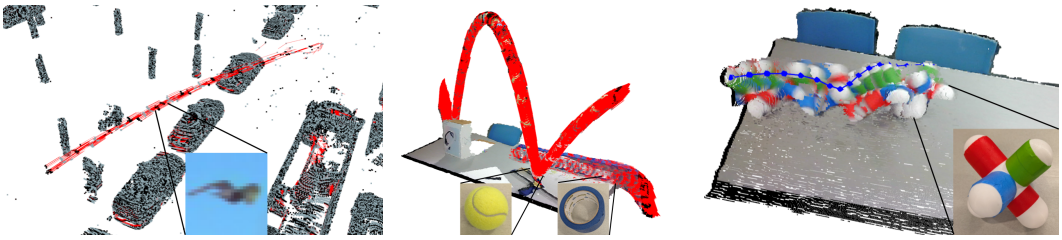
NEURAL EULERIAN SCENE FLOW FIELDS

Kyle Vedder[1,2*] **Neehar Peri**[2,3] **Ishan Khatri**[3] **Siyi Li**[1] **Eric Eaton**[1]
Mehmet Kocamaz[2] **Yue Wang**[2] **Zhiding Yu**[2] **Deva Ramanan**[3] **Joachim Pehserl**[2]
[1]University of Pennsylvania [2]NVIDIA [3]Carnegie Mellon University



(a) Small object motion extraction... (b) ...in diverse, dynamic scenes... (c) ...with emergent 3D point tracking behavior!

Figure 1: EulerFlow is able to capture the motion of small, fast moving objects with few lidar points, such a bird flying in front of an autonomous vehicle (Figure 1a). EulerFlow's flexibility allows it to estimate scene flow for fast-moving table top objects *without additional hyperparameter tuning* (Figure 1b). EulerFlow's ODE estimate exhibits emergent 3D point tracking behavior without explicit long-horizon supervision (Figure 1c). Note that point clouds are shown in color for visualization purposes only; RGB is not used during optimization.

ABSTRACT

We reframe scene flow as the task of estimating a continuous space-time ordinary differential equation (ODE) that describes motion for an entire observation sequence, represented with a neural prior. Our method, *EulerFlow*, optimizes this neural prior estimate against several multi-observation reconstruction objectives, enabling high quality scene flow estimation via self-supervision on real-world data. EulerFlow works out-of-the-box without tuning across multiple domains, including large-scale autonomous driving scenes and dynamic tabletop settings. Remarkably, EulerFlow produces high quality flow estimates on small, fast moving objects like birds and tennis balls, and exhibits emergent 3D point tracking behavior by solving its estimated ODE over long-time horizons. On the Argoverse 2 2024 Scene Flow Challenge, EulerFlow outperforms *all* prior art, surpassing the next-best *unsupervised* method by more than $2.5\times$, and even exceeding the next-best *supervised* method by over 10%. See `vedder.io/eulerflow` for interactive visuals.

1 INTRODUCTION

Scene flow estimation is the task of describing motion with per-point 3D motion vectors between temporally successive point clouds (Dewan et al., 2016; Liu et al., 2019; Erçelik et al., 2022; Jund et al., 2021; Zhang et al., 2024b; Vedder et al., 2024; Khatri et al., 2024). Such per-point motion estimates are critical for autonomy in diverse environments, e.g., maneuvering around open-world objects like debris (Peri et al., 2022a) or folding deformable cloth (Weng et al., 2022). Importantly, scene flow estimation requires not only an understanding of object *geometry*, but also its *motion*. However, scene flow methods broadly do not work on smaller objects (Khatri et al., 2024). For example, in the autonomous vehicles domain, Khatri et al. highlight that even supervised methods struggle to describe the majority of pedestrian motion, with unsupervised methods failing dramatically. Scene flow promises to be a powerful primitive for understanding the dynamic world, but such failures explain why it has limited adoption in downstream applications like tracking (Zhai et al., 2020) or open-world object extraction (Najibi et al., 2022).

*Corresponding email: `kvedder@seas.upenn.edu`

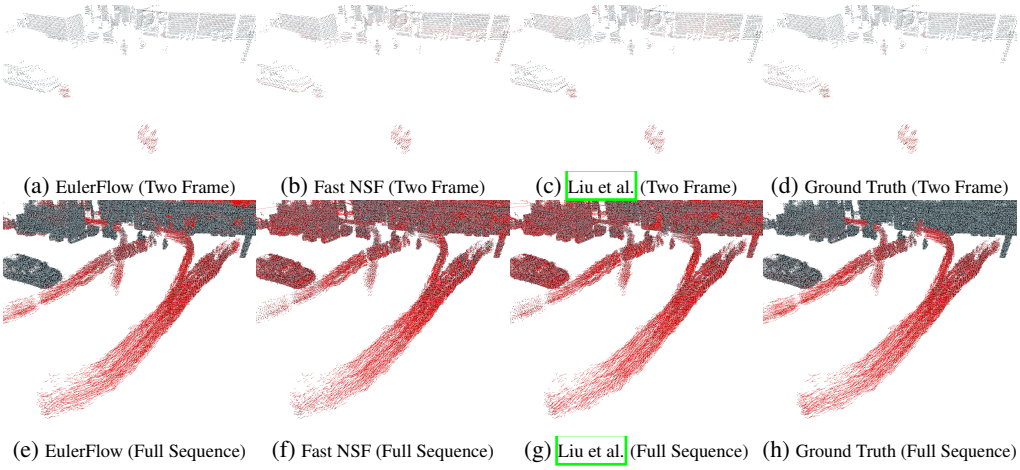


Figure 2: We visualize an example of five pedestrians crossing the street in front of a stopped car, cherrypicked to have unusually high density lidar returns, making it particularly easy to estimate flow. Figures 2a–2d depict a two-frame flow visualization of EulerFlow and several strong baselines. Notably, only visualizing flow over two frames makes it difficult to distinguish flow quality. In contrast, Figures 2e–2h depict flow vectors over the full sequence, making differences in quality clear; for example, EulerFlow is the only one without artifacts on the stopped car.

Scene Flow via ODE. In Figure 2, visual assessment of scene flow quality is much easier in an accumulated global frame; while incomplete due to an implicit time axis, these accumulated flow vectors allow viewers to imagine how positions in 3D space evolve over *many* timesteps, and compare that to predicted flows. This imagination of scene flow as continuous motion over large time intervals motivates us to model scene flow as an ordinary differential equation (ODE) that describes the scene's instantaneous motion across continuous position and time. Scene flow estimation then becomes the task of estimating this ODE. We can straightfowardly represent this ODE estimate with a neural prior (Li et al., 2021b) and optimize it against scene flow surrogate objectives, both over single frame pairs and extended *across arbitrary time intervals* to produce better quality estimates. We formalize this in Section 3 and propose the *Scene Flow via ODE* framework.

EulerFlow. We instantiate Scene Flow via ODE with standard point cloud distance objectives like Chamfer Distance to create *EulerFlow*. Notably, EulerFlow outperforms *all* prior art, supervised or unsupervised, on the Argoverse 2 2024 Scene Flow Challenge and Waymo Open Scene Flow benchmark. It outperforms prior *unsupervised* methods by a large margin ($> 2.5\times$ mean dynamic error reduction), and is able to capture small, fast moving objects, including those outside of labeled taxonomies (e.g. the flying bird in Figure 1a). Due to its simplicity, EulerFlow is able to provide high quality scene flow out-of-the-box on real-world data for other important domains such as dynamic tabletop settings (Figure 1b) *without* domain-specific tuning. Finally, we show that simple ODE solving techniques like Euler integration can be used to extract 3D point tracks (Figure 1c), which serves as both an exciting emergent behavior as well as a mechanism for visualizing and interpreting the quality of the continuous ODE estimate.

We present four primary contributions:

- We propose *Scene Flow via ODE* (SFvODE), a reframing of scene flow estimation as the task of fitting an ODE over all observations to describe the change of continuous positions over continuous time, unlocking a new class of surrogate objectives that enable better scene flow estimates.
- We instantiate SFvODE with *EulerFlow*, a flexible **unsupervised** scene flow method that achieves **state-of-the-art** performance on the Argoverse 2 2024 Scene Flow Challenge, **beating all prior supervised and unsupervised methods**.
- We study EulerFlow and show its strong performance is derived from its ability to optimize its ODE estimate against the full sequence of observations over arbitrary time horizons.
- We show that EulerFlow's simple, flexible formulation allows it to run unmodified on a variety of domains, with emergent capabilities like 3D point tracking behavior.

2 Background and Related Work

Evaluation. Dewan et al. formalized scene flow for point clouds as the task of estimating motion between point cloud P_t at time t and point cloud P_{t+1} at $t+1$ by estimating the true flow $\mathcal{F}_{t,t+1}$, i.e. true residual vectors for every point in P_t that describe its movement to its associated position at $t+1$. Error is computed by measuring the per-point endpoint distance between estimated and ground truth vectors. Historically, these errors are reported with a per-point average (*Average EPE*); however, as Chodosh et al. show, Average EPE is dominated by background points, preventing meaningful measurement of non-ego object motion descriptions. Khatri et al. address this shortcoming with *Bucket Normalized EPE*, which reports per-class performance normalized by speed, allowing for direct comparisons across classes with very different average speeds (e.g. pedestrians and cars). Bucket Normalized EPE is the basis for the *Argoverse 2024 Scene Flow Challenge*[1], where methods are ranked by the mean error of their motion descriptions (*mean Dynamic Normalized EPE*).

Input / Output Formulation. Dewan et al.'s choice to formulate scene flow using *only* two input frames is arbitrary; it's the minimal information needed to extract rigid motion, but there are not real-world problems constrained to *only* have access to two frames. Indeed, using five or ten frames of past observations is standard practice in the 3D detection literature (Zhu et al., 2019; Vedder & Eaton, 2022; Peri et al., 2022b; 2023; Nalty et al., 2022), and multi-frame formulations have started to appear in the scene flow literature: Liu et al. (2024) and Flow4D (Kim et al., 2024) use three (P_{t-1}, P_t, P_{t+1}) and five input frames $(P_{t-3}, \ldots, P_{t+1})$ respectively to predict $\hat{\mathcal{F}}_{t,t+1}$. As we discuss in Section 3, rather than just using more observations to estimate flow for a single frame pair, we formulate scene flow as a joint estimation problem: given the full observation sequence $(P_0, \ldots, P_N)$, we estimate *all* flows $\hat{\mathcal{F}}_{0,1}, \ldots, \hat{\mathcal{F}}_{N-1,N}$ between *all* adjacent observations.

Feedforward Methods. Feedforward networks are a popular class of scene flow methods due to their fast inference speed (Liu et al., 2019; Behl et al., 2019; Tishchenko et al., 2020; Kittenplon et al., 2021; Wu et al., 2020; Puy et al., 2020; Li et al., 2021a; Jund et al., 2021; Gu et al., 2019; Battrawy et al., 2022; Wang et al., 2022b; Kim et al., 2024; Zhang et al., 2024a). While they are often trained with supervised labels, recent work has developed distillation pipelines that leverage unsupervised pseudolabelers (Vedder et al., 2024; Zhang et al., 2024b; Lin & Caesar, 2024).

Neural Scene Flow Prior. Li et al. (2021b) propose Neural Scene Flow Prior (NSFP), a widely adopted unsupervised scene flow approach. NSFP uses the inductive bias of the smooth, restricted learnable function class of two ReLU MLP coordinate networks (8 hidden layers of 128 neurons); θ to estimate forward flow and θ' to estimate backwards flow, minimizing

$$\text{TruncatedChamfer}(P_t + \theta(P_t), P_{t+1}) + \left\| P_t + \theta(P_t) + \theta'(P_t + \theta(P_t)) - P_t \right\|_2 \quad , \tag{1}$$

where TruncatedChamfer is defined as the standard L_2 Chamfer distance, but with per-point distances above 2 meters set to zero in order to reduce the influence of outliers. NSFP is optimized for at most 1000 steps with early stopping.

Motion Beyond Two Frames. Wang et al. (2022a) tackles the adjacent problem of estimating 3D point *trajectories* over 25 frames with Neural Trajectory Prior (NTP) by jointly optimizing three separate ReLU MLP neural priors: 1) a sinusoidal embedded, time conditioned, 25 frame trajectory basis estimator (embed$(t) \mapsto 256 \times 25 \times 3$ tensor, where 256 is the dimension of the trajectory basis), 2) a coordinate network bottleneck encoder, and 3) a bottleneck decoder to estimate a per-point linear combination over the learned trajectories. Trajectories are optimized for both a one-frame lookahead L_2 Chamfer loss and a cyclic consistency loss over the entire velocity space trajectory.

Deformation in Reconstruction. Nerfies (Park et al., 2021) and DynamicFusion (Newcombe et al., 2015) estimate a deformation field to warp a canonical frame to explain the observed frame. While capable of describing small motions, these methods require a canonical frame that contains all of the relevant geometry to deform; however, in large, highly dynamic scenes like autonomous driving, there is often no frame that contains all moving objects. By comparison, Scene Flow via ODE does not assume the existence of a canonical frame, instead only describing how the scene changes.

[1] https://www.argoverse.org/sceneflow

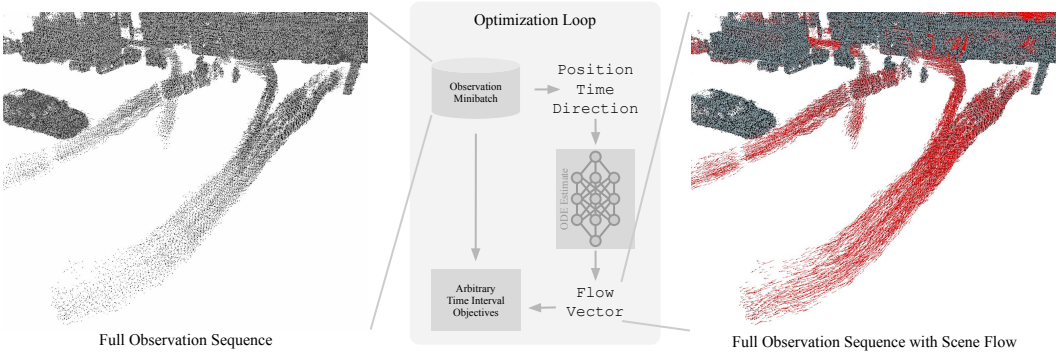


Figure 3: Overview of our *Scene Flow via ODE* framework, which estimates an ODE across the entire observation sequence by optimizing against multi-frame objectives. This ODE estimate is represented with a neural prior (Li et al., 2021b), providing a flexible, general representation for describing position-time motion.

3 Scene Flow via ODE

Prior art consumes multiple frames $(P_{t-N}, \ldots, P_{t+1})$ as input, but these methods are ultimately only tasked with estimating flow vectors between P_t and P_{t+1}. We instead pose the problem of estimating a time-conditioned flow field that describes motion for *all* adjacent point clouds P_t, P_{t+1} in the entire sequence $(P_0, \ldots, P_N)$. To do this, rather than describing scene flow as positional change over a fixed interval ($\mathcal{F}_{t,t+1}$ are residual vectors over the interval t to $t+1$) as we did in Section 2, we can instead express these changes as a differential equation that describes positional change over *continuous* time.

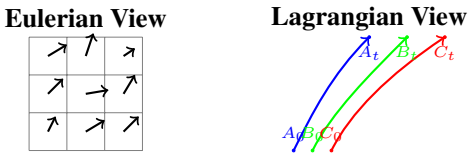


Figure 4: Comparison of Eulerian and Lagrangian descriptions of 2D flow. An Eulerian view characterizes a flow field via instantaneous velocities at many different points, while a Lagrangian view characterizes a flow field via trajectories of many different particles across time. Both approaches are valid ways of describing an underlying flow field, and with sufficient characterization one view can be readily converted to another, but the Lagrangian view relies on a definition of the definition of consistent canonical frame.

Formally, given a scene, let $L(x_0, y_0, z_0, t)$ be the Lagrangian view of the scene's true flow field, i.e. a continuous function that, given a canonical particle (x_0, y_0, z_0) in a canonical frame 0, describes its (x, y, z) position at frame t. As we discuss in Section 2, this Lagrangian view is common in the the deformable reconstruction literature, and the requirement for a canonical frame means such approaches struggle to describe scenes where there is no frame that contains all moving objects.

To break this canonical frame dependence, we choose to take an Eulerian view of motion, i.e. $F(x, y, z, t) = (\frac{\partial L_x}{\partial t}, \frac{\partial L_y}{\partial t}, \frac{\partial L_z}{\partial t})$, which describes the velocity at position (x, y, z) at time t. As we show in our derivation in Appendix C.1, this formulation does not require an external canonical frame to estimate a point's trajectory from t to t'; instead, we can simply set the initial conditions of the ODE at t to x_t, y_t, z_t and utilize an off-the-shelf ODE solver (e.g. Euler integration) to extract flow from t to t', expressed as $E(x_t, y_t, z_t, t, t')$.

We do not know the true flow field F when estimating scene flow; however, we can represent F with a neural prior θ ($F \approx \theta$), and optimize θ against surrogate objectives. This framing, which we formalize into the *Scene Flow via ODE* framework (SFvODE; Figure 3), allows θ to benefit from constructive interference between objectives, as well as enables us to formulate objectives over arbitrarily long time horizons, unlocking high quality estimates.

4 EULERFLOW

Scene Flow via ODE proposes a framework where the neural prior θ represents an estimate of the Eulerian flow field F (i.e. $F \approx \theta$); however, it does not prescribe the optimization objectives for θ. Thus, we instantiate Scene Flow via ODE with *EulerFlow*, a point cloud only scene flow method[2] with reconstruction and cyclic consistency objectives across the entire sequence of observations.

As we show in Equation 17 (Appendix C.4), we can use θ's Eulerian flow field estimate to extract an estimated point trajectory from x_t, y_t, z_t at t to some future location at time t' via Euler integration over θ without requiring a canonical frame definition, i.e. $E_\theta(x_t, y_t, z_t, t, t')$. By extracting point trajectories for every point p in P_t using E_θ, we can not only construct a two-frame scene flow estimate of $\mathcal{F}_{t,t+1}$, but also estimate flow to arbitrary future or prior timesteps (e.g. $\mathcal{F}_{t,t+2}$ or $\mathcal{F}_{t,t-1}$). This allows us to optimize over multi-frame reconstruction objectives: we can now pose reconstruction surrogate objectives between *any* two point clouds in our observation sequence, not just adjacent point clouds P_t and P_{t+1}. Similarly, we can straightforwardly pose cyclic consistency objectives by composing $\mathcal{F}_{t,t+1}$ and $\mathcal{F}_{t+1,t}$. Formally, for P_t's $\mathcal{F}_{t,t+k}$ (for any $k \in \mathbb{Z}$), we define

$$\text{Euler}_\theta\left(P_t, k\right) = P_t + \mathcal{F}_{t,t+k} = \forall p \in P_t : E_\theta(p_x, p_y, p_z, t, t+k) \ , \tag{2}$$

enabling us to pose θ's optimization objective $\forall P_t \in (P_0, \ldots, P_N)$ across the window of size W

$$\arg\min_\theta \sum \begin{array}{l} \forall k \in \{-W, \ldots, W\} \setminus \{0\} : \text{TruncatedChamfer}(\text{Euler}_\theta\left(P_t, k\right), P_{t+k}) \\ \alpha \left\|\text{Euler}_\theta\left(\text{Euler}_\theta\left(P_t, 1\right), -1\right) - P_t\right\|_2 \end{array} \tag{3}$$

In practice, we set W to 3 and α to 0.01. We provide additional implementation details in Appendix C. In order to optimize θ, our estimate of the Eulerian flow field F, we perform Euler integration to extract point cloud flow estimates as part of reconstruction losses. Notably, EulerFlow only requires a single optimization loop over a single neural prior θ compared to NSFP's two priors θ and θ'. Our neural prior θ is a straightforward extension to NSFP's coordinate network prior. Like with NSFP, TruncatedChamfer is defined as the standard L_2 Chamfer distance with per-point distances below 2 meters. As we show in Section 5, EulerFlow's simple ODE estimation formulation across multiple observations produces high quality flow, and solving this ODE over arbitrary time spans unlocks emergent point tracking behavior.

5 EXPERIMENTS

In order to validate EulerFlow's construction and better understand the impact of its design choices, we perform extensive experiments on the Argoverse 2 (Wilson et al., 2021) and Waymo Open (Sun et al., 2020) autonomous vehicle datasets. We compare against open source implementations of FastNSF (Li et al., 2023), Liu et al., NSFP (Li et al., 2021b), FastFlow3D (Jund et al., 2021), and variants of ZeroFlow (Vedder et al., 2024) provided by the ZeroFlow model zoo[3], a third-party implementation of NTP (Wang et al., 2022a) from Vidanapathirana et al., and Argoverse 2 2024 Scene Flow Challenge leaderboard submission results from the authors of Flow4D (Kim et al., 2024), TrackFlow (Khatri et al., 2024), DeFlow++/DeFlow (Zhang et al., 2024a), ICP Flow (Lin & Caesar, 2024), and SeFlow (Zhang et al., 2024b). As discussed in Khatri et al. and used in the Argoverse 2 2024 Scene Flow Challenge, methods are ranked by their speed normalized *mean Dynamic Normalized EPE*.

Implementation Details. To showcase the flexibility of EulerFlow without hyperparameter tuning, for all quantitative experiments we run with a neural prior of depth 8 (NSFP's default depth), except for our submission to the Argoverse 2 2024 Scene Flow Challenge (Section 5.1) where, based on our depth ablation study on the val split (Section 5.2.3), we set the depth of the neural prior to 18. As discussed in NTP's original paper (Wang et al., 2022a) and confirmed by our experiments, NTP struggles to converge beyond 25 frames, so we only compare against it in a 20 frame settings. As is

[2]Visualizations shown in color for better viewing. EulerFlow can also use monocular depth estimates (Appendix A.2)

[3]https://github.com/kylevedder/SceneFlowZoo, from Vedder et al. (2024).

typical in the scene flow literature (Chodosh et al., 2023), we perform ego compensation and ground point removal on both Argoverse 2 and Waymo Open using the dataset provided map and ego pose.

5.1 How does EulerFlow compare to prior art on real data?

EulerFlow achieves **state-of-the-art** performance on the *Argoverse 2 2024 Scene Flow Challenge* leaderboard. Despite being unsupervised, EulerFlow **surpasses *all* prior art, supervised or unsupervised**, including Flow4D (Kim et al., 2024)[4] and ICP Flow (Lin & Caesar, 2024)[5]. Notably, EulerFlow achieves $< 2.5\times$ lower error mean Dynamic EPE than ICP Flow and beats Flow4D by over 10%.

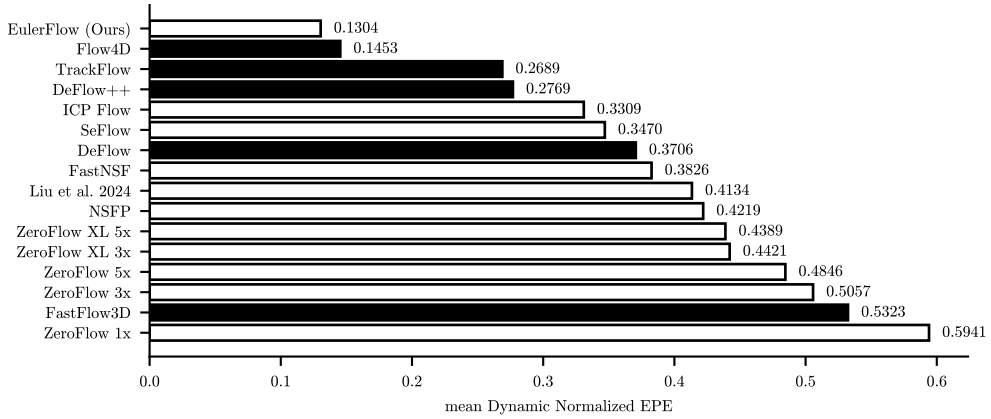


Figure 5: Mean Dynamic Normalized EPE of EulerFlow compared to prior art on the Argoverse 2 2024 Scene Flow Challenge test set. EulerFlow is state-of-the-art, beating all supervised (shown in black) and unsupervised (shown in white) methods. Lower is better.

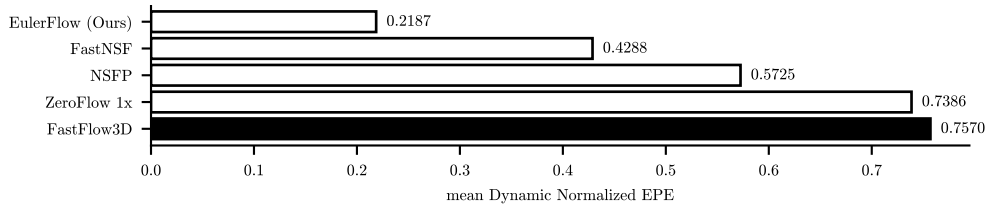


Figure 6: Mean Dynamic Normalized EPE of EulerFlow compared to prior art on the Waymo Open validation set. EulerFlow is state-of-the-art, beating all supervised (shown in black) and unsupervised (shown in white) methods. Lower is better.

EulerFlow's dominant performance also holds on Waymo Open (Sun et al., 2020); we compare against several popular methods (Figure 6), and EulerFlow again out-performs the baselines by a wide margin, more than halving the error over the next best method.

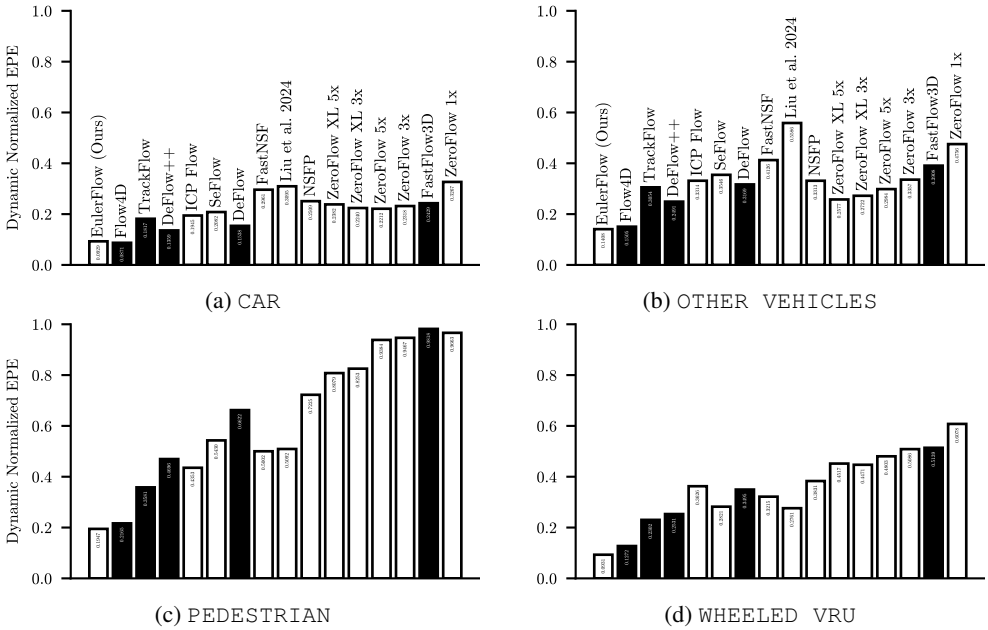
5.2 What contributes to EulerFlow's state-of-the-art performance?

We find that EulerFlow's lower mean Dynamic EPE can be attributed to better performance on smaller objects. On Argoverse 2, compared to Flow4D, EulerFlow's improves on `WHEELED VRU` (Figure 7d), a small, rare, fast moving class. Compared to ICP Flow, EulerFlow's improves on all classes (at least halving the error on every class!), with the largest improvements coming from the smaller and harder to detect objects `PEDESTRAIN` and `WHEELED VRU` (Figures 7c–7d). On Waymo Open, the same story holds; the most dramatic performance improvements come from the small object classes of `CYCLIST` and `PEDESTRIAN` (Figure 8).

These results are consistent with our qualitative visualizations. Figure 13 shows EulerFlow is able to cleanly extract the motion of a bird flying past the ego vehicle. Euler integration using EulerFlow's

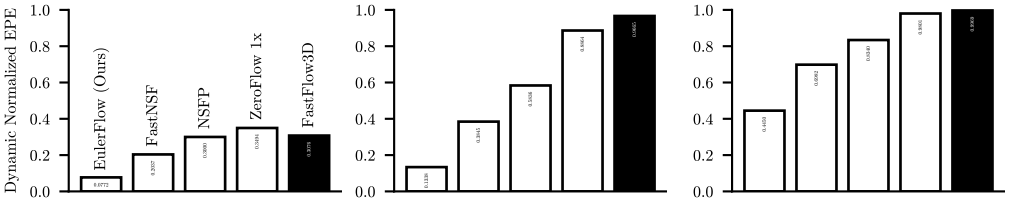
[4]Flow4D is the winner of the 2024 Argoverse 2 Scene Flow Challenge supervised track.

[5]ICP Flow is the winner of the 2024 Argoverse 2 Scene Flow Challenge unsupervised track.



(a) CAR (b) OTHER VEHICLES (c) PEDESTRIAN (d) WHEELED VRU

Figure 7: Per class Dynamic Normalized EPE of EulerFlow compared to prior art on the Argoverse 2 2024 Scene Flow Challenge test set. Supervised methods shown in black, unsupervised methods shown in white. Methods are ordered left to right by increasing mean Dynamic Normalized EPE. Lower is better.



(a) VEHICLE (b) CYCLIST (c) PEDESTRIAN

Figure 8: Per class Dynamic Normalized EPE of EulerFlow compared to prior art on the Waymo Open validation set. Supervised methods shown in black, unsupervised methods shown in white. Methods are ordered left to right by increasing mean Dynamic Normalized EPE. Lower is better.

ODE, starting at the bird's takeoff position and ending when it loses lidar returns, produces emergent 3D point tracking behavior on the bird through its trajectory (Figure 9), further demonstrating the quality of EulerFlow's model of the scene's motion.

(a) Bird trajectory via Euler integration from takeoff (b) Bird being tracked

Figure 9: EulerFlow is able to track the bird over 20 frames by performing Euler integration starting from takeoff until it loses all point cloud lidar returns.

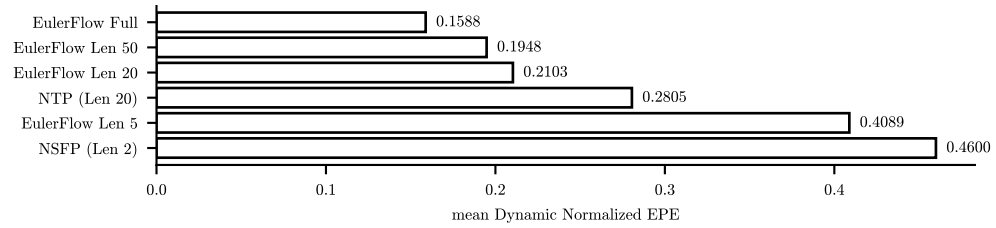


Figure 10: Mean Dynamic Normalized EPE of EulerFlow for various sequence lengths on the Argoverse 2 val split, compared against representative baselines. These results demonstrate that EulerFlow improves with sequence length; however, at a sequence length of 20, our method significantly outperforms NTP, suggesting that EulerFlow's performance cannot solely be attributed to longer sequence modeling.

5.2.1 How does observation sequence length impact EulerFlow?

As we discuss in Section 3, EulerFlow benefits from constructive interference from ODE estimation over many observations. Does this sufficiently explain EulerFlow's performance? Figure 10 shows the performance of EulerFlow at length 5, 20, 50, and full sequence (roughly 160 frames) compared to NSFP and NTP at length 20. EulerFlow sees clear continual improvements as the number of frames increases without signs of saturation. However, sequence length alone does not explain EulerFlow's performance; even at the same sequence length of 20, EulerFlow demonstrates significantly better performance than NTP.

5.2.2 How do multi-frame optimization objectives impact EulerFlow?

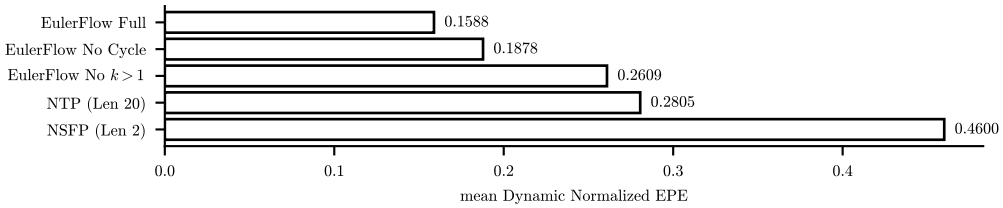


Figure 11: Mean Dynamic Normalized EPE of EulerFlow for various losses on the Argoverse 2 val split, compared against representative baselines. These results demonstrate that EulerFlow's multi-observation optimization objectives significantly improve overall performance.

Equation 3 outlines two major components of EulerFlow's loss: multi-frame Euler integration for Chamfer Distance reconstruction, and cycle consistency. Figure 11 compares EulerFlow without more than one integration step (No $k > 1$) and without cycle consistency rollouts (No Cycle) to better understand the impact of these components. Ablating multi-step Euler integrated rollouts results in significant degredation, as they are a strong forcing function to have consistent, smooth flow volumes; indeed, despite consuming the entire sequence, EulerFlow (No $k > 1$) is only slightly better than NTP with a sequence length of 20. These results highlight the power of multi-step rollouts and their potential as a objective for other test-time optimization methods and feedforward methods.

5.2.3 How does the capacity of the neural prior impact EulerFlow?

Li et al. ablate the capacity of NSFP's neural prior to characterize underfitting and overfitting to optimization objective noise, ultimately selecting a depth of 8. EulerFlow's neural prior is structured similarly; however, NSFP is fitting a single snapshot in time, while EulerFlow is fitting an entire ODE over significant time intervals. Intuitively, one would expect that full sequence modeling would benefit from greater network capacity.

To evaluate this, we perform a sweep of EulerFlow's network depth on the Argoverse 2 validation split (Figure 12). While EulerFlow with NSFP's default of depth 8 performs well on our Argoverse 2 evaluations (0.1% worse than the supervised state-of-the-art Flow4D), we see that performance improves as the neural prior's depth increases until depth 18 (indicating underfitting), where we start

to see degradation (indicating overfitting to noise). Based on these results our Argoverse 2 2024 Scene Flow Challenge leaderboard submission uses a depth 18 neural prior (Figure 5).

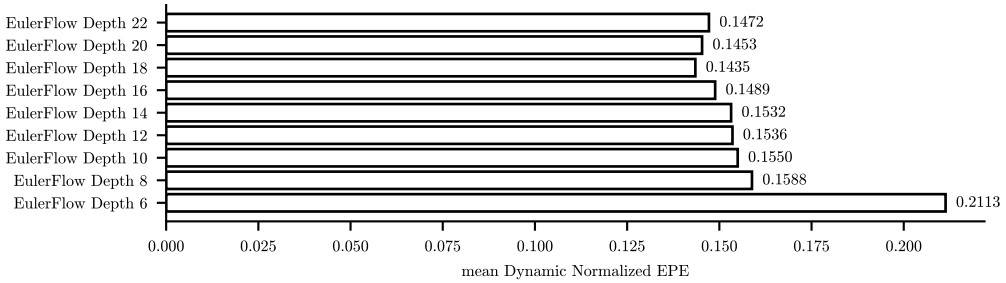


Figure 12: Mean Dynamic Normalized EPE of EulerFlow on the Argoverse 2 val split for different neural prior capacities. Shallow networks underfit the ODE, while deeper networks overfit to noise in the optimization objectives.

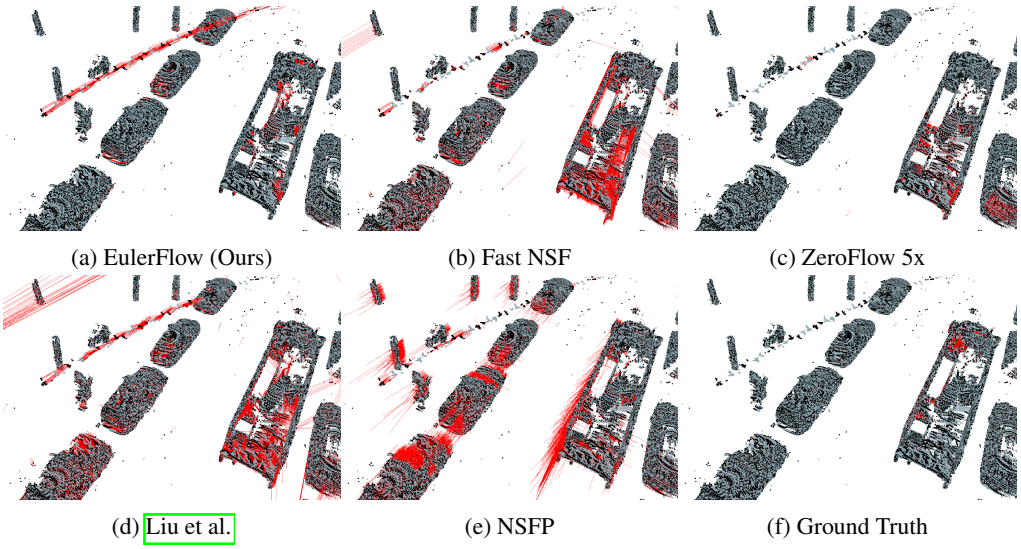


Figure 13: Visualization of EulerFlow compared to prior art for the same scene as Figure 1a and Figure 9a. EulerFlow is able to extract the bird's trajectory; however, all other methods except Liu et al. fail to recognize this motion, and Liu et al.'s flow is marred by severe scene artifacts. The bird is outside the labeled object taxonomy, and so its motion is unlabeled in the ground truth (Figure 13f).

5.3 Beyond Autonomous Vehicles

Due to a dearth of real-world, labeled scene flow data, prior scene flow work on real data overwhelmingly evaluates on autonomous vehicle datasets (Dewan et al., 2016; Li et al., 2021b; Jund et al., 2021; Li et al., 2023; Chodosh et al., 2023; Liu et al., 2024; Vedder et al., 2024; Khatri et al., 2024); consequently, motion understanding in other important domains like tabletop manipulation has been neglected. To showcase EulerFlow's out-of-the-box flexibility and generalizability, we visualize EulerFlow on several dynamic tabletop scenes we collected using the ORBBEC Astra, a low cost depth camera commonly used in robotics (Figure 14). For viewing ease, we paint our point clouds with color; however, RGB information is not provided to EulerFlow during optimization. While EulerFlow only reasons about point clouds, it can leverage video mono depth estimates to describe RGB-only scene flow (Appendix A.2). Interactive visuals are available at `vedder.io/eulerflow`.

6 Conclusion

By reframing scene flow as fitting an ODE over positions for a full sequence of observations, we are able to construct EulerFlow, a simple unsupervised scene flow method that achieves state-of-the-art performance on the Argoverse 2 2024 Scene Flow Challenge and Waymo Scene Flow benchmark,

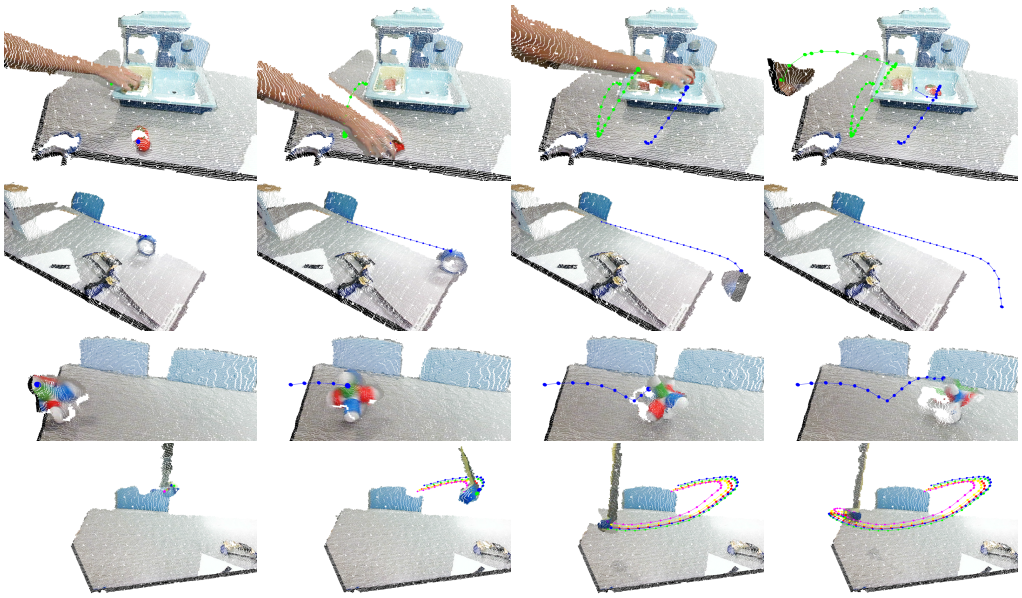


Figure 14: Visualizations of EulerFlow's emergent 3D point tracking behavior that demonstrate the quality of its ODE estimate. Row 1 depicts tracking a tomato placed in the sink by a human hand; note the point does not move despite the hand grasping the tomato. Row 2 depicts tracking of painters tape rolling off a table; EulerFlow is able to estimate its trajectory even after it disappears out of frame. Row 3 depicts tracking of the motion of a jack commonly used in tabletop manipulation experiments (Venkatesh et al., 2023). Row 4 depicts tracking of a tennis ball taped to a flexible rod. All tracks are produced by Euler integration through the estimated ODE from the initial conditions shown in the left column. Note that point clouds are shown in color for visualization purposes only.

where it beats all prior art, supervised or unsupervised. EulerFlow is able to describe motion on small, fast moving, out of distribution objects unable to be captured by prior art, suggesting that it makes good on the promises of scene flow as a powerful primitive for understanding the dynamic world. It also exhibits other emergent capabilities, like basic 3D point tracking behavior.

We believe that this ODE formulation has implications for scene flow at large, including beyond test-time optimization methods; the power of multi-step Euler integration may translate to feedforward network training. Future work should explore feedforward models that perform autoregressive rollouts or directly learn to estimate multiple steps into the future.

6.1 Limitations and Future Work

EulerFlow's strong performance opens the book on an exciting new line of work; however, we feel that it's important to be candid about EulerFlow's current limitations in order to make future progress.

EulerFlow is point cloud only. Point cloud sparsity bottlenecks performance; for instance, in Figure 9 and Figure 13 we were only able to track the bird for 20 frames because we lost lidar observations of the bird, while it remained visible in the car's RGB cameras. Future works should explore multi-modal fusion for better long-term motion descriptions.

EulerFlow is expensive to optimize. With our implementation, optimizing EulerFlow for a single Argoverse 2 sequence takes 24 hours on one NVIDIA V100 16GB GPU, putting it on par with the original NeRF paper's computation expense (Mildenhall et al., 2021). However, like with NeRF, we believe algorithmic, optimization, and engineering improvements can significantly reduce runtime.

EulerFlow does not understand ray casting geometry. During ego-motion, a static foreground occluding object casts a moving shadow on the background; this causes Chamfer Distance to estimate this as a leading edge of moving structure, encouraging false motion artifacts (Li et al., 2021b). This can be addressed with optimization losses that model point clouds as originating from a time of flight sensor with limited visibility, as has been successfully demonstrated in the reconstruction (Chodosh et al., 2024) and forecasting literature (Khurana et al., 2023; Agro et al., 2024), rather than an unstructured set of points to be associated via local point distance.

REFERENCES

Ben Agro, Quin Sykora, Sergio Casas, Thomas Gilles, and Raquel Urtasun. UnO: Unsupervised Occupancy Fields for Perception and Forecasting. In *CVPR*, 2024.

Ramy Battrawy, René Schuster, Mohammad-Ali Nikouei Mahani, and Didier Stricker. 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.

Aseem Behl, Despoina Paschalidou, Simon Donné, and Andreas Geiger. Pointflownet: Learning representations for rigid motion estimation from point clouds. In *Int. Conf. Comput. Vis.*, pp. 7962–7971, 2019.

D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.) (ed.), *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pp. 611–625. Springer-Verlag, October 2012.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NeurIPS'18, pp. 6572–6583, Red Hook, NY, USA, 2018.

Shin-Fang Chng, Sameera Ramasinghe, Jamie Sherrah, and Simon Lucey. Gaussian Activated Neural Radiance Fields for High Fidelity Reconstruction and Pose Estimation. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*, pp. 264–280, Berlin, Heidelberg, 2022. Springer-Verlag. ISBN 978-3-031-19826-7.

Nathaniel Chodosh, Deva Ramanan, and Simon Lucey. Re-Evaluating LiDAR Scene Flow for Autonomous Driving. *arXiv preprint*, 2023.

Nathaniel Chodosh, Anish Madan, Deva Ramanan, and Simon Lucey. Simultaneous Map and Object Reconstruction, 2024. URL https://arxiv.org/abs/2406.13896.

Ayush Dewan, Tim Caselitz, Gian Diego Tipaldi, and Wolfram Burgard. Rigid scene flow for 3d lidar scans. In *Int. Conf. Intel. Rob. Sys.*, pp. 1765–1770. IEEE, 2016.

Emeç Erçelik, Ekim Yurtsever, Mingyu Liu, Zhijie Yang, Hanzhen Zhang, Pınar Topçam, Maximilian Listl, Yılmaz Kaan Çaylı, and Alois Knoll. 3D Object Detection with a Self-supervised Lidar Scene Flow Backbone. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 247–265, Cham, 2022. Springer Nature Switzerland.

Thomas Grandits, Alexander Effland, Thomas Pock, Rolf Krause, Gernot Plank, and Simone Pezzuto. GEASI: Geodesic-based earliest activation sites identification in cardiac models. *International Journal for Numerical Methods in Biomedical Engineering*, 37(8):e3505, 2021. ISSN 2040-7947. doi: 10.1002/cnm.3505. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cnm.3505.

Xiuye Gu, Yijie Wang, Chongruo Wu, Yong Jae Lee, and Panqu Wang. 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.

Ramin Hasani, Mathias Lechner, Alexander Amini, Daniela Rus, and Radu Grosu. Liquid Time-constant Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35:7657–7666, May 2021.

Wenbo Hu, Xiangjun Gao, Xiaoyu Li, Sijie Zhao, Xiaodong Cun, Yong Zhang, Long Quan, and Ying Shan. DepthCrafter: Generating Consistent Long Depth Sequences for Open-world Videos. *arXiv preprint arXiv:2409.02095*, 2024.

Philipp Jund, Chris Sweeney, Nichola Abdo, Zhifeng Chen, and Jonathon Shlens. Scalable Scene Flow From Point Clouds in the Real World. *IEEE Robotics and Automation Letters*, 12 2021.

Ishan Khatri, Kyle Vedder, Neehar Peri, Deva Ramanan, and James Hays. I Can't Believe It's Not Scene Flow! In *European Conference on Computer Vision (ECCV)*, 2024.

Tarasha Khurana, Peiyun Hu, David Held, and Deva Ramanan. Point Cloud Forecasting as a Proxy for 4D Occupancy Forecasting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Jaeyeul Kim, Jungwan Woo, Ukcheol Shin, Jean Oh, and Sunghoon Im. Flow4D: Leveraging 4D Voxel Network for LiDAR Scene Flow Estimation, 2024. URL https://arxiv.org/abs/2407.07995.

Yair Kittenplon, Yonina C Eldar, and Dan Raviv. Flowstep3d: Model unrolling for self-supervised scene flow estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 4114–4123, 2021.

Ruibo Li, Guosheng Lin, Tong He, Fayao Liu, and Chunhua Shen. 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, 2021a.

Xueqian Li, Jhony Kaesemodel Pontes, and Simon Lucey. Neural Scene Flow Prior. *Advances in Neural Information Processing Systems*, 34, 2021b.

Xueqian Li, Jianqiao Zheng, Francesco Ferroni, Jhony Kaesemodel Pontes, and Simon Lucey. Fast Neural Scene Flow. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9878–9890, October 2023.

Yancong Lin and Holger Caesar. ICP-Flow: LiDAR Scene Flow Estimation with ICP. 2024.

Dongrui Liu, Daqi Liu, Xueqian Li, Sihao Lin, Hongwei xie, Bing Wang, Xiaojun Chang, and Lei Chu. Self-supervised multi-frame neural scene flow, 2024. URL https://arxiv.org/abs/2403.16116.

Xingyu Liu, Charles R Qi, and Leonidas J Guibas. FlowNet3D: Learning Scene Flow in 3D Point Clouds. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. 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.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: representing scenes as neural radiance fields for view synthesis. *Commun. ACM*, 65(1):99–106, dec 2021. ISSN 0001-0782.

Mahyar Najibi, Jingwei Ji, Yin Zhou, Charles R. Qi, Xinchen Yan, Scott Ettinger, and Dragomir Anguelov. Motion Inspired Unsupervised Perception and Prediction in Autonomous Driving. European Conference on Computer Vision (ECCV), 2022.

Chrisopher Nalty, Neehar Peri, Joshua Gleason, Carlos Castillo, Shuowen Hu, Thirimachos Bourlai, and Rama Chellappa. A Brief Survey on Person Recognition at a Distance. 12 2022. doi: 10.48550/arXiv.2212.08969.

Richard A. Newcombe, Dieter Fox, and Steven M. Seitz. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 343–352, 2015. doi: 10.1109/CVPR.2015.7298631.

Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable Neural Radiance Fields. *ICCV*, 2021.

Neehar Peri, Achal Dave, Deva Ramanan, and Shu Kong. Towards Long Tailed 3D Detection. *CoRL*, 2022a.

Neehar Peri, Jonathon Luiten, Mengtian Li, Aljosa Osep, Laura Leal-Taixe, and Deva Ramanan. Forecasting from LiDAR via Future Object Detection. *arXiv:2203.16297*, 2022b.

Neehar Peri, Mengtian Li, Benjamin Wilson, Yu-Xiong Wang, James Hays, and Deva Ramanan. An empirical analysis of range for 3d object detection. *arXiv preprint arXiv:2308.04054*, 2023.

Gilles Puy, Alexandre Boulch, and Renaud Marlet. Flot: Scene flow on point clouds guided by optimal transport. In *Eur. Conf. Comput. Vis.*, pp. 527–544. Springer, 2020.

Sameera Ramasinghe, Hemanth Saratchandran, Violetta Shevchenko, Alexander Long, and Simon Lucey. On the Optimality of Activations in Implicit Neural Representations, 2024. URL https://openreview.net/forum?id=0Lqyut1y7M.

Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3D Deep Learning with PyTorch3D. *arXiv:2007.08501*, 2020.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. 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.

Ivan Tishchenko, Sandro Lombardi, Martin R Oswald, and Marc Pollefeys. Self-supervised learning of non-rigid residual flow and ego-motion. In *Int. Conf. 3D Vis.*, pp. 150–159. IEEE, 2020.

Kyle Vedder and Eric Eaton. Sparse PointPillars: Maintaining and Exploiting Input Sparsity to Improve Runtime on Embedded Systems. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2022.

Kyle Vedder, Neehar Peri, Nathaniel Chodosh, Ishan Khatri, Eric Eaton, Dinesh Jayaraman, Yang Liu, Deva Ramanan, and James Hays. ZeroFlow: Scalable Scene Flow via Distillation. In *Twelfth International Conference on Learning Representations (ICLR)*, 2024.

Sharanya Venkatesh, Bibit Bianchini, Alp Aydinoglu, and Michael Posa. Sampling-Based Model Predictive Control for Contact-Rich Manipulation. In *IROS 2023 Workshop on Leveraging Models for Contact-Rich Manipulation*, 2023.

Kavisha Vidanapathirana, Shin-Fang Chng, Xueqian Li, and Simon Lucey. Multi-body neural scene flow. In *2024 International Conference on 3D Vision (3DV)*. IEEE, 2024.

Chaoyang Wang, Xueqian Li, Jhony Kaesemodel Pontes, and Simon Lucey. Neural Prior for Trajectory Estimation. In *CVPR*, pp. 6522–6532, 2022a. doi: 10.1109/CVPR52688.2022.00642.

Jun Wang, Xiaolong Li, Alan Sullivan, Lynn Abbott, and Siheng Chen. 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, 2022b.

Thomas Weng, Sujay Man Bajracharya, Yufei Wang, Khush Agrawal, and David Held. Fabricflownet: Bimanual cloth manipulation with a flow-based policy. In *Conference on Robot Learning*, pp. 192–202. PMLR, 2022.

Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. 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.

Wenxuan Wu, Zhi Yuan Wang, Zhuwen Li, Wei Liu, and Li Fuxin. Pointpwc-net: Cost volume on point clouds for (self-) supervised scene flow estimation. In *Eur. Conf. Comput. Vis.*, pp. 88–107. Springer, 2020.

Guangyao Zhai, Xin Kong, Jinhao Cui, Yong Liu, and Zhen Yang. FlowMOT: 3D Multi-Object Tracking by Scene Flow Association. *ArXiv*, abs/2012.07541, 2020.

Qingwen Zhang, Yi Yang, Heng Fang, Ruoyu Geng, and Patric Jensfelt. DeFlow: Decoder of Scene Flow Network in Autonomous Driving. *ICRA*, 2024a.

Qingwen Zhang, Yi Yang, Peizheng Li, Olov Andersson, and Patric Jensfelt. Seflow: A self-supervised scene flow method in autonomous driving. *arXiv preprint arXiv:2407.01702*, 2024b.

Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced Grouping and Sampling for Point Cloud 3D Object Detection. *arXiv preprint arXiv:1908.09492*, 2019.