)
54. Zhang, Q., Yang, Y., Li, P., Andersson, O., Jensfelt, P.: Seflow: A self-supervised scene flow method in autonomous driving. arXiv preprint arXiv:2407.01702 (2024) [17](#)
55. Zheng, Y., Harley, A.W., Shen, B., Wetzstein, G., Guibas, L.J.: PointOdyssey: A Large-Scale Synthetic Dataset for Long-Term Point Tracking. In: ICCV (2023) [3](#)
56. Zhu, B., Jiang, Z., Zhou, X., Li, Z., Yu, G.: Class-balanced Grouping and Sampling for Point Cloud 3D Object Detection. arXiv preprint arXiv:1908.09492 (2019) [2](#), [4](#), [7](#)

A Argoverse 2 2024 Scene Flow Challenge

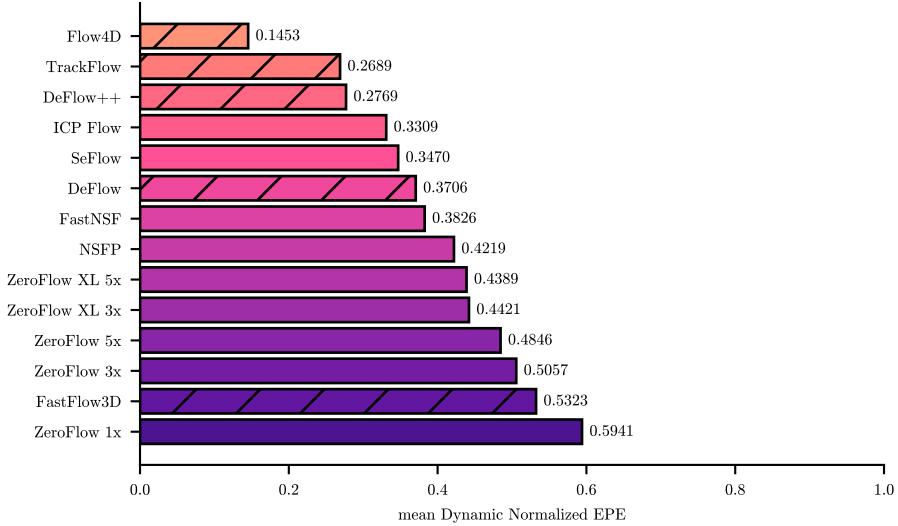


Fig. 9: mean Dynamic Normalized EPE of submissions to the *Argoverse 2 2024 Scene Flow Challenge* on Argoverse 2’s *test* split. Supervised methods shown with hatching. Lower is better.

Bucket Normalized EPE was the basis for the *Argoverse 2 2024 Scene Flow Challenge*⁴. The competition featured two tracks: a supervised track, and an unsupervised track, with TrackFlow serving as a baseline in the supervised track. Leaderboards for both tracks are ranked by minimum *mean Dynamic* component of Bucket Normalized EPE.

Notably, Flow4D [14] significantly improved over all prior supervised methods, halving the dynamic error of TrackFlow. Interestingly, Flow4D does not feature any class-aware loss features, instead focusing on architectural improvements over FastFlow3D [12]-based architectures (e.g. ZeroFlow [42], DeFlow [53]). Unsupervised scene flow methods also saw meaningful improvements; ICP-Flow [24] significantly outperformed FastNSF [20], the best performing unsupervised baseline, closely followed by SeFlow [54].

B Bucket Normalized EPE Structure

Table 4 show the structure of the class-speed matrix of Bucket Normalized EPE on Argoverse 2.

⁴ Full details about the competition can be found at <http://argoverse.org/sceneflow>

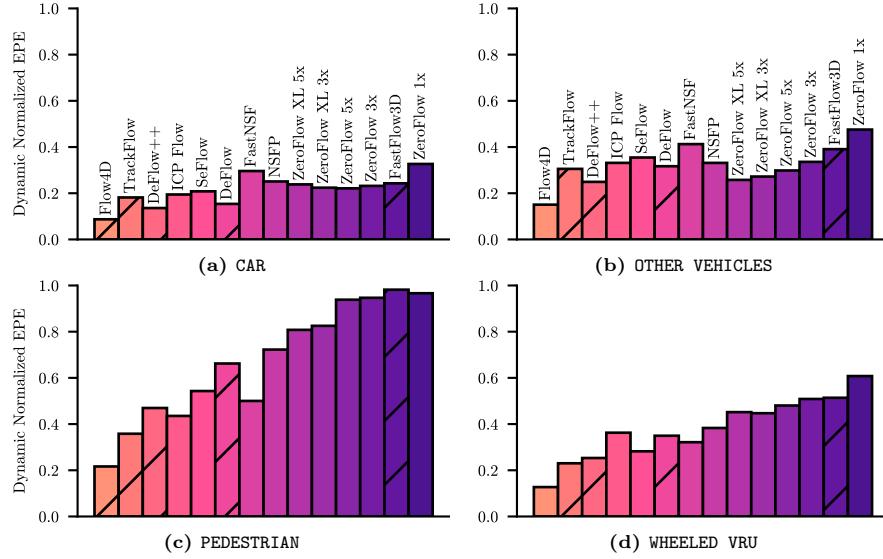


Fig. 10: Per meta-class Dynamic Normalized EPE of submissions to *the Argoverse 2 2024 Scene Flow Challenge* on Argoverse 2’s *test* split. Supervised methods shown with hatching. Lower is better. Method color and position is consistent between plots.

Class	Speed Columns				
	0-0.4m/s	0.4-0.8m/s	0.8-1.2m/s	...	20-∞m/s
BACKGROUND	-	-	-	-	-
CAR	-	-	-	-	-
OTHER VEHICLES	-	-	-	-	-
PEDESTRIAN	-	-	-	-	-
WHEELED VRU	-	-	-	-	-

Table 4: Example of Bucket Normalized EPE’s class-speed matrix.

C Bucket Normalized EPE Without Semantics

In Section 3 we present Bucket Normalized EPE with the object distribution broken down by semantic classes. While this makes sense when semantics are available, this is not a fundamental requirement for Bucket Normalized EPE. To demonstrate this, we break down Argoverse 2’s bounding boxes by *size* instead of semantics. We group the ground truth boxes into one of three volume based clusters: **SMALL**: $< 9.5m^3$, **MEDIUM**: $\geq 9.5m^3 \wedge < 40m^3$, or **LARGE**: $\geq 40m^3$. As shows in Fig. 11, this distribution breakdown still highlights the poor performance of prior art on small objects.

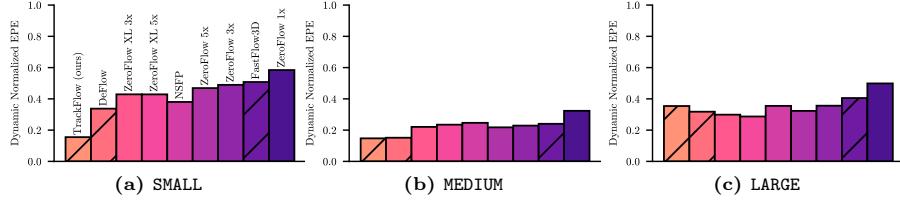


Fig. 11: Bucket Normalized EPE using ground truth size based clustering.

D FAQ

D.1 TrackFlow is *just* a tracking method

Yes, TrackFlow is a tracking method applied to the scene flow problem. The state-of-the-art performance of TrackFlow suggests that Scene Flow via Tracking is a fruitful area of exploration for future work on supervised scene flow.

D.2 TrackFlow uses bounding boxes and thus can only estimate rigid flow — what does this paper have to say about non-rigid scene flow?

It’s true that TrackFlow operates on the level of bounding boxes, but as we discuss in Section 2.1, public real-world datasets derive motion annotations from bounding box tracks. If non-rigid labels were available, one could train a detector to also regress keypoints (or use an off-the-shelf pretrained method [51]) and track across those keypoints.

D.3 TrackFlow uses bounding boxes from a detector — does this mean it cannot detect open-set objects?

TrackFlow uses a class-aware object detector as its bounding box proposer. However, the Scene Flow via Tracking framework does not require class annotations – nothing prevents the use of a class agnostic open world bounding box proposer, either trained like FasterRCNN’s RPN [8, 39], Object Localization Network [13], or via geometric priors [11].

D.4 Our metric is “just” Threeway EPE extended to multiple classes and multiple speed buckets with normalization, and our method “just” combines a detector and tracker. Where is the novelty in this idea?

The ideas presented in this paper are simple and post-hoc obvious, but serve to highlight catastrophic failures currently overlooked in existing approaches.