# SMORE: Simultaneous Map and Object REconstruction

Nathaniel Chodosh*
Villanova University
nchodosh@villanova.edu

Anish Madan*
Carnegie Mellon University
anishmad@cs.cmu.edu

Simon Lucey
University of Adelaide
simon.lucey@adelaide.edu.au

Deva Ramanan
Carnegie Mellon University
deva@cs.cmu.edu

## Abstract

*We present a method for dynamic surface reconstruction of large-scale urban scenes from LiDAR. Depth-based reconstructions tend to focus on small-scale objects or large-scale SLAM reconstructions that treat moving objects as outliers. We take a holistic perspective and optimize a compositional model of a dynamic scene that decomposes the world into rigidly-moving objects and the background. To achieve this, we take inspiration from recent novel view synthesis methods and frame the reconstruction problem as a global optimization over neural surfaces, ego poses, and object poses, which minimizes the error between composed spacetime surfaces and input LiDAR scans. In contrast to view synthesis methods, which typically minimize 2D errors with gradient descent, we minimize a 3D point-to-surface error by coordinate descent, which we decompose into registration and surface reconstruction steps. Each step can be handled well by off-the-shelf methods without any retraining. We analyze the surface reconstruction step for rolling-shutter LiDARs, and show that deskewing operations common in continuous time SLAM can be applied to dynamic objects as well, improving results over prior art by an order of magnitude. Beyond pursuing dynamic reconstruction as a goal in and of itself, we propose that such a system can be used to auto-label partially annotated sequences and produce ground truth annotation for hard-to-label problems such as depth completion and scene flow. Please refer https://anishmadan23.github.io/smore/ for more visual results.*

## 1. Introduction

Dynamic scene understanding aims to produce a model of the world that explains all measurements over time. In the context of depth sensors, this problem is posed as dynamic surface reconstruction, where the goal is to produce a time-varying surface that matches a sequence of depth measurements. This problem has been widely studied in the context of handheld RGB-D sensors capturing human-scale scenes [23, 29, 44, 48]. However, investment in autonomous driving has created a new mode of depth capture — spinning LiDAR sensors atop moving vehicles — which is largely unaddressed by the existing research. Existing surface-based methods focus on reconstructing a few densely-scanned non-rigid objects, but autonomous driving scenes are typically composed of many sparsely-scanned rigid objects [4, 9]. In contrast, recent novel view synthesis methods have modeled AV scenes with a composition of rigid models [24, 33, 34, 43]. However, we find that they produce poor-quality reconstructions of the underlying geometry (see Fig. 3). In this work, we aim to achieve the best of both worlds and present a compositional surface model of LiDAR sequences. We find that this approach produces superior estimates of world and actor geometry as well as superior pose estimates of the ego-vehicle and tracked objects.

**Approach:** We address the dynamic scene reconstruction problem from a classic "analysis by synthesis" perspective; we synthesize a spacetime reconstruction via a compositional model of geometry and motion. We then measure the 3D error of the reconstruction with respect to the observed LiDAR scans. Finally, we optimize the geometry and motion to minimize this 3D error. We take care to formulate the optimization so that it can be efficiently decomposed into alternating steps of 1) estimating 6-DOF motion parameters of rigidly-moving components (including the moving ego-vehicle) and 2) estimating the geometry of each rigid component (including the static background). Such a decomposition allows us to leverage off-the-shelf solutions to the point registration and point-to-surface reconstruction problems, respectively. However, a naive imple-

*Equal Contribution

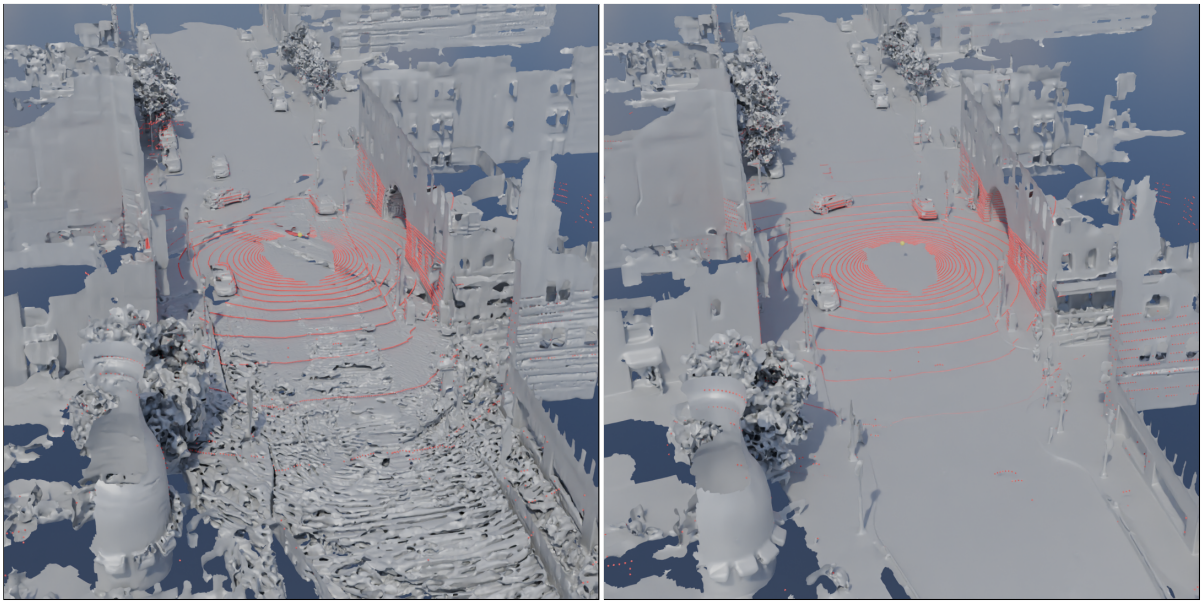

Figure 1. NuScenes surface reconstruction produced by aggregating LiDAR scans using human-annotated ego-pose and dynamic object bounding boxes (**left**). We introduce a global optimization that refines both ego and object poses so as to minimize a scan-to-surface reconstruction error, dramatically improving results (**right**). To do so, we find it crucial to model rolling shutter LiDAR effects, particularly for dynamic objects (Fig. 2). Please see the animation in the supplement.

mentation does not produce high-quality results. Instead, we find that the continuous shutter of the LiDAR sensor needs to be modeled. Recent radiance field methods (Neurad [33]) accomplish this by modeling each individual LiDAR return. Instead, we take inspiration from continuous SLAM literature, which *deskews* LidAR scans to compensate for ego motion [5, 8, 16, 20, 26]. We extend this idea and show how deskewing can *also* compensate for dynamic actor motion.

**Applications:** We view this work as a step towards generating dynamic scene reconstructions that provide high-quality annotations for downstream autonomous driving tasks. Labeling in-the-wild data is extremely costly, and as a result, many autonomous driving tasks rely on re-processing existing data of varying quality. For example, depth completion benchmarks use aggregated LiDAR sweeps to generate ground truth "dense" depth reconstructions [35]. This results in annotated data with well-documented occlusion errors and motion artifacts that are nonetheless still used for training and evaluation [41, 50]. An example of the depth maps produced by our method is shown in Fig. 3. Scene flow is another autonomous driving task that re-processes existing AV datasets, using annotated bounding box motion between frames as a proxy for the underlying ground-truth motion field [1, 18], which also has well-documented issues in evaluation [4]. Recent scene flow work [36] has also shown promise when scaled to the large amount of unlabeled LiDAR data that is available [42]. Accurate time-space reconstructions of the rigidly moving objects in the scene are critical to these tasks. We demonstrate that the ground-truth motion annotations are insufficient for producing these reconstructions and that our system significantly improves upon them.

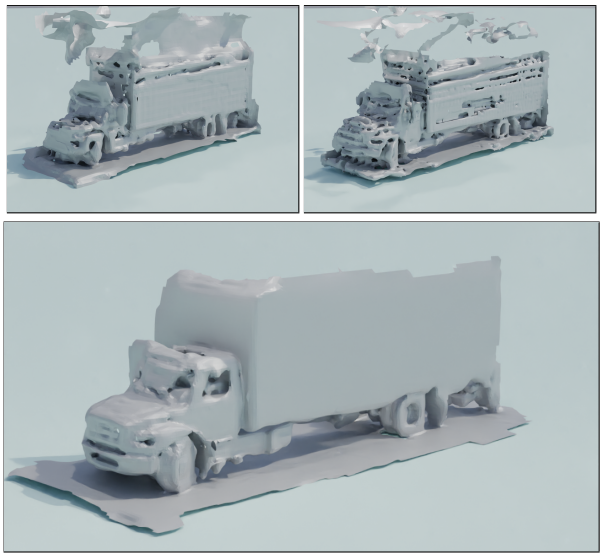

Figure 2. Dynamic object reconstructions using human-annotated bounding-box annotations (**top left**) tend to be noisy. Optimizing over object pose (**top right**) improves accuracy, while *de-skewing* scans to account for dynamic object motion is even more helpful (**bottom**). Please see the animation in the supplement.

Our main contributions are (1) posing the classic dynamic surface reconstruction problem in the context of

LiDAR-based urban scenes, (2) combining insights from actor decomposition of radiance fields and continuous-time SLAM to produce high-quality reconstructions that reduce error by 10X over prior art, and (3) proposing new downstream applications of such spacetime reconstructions.

## 2. Related Work

**Dynamic Surface Reconstruction:** Reconstruction of non-rigid surfaces from depth scanners has been studied for over two decades[22]. Early work overcame the inherent ill-posedness of the problem by relying on object-specific shape models for humans[31], faces[17] and hands[28]. Since then, many works have demonstrated template-free reconstruction in both the online[23] and offline settings[25]. This line of work is focused on highly deformable objects such as people and animals, which are very close to the depth sensor. As a result, they do not apply to the long-range, generally rigid world of autonomous driving scenes.

**Dynamic SLAM:** Since we are solving for the global map of the world as well as the sensor's location within it, our work is closely related to SLAM in general and specifically to Dynamic SLAM, sometimes called SLOT (Simultaneous Localization and Object Tracking). Many works identify dynamic objects to remove them from the global map[10], but some track dynamic objects and register new observations to an object template. Similar to our work, these approaches typically represent the world as a composition of rigid bodies[2, 7, 11, 32, 39, 45]. These methods are focused on real-time operation from RGB inputs rather than offline LiDAR processing. As a result, they generally do not reconstruct detailed surface representations of the tracked objects, although that has been proposed as a post-processing step to the tracked objects[14]. Also similar to our work are SLAM methods, which create a dense surface reconstruction of the global map[17, 37]. However, to our knowledge, none of these approaches reconstruct dynamic objects. Many works on continuous time SLAM with LiDAR sensors have tackled the rolling shutter problem[5, 8, 16, 20]. We take inspiration from their method of deskewing sweeps for ego-motion and apply it to dynamic actors (for the first time, to our knowledge).

**Asset Generation for Autonomous Driving:** Related to the object reconstruction component of our system is the line of work focused on creating high-quality mesh reconstructions of vehicles for simulation purposes[21, 40, 46]. These methods are similar to ours in that they reconstruct dense meshes of in-the-wild vehicles but have several key differences. First, since these systems aim to extract assets, not reconstruct complete sequences, they focus on objects that are close to the sensor and have accurate poses from object detection. Although this is not made explicit, the result is that these systems are made to operate on stationary objects, not dynamic ones. Second, they rely heavily on RGB information as well as depth sensors. As we show, reconstructing moving objects from spinning LiDARs requires careful handling of the rolling-shutter effect. This makes it challenging to incorporate global-shutter RGB cameras. The fact that these works make no mention of this further indicates that they do not handle dynamic objects.

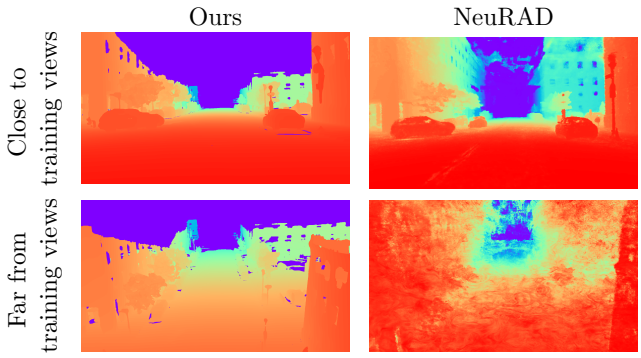


Figure 3. Depth maps produced by our method (**left**) as compared to those from a SOTA NeRF-based method[33] (**right**). In the **top** row we see that when depth maps are close to the training views, [33] and our method produce comparable results. However, in the **bottom** row, we see that moving the camera far from the training poses reveals large errors in the density field not present in our surface-based method.

**NeRFs** Neural Radiance fields have also been applied to automotive data recently [24, 33, 34, 43]. We take inspiration from the methods which model the world as a composition of rigid actors [33, 43]. These methods typically also make use of sparse LiDAR signals for geometry estimation, but we find that using an explicit surface-based representation yields superior results.

## 3. Problem Statement

We assume as input a sequence of LiDAR sweeps measured at timestamps $t \in \mathcal{T}$, and coarse tracks of $K$ objects. Since we are using a compositional model of the scene, we will need a coordinate frame for each component.

- **Ego coordinates:** This is the moving ego-vehicle coordinate frame used to measure input points, denoted as $e_t$.
- **Object coordinates:** Each dynamic object $i$ is assigned its own canonical coordinate frame at time $t$ where the $x$-direction is "forward", denoted as $o_t^i$.
- **World coordinates:** We represent the static background in a fixed world coordinate frame $w$, which we set equal to $e_1$ without loss of generality.

We use subscripts to denote the coordinate frame of given point $\mathbf{x}$ or set of points $\mathbf{X}$: e.g., $\mathbf{x}_{e_t}$,$\mathbf{X}_{e_t}$. We express the rigid transformation between coordinate frames as $\mathbf{T} \in R^{4\times4}$; e.g., the transformation from world to sensor coordinates $e_t$ is $\mathbf{T}_w^{e_t}$, while the transformation from object

to world coordinates is $\mathbf{T}^w_{o^i_t}$. Then, transformation from object to sensor coordinates at time $t$ is $\mathbf{T}^{e_t}_{o^i_t} = \mathbf{T}^{e_t}_w \mathbf{T}^w_{o^i_t}$.

We aim to decompose the scene into a set of surfaces that transform rigidly over time. Our approach is agnostic to the particular choice of surface representation, but we use triangular meshes since they are lightweight and widely used. We write the surface (mesh) of each of $K$ objects in the scene $\{\mathcal{M}_i\}_{i=1}^K$ as well as the background "object" $\mathcal{M}_0$. In a slight abuse of notation, we write $\mathbf{T}\mathcal{M}$ to denote the surface $\mathcal{M}$ transformed by $\mathbf{T}$ and write $\mathbf{TX}$ to express the transformed points $\mathbf{X}$. We write the composition of 2 surfaces as $[\mathcal{M}_1, \mathcal{M}_2]$. Finally, we will measure the 3D distance between a surface and a point cloud using the nearest neighbor loss

$$\mathcal{D}(\mathcal{M}, \mathbf{X}) = \sum_{\mathbf{x} \in \mathbf{X}} \min_{\mathbf{m} \in \mathcal{M}} \|\mathbf{m} - \mathbf{x}\|. \tag{1}$$

# 4. Objective

Our method aims to find the surfaces and object motions that best explain the LiDAR measurements. Using the notation from the previous section, we can express this goal as the optimization objective:

$$\min_{\{\mathcal{M}_i, \mathbf{T}^{e_t}_{o^i_t}, \mathbf{T}^{e_t}_w\}} \sum_{t \in \mathcal{T}} \mathcal{D}(\left[\mathbf{T}^{e_t}_w \mathcal{M}_0, \mathbf{T}^{e_t}_{o^1_t} \mathcal{M}_1, \ldots, \mathbf{T}^{e_t}_{o^K_t} \mathcal{M}_K\right], \mathbf{X}_{e_t}). \tag{2}$$

Intuitively, this equation sums up, for all $t$, the error between the point cloud at time $t$ and the reconstruction at time $t$. Each term in the summation uses the object and ego motions ($\mathbf{T}$) to compose all of the meshes ($\mathcal{M}$) into a reconstruction at time $t$. The reconstruction is then compared to the point cloud at that time ($\mathbf{X}_{e_t}$) using the error metric $\mathcal{D}$.

## 4.1. Decomposition

Our approach consists of applying coordinate descent to Equation (2): alternating between fixing the poses to **optimize surfaces** and then fixing the surfaces to **optimize poses**. This approach allows us to leverage off-the-shelf tools for each step of the optimization. In the following sections we derive the appropriate surface and pose optimization steps from the global objective.

The first step of both derivations decomposes the objective across objects. Let $\mathbf{X}^i_{e_t}$ denote the subset of points from $\mathbf{X}_{e_t}$ which fall on object $i$. This assignment can be coarse, and in practice, we assign points to the bounding box they fall into and to the background if they are not contained in any bounding box. Using this notation, we can further break down Eq. (2) into:

$$\min_{\{\mathcal{M}_i, \mathbf{T}^{e_t}_{o^i_t}, \mathbf{T}^{e_t}_w\}} \sum_{t \in \mathcal{T}} \sum_{i=0}^{K} \mathcal{D}(\mathbf{T}^{e_t}_{o^i_t} \mathcal{M}_i, \mathbf{X}^i_{e_t}), \tag{3}$$

where we let $o^0_t = w$ for notional simplicity. We can now derive the pose and surface optimization steps.

## 4.2. Optimize surfaces

Assuming fixed poses (initialized from LiDAR odometry and object tracks), we estimate new surfaces by solving

$$\mathcal{M}_i \leftarrow \arg\min_{\mathcal{M}_i} \sum_{t \in \mathcal{T}} \mathcal{D}(\mathbf{T}^{e_t}_{o^i_t} \mathcal{M}_i, \mathbf{X}^i_{e_t}). \tag{4}$$

We make use of two identities to simplify this optimization. First, we use the fact the distance is unaffected by a global rigid transformation to see that $\mathcal{D}(\mathbf{T}\mathcal{M}, \mathbf{X}) = \mathcal{D}(\mathcal{M}, \mathbf{T}^{-1}\mathbf{X})$. Second, if we write a set of points $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2]$ as a union of two disjoint sets $\mathbf{X}_1$ and $\mathbf{X}_2$, we can see that $\mathcal{D}(\mathcal{M}, [\mathbf{X}_1, \mathbf{X}_2]) = \mathcal{D}(\mathcal{M}, \mathbf{X}_1) + \mathcal{D}(\mathcal{M}, \mathbf{X}_2)$. Now we combine them to get:

$$\begin{aligned} \mathcal{M}_i &\leftarrow \arg\min_{\mathcal{M}_i} \sum_{t \in \mathcal{T}} \mathcal{D}(\mathbf{T}^{e_t}_{o^i_t} \mathcal{M}_i, \mathbf{X}^i_{e_t}) \\ &= \arg\min_{\mathcal{M}_i} \sum_{t \in \mathcal{T}} \mathcal{D}(\mathcal{M}_i, (\mathbf{T}^{e_t}_{o^i_t})^{-1} \mathbf{X}^i_{e_t}) \\ &= \arg\min_{\mathcal{M}_i} \mathcal{D}\left(\mathcal{M}_i, \left[(\mathbf{T}^{e_t}_{o^i_t})^{-1} \mathbf{X}^i_{e_t}, \ldots\right]\right). \end{aligned} \tag{5}$$

The final form of this equation can be interpreted as a standard static point-to-surface reconstruction problem. We use the recent Neural Kernel Surface Reconstruction [12], but any technique, such as Poisson surface reconstruction [15], could be used.

## 4.3. Optimize poses

Assuming fixed surfaces, we can estimate new poses by solving

$$\begin{aligned} \mathbf{T}^{e_t}_{o^i_t} &\leftarrow \arg\min_{\mathbf{T}^{e_t}_{o^i_t}} \mathcal{D}\left(\mathbf{T}^{e_t}_{o^i_t} \mathcal{M}_i, \mathbf{X}^i_{e_t}\right) \\ &= \arg\min_{\mathbf{T}^{e_t}_{o^i_t}} \mathcal{D}\left(\mathcal{M}_i, (\mathbf{T}^{e_t}_{o^i_t})^{-1} \mathbf{X}^i_{e_t}\right). \end{aligned} \tag{6}$$

This is a point-to-surface registration problem that is well-studied under the family of Iterative Closest Point (ICP) methods. However, there is an remaining complication due to the rolling shutter of LiDAR capture.

## 4.4. What *is* a LiDAR sweep?

Revolving LiDAR sensors do not have a global shutter. Instead, they rotate continuously and measure depth across 16-128 vertically arranged lasers, typically taking 100ms to complete a 360-degree rotation. Depth along each laser ray is measured with respect to the (potentially moving) ego sensor frame (see Fig. 5). Most datasets abstract away this continuous capture and instead provide LiDAR returns as

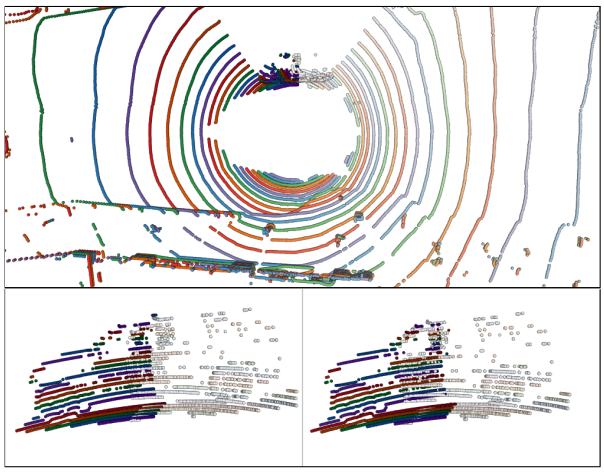

Figure 4. (**Left**) A LiDAR sweep where each point has been colored according to which laser it belongs to (hue) and the time within the sweep it was acquired (lighter is earlier, darker is later). A moving car is passing the ego-vehicle (traveling right) on the left and is captured at both the start and end of the sweep (**top right**), leading to distortion (the driver-side window is captured twice in different locations). Accounting for this distortion by modeling the object motion is key to the quality of our reconstructions (**bottom right**).

groups of 360-degree sweeps. This is convenient for many downstream applications but introduces a few subtle issues which we hope to shed some light on.

**Ego-Motion Distortion:** When the sensor moves during a sweep the resulting point cloud is distorted. This distortion has been removed in all of the popular LiDAR datasets through "deskewing", which transforms all of the measured points into the sensor's reference frame at the beginning or end of the sweep. We can use our existing notation to explain this process by letting our time index $t$ be a continuous variable rather than an integer frame index. For example, if we have a 16-beam LiDAR sensor that takes 1080 measurements in a single rotation, the first 16 points of our sequence are written as $\mathbf{X}_{e_{1/1080}}$. Importantly, our global optimization Eq. (2), mesh step Eq. (4), and pose step Eq. (6) are just as valid under this interpretation of a sweep "slice", but with 1080 times as many poses. This is computationally expensive and may underconstrain the optimization. To avoid this, we adopt a constant velocity model for poses between "keyframes" placed at the end of every complete sensor rotation. For example, we can express the continuous pose of the sensor for $0 < t < 1$ using the keyframe poses $\mathbf{T}^w_{e_0}, \mathbf{T}^w_{e_1}$ like so:

$$\begin{aligned} \mathbf{T}^{e_1}_{w} \left(\mathbf{T}^{e_0}_{w}\right)^{-1} &= \begin{bmatrix} \mathbf{R}_{3\times3} & \mathbf{v}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}, \quad \mathbf{w} = \log\left(\mathbf{R}\right) \\ \mathbf{T}^w_{e_t} &= \mathbf{T}^w_{e_1} \begin{bmatrix} e^{\mathbf{w}(1-t)} & \mathbf{v}(1-t) \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} = \mathbf{T}^w_{e_1} \mathbf{T}^{e^1}_{e_t}. \end{aligned} \tag{7}$$

Using these equations, points can be deskewed by transforming all of the points in a sweep to the coordinate frame $e_0$. *While deskewing generates the correct virtual point cloud for a static world, it fails to compensate for moving objects and complicates reasoning about the free space.*

**Actor-Motion Distortion:** Unlike ego-motion distortion, actor-motion distortion is still present in essentially all AV datasets. However, our spacetime optimization can correctly model moving objects by applying the same insight; just as we assumed that the ego-vehicle obeys a constant velocity model between keyframes, we can make the same assumption about *other* moving objects. However, care needs to be taken to ensure that the constant velocity assumption is applied to the object's motion *with respect to the world* as opposed to with respect to the ego-vehicle. Constant velocity in the world frame is not equivalent to constant velocity in the moving sensor frame due to the presence of rotations. With this in mind, we represent the object poses like so:

$$\begin{aligned} \left(\mathbf{T}^{o_1}_{e_1} \mathbf{T}^{e_1}_{w}\right) \left(\mathbf{T}^{o_0}_{e_0} \mathbf{T}^{e_0}_{w}\right)^{-1} &= \begin{bmatrix} \mathbf{R}_{3\times3} & \mathbf{v}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}, \quad \mathbf{w} = \log\left(\mathbf{R}\right) \\ \mathbf{T}^{o^i_t}_{e_t} &= \begin{bmatrix} e^{\mathbf{w}(1-t)} & \mathbf{v}(1-t) \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \mathbf{T}^{o^i_1}_{e_1} \mathbf{T}^{e_1}_{e_t} = \mathbf{T}^{o^i_t}_{o^i_1} \mathbf{T}^{o^i_1}_{e_1} \mathbf{T}^{e_1}_{e_t}. \end{aligned} \tag{8}$$

This factorization is not only the correct way of applying the constant velocity assumption but also makes it easy to deal with the fact that, in many cases, public datasets only release the ego-motion compensated points $\mathbf{T}^{e_1}_{e_t} \mathbf{X}_{e_t}$. With the above factorization, we can omit the first $\mathbf{T}^{e_1}_{e_t}$ transformation as it has already been applied. This is one of those fortunate situations where the easy and correct approaches are the same! In our algorithm, both ego and actor distortion can be accounted for by plugging Eq. (7) and Eq. (8) into our mesh and pose steps.

**Freespace errors:** Finally, we point out that the naive deskewing common in existing datasets may introduce a subtle "error": the deskewed point cloud no longer reveals the true free space in the scene. Many radiance field [6, 13, 21] and surface reconstruction [12] methods explicitly reason about the free space travelled by a LiDAR pulse along a ray from the sensor to a surface. However, such constraints no longer hold after deskewing. Our approach handles this problem by explicitly reasoning about the correct (time-varying) origin for each ray, and therefore the implied freespace, for each point. We provide additional visuals in the supplement.

## 5. Experiments

**Datasets:** All of our experiments are conducted on nuScenes[3] and Argoverse 2.0[42]. We focus primarily on nuScenes as its noisy annotations and sparse LiDAR present the greatest challenge to accurate geometry recovery. The nuScenes experiments are conducted on the logs

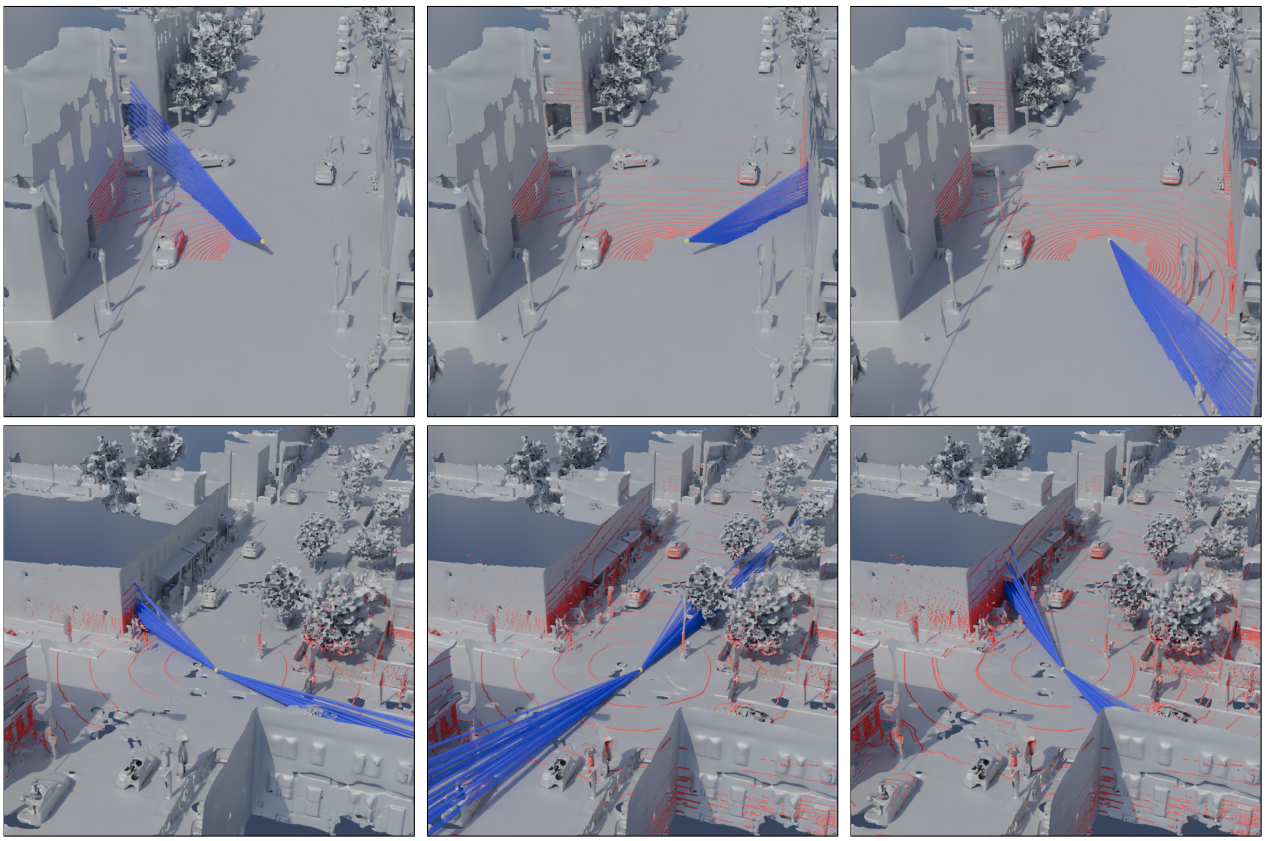

Figure 5. LiDAR is often abstracted as 360-degree sweeps captured with a global shutter, but is actually captured with a continuous rotating shutter from a moving ego-car. Our continuous-time optimization framework correctly models this, dramatically improving the quality of urban reconstructions. Here, we visualize the set of rays captured at a time instant (blue lines) for NuScenes (**top**) and Argoverse [42] (**bottom**). Interestingly, our approach is even more effective for recent AV datasets [30, 42] that employ *multiple* spinning lidars, which are often set to be out-of-phase to minimize interference (but adding to the inconsistency of a global shutter approximation).

used in the miniature set released by the dataset authors (we omit scene-0553 since it does not contain any ego-motion), except for the NVS experiment, which was conducted on scene-0103 due to computation constraints. The Argoverse 2.0 experiments are conducted on a similarly sized subset of the validation set; sequences are listed in the supplement.

### 5.1. Lidar Novel View Synthesis

Following previous work, we use LiDAR Novel View synthesis for our primary evaluation[13, 33]. In this task, we withhold 10% of the LiDAR sweeps when training our model and then aim to predict the held-out sweeps. We compare our method and NeuRAD[33] on this task by evaluating the chamfer distance and median L2 distance error between the synthesized point cloud and ground truth point cloud. Since the ground truth poses for both the ego-vehicle and actors are noisy, we face the same challenge as many NeRF-in-the-wild works[19, 49]: what pose to use when running test queries? We follow [49] and let each method use the test pose that achieves the lowest error.

**SMORE Details:** We run the SMORE optimization on the training views to obtain reconstructed meshes for objects and the background. Then, we optimize the test poses to by running one pose step (ICP). Finally, we iterate over the test sweeps to synthesize point clouds by computing ray-mesh intersections. We initialize the object tracks and bounding boxes either by using linear interpolation on the provided object annotations or with the output of an off-the-shelf LiDAR object tracker[27]. We initialize the ego-pose using an off-the-shelf LiDAR odometry method[37]. We then run 100 refinement iterations on all objects and background maps, with early stopping criteria to avoid wasted computation. Iterations are stopped if the mean registration error for an object falls below 1 centimeter for three consecutive iterations. For the mesh step, we use the default parameters of the publicly released Neural Kernel Surface Reconstruction model[12]. For the pose step, once we have deskewed the dynamic objects, we use a standard ICP implementation[51] with a point-to-plane loss, a robust Huber kernel with $k = 0.2$ and a matching threshold of 1.5

meters. For the tracking results, we use the Centerpoint-based[47] object detector LT3D[27] to extract bounding boxes in all frames. We then use greedy association to turn the detections into object tracks.

**NeuRAD Details:** We train NeuRAD on the training split using the same initialization as our method. For testing, however, the reference implementation does not support optimizing new poses that were not present at train time. Instaed, for each test sweep, we interpolate between the closest optimized training poses. Alternatively, we completely disable pose optimization and use the final optimized poses from our method. Note that, by default, NeuRAD also optimizes its dynamic actor poses. In this setting, we observed worse results when running train pose optimization. In the results shown, we fix actor poses from the initialization. Finally, we synthesize a point cloud by computing the expected depth along the rays given by each test sweep.

**Results**: In Tab. 3, we see that SMORE outperforms NeuRAD by order of magnitude in both chamfer distance and median depth error. This is the case even when NeuRAD is given the final optimized poses from our method, which we note do outperform the poses found through NeuRAD's optimization. We observe that the error persists due to rays going through "holes" in the density field, causing outlier points and uneven surface geometry, leading to high errors on oblique surfaces such as the road. See Fig. 7 for an example of the latter phenomenon.

### 5.2. Pose Estimation

**Actor Pose:** A key advantage of our method is its ability to recover from large errors in the input actor and ego poses. To test this, we run our method with annotations taken at various sample rates. Then, we compare our method's predicted object locations to the unsampled ground truth annotations using Average Translation Error metric[3]. We compute this metric over the nine test sequences and filter out objects that follow linear trajectories. As shown in Tab. 2, our method is robust even to extreme subsampling and consistently improves over the linearly interpolated poses, which are often used for downstream tasks[4]. We also evaluate the surface quality of our method across the different subsampling rates and when using an off-the-shelf tracker. The results in Tab. 1 further confirm the robustness of our method to input annotation errors.

**Ego Pose:** We wish to evaluate our method's ability to recover accurate ego-poses. However, the ground truth poses are too noisy to make a straightforward ground truth comparison useful. Instead, we show that the poses from our method are superior to the ground truth and [38] by training our NeRF baseline[33] on all three sets of poses. In Tab. 4, we can see that our poses result in far better scene reconstruction. We note that the large disparity between the NeuRAD results when trained on the exact same poses as our method (last row) confirms that the improvement in our method is not only due to better poses but also better geometry estimates.

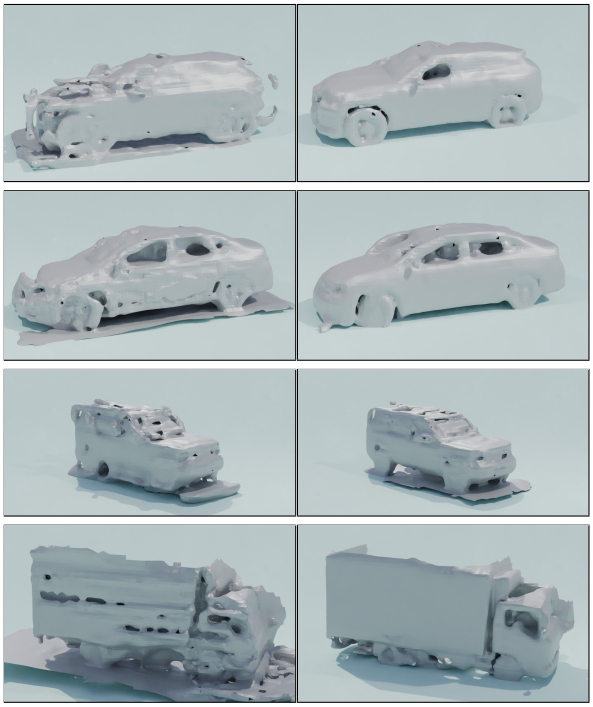

Figure 6. (**Left**) Each column shows nuScenes and Argoverse object reconstructions using ground truth poses compared to (**right**) ours.

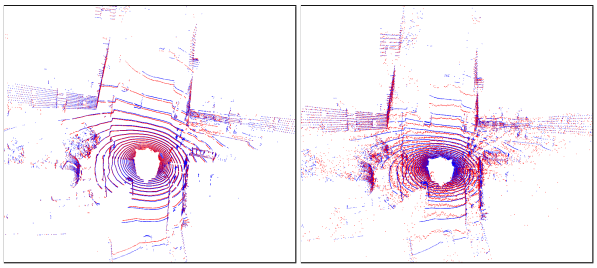

Figure 7. Examples of the synthesized point clouds from our method (**left**) versus NeuRAD [33] (**right**). Ground truth is shown in blue and the predicted point clouds are shown in red. We see that the density-based geometry of [33], while producing high-quality RGB renderings, is uneven, leading to high errors on flat oblique surfaces such as the road.

## 6. Qualitative Results

Visualizations of our foreground reconstructions on nuScenes and Argoverse are shown in Fig. 6. In the visualizations, we show that the ground-truth object annotations are not accurate enough to yield good reconstructions. Errors in bounding box alignment and orientation lead to point

| | NN Dist (m) ↓ | Acc Relax ↑ | Acc Strict ↑ |
|---|---|---|---|
| NKSR[12] + GT tracks | 0.071 | 0.9 | 0.76 |
| NKSR[12] + LT3D[27] | 0.071 | 0.9 | 0.76 |
| Ours + GT tracks | 0.048 | 0.96 | 0.91 |
| Ours + GT tracks (1 Hz) | 0.050 | 0.96 | 0.90 |
| Ours + GT tracks (0.5 Hz) | 0.048 | 0.96 | 0.91 |
| Ours + GT tracks (0.25 Hz) | 0.048 | 0.96 | 0.91 |
| Ours + LT3D[27] | 0.048 | 0.96 | 0.90 |

Table 1. Evaluation of our method's robustness to actor annotation errors (subsampling or real tracks). We measure reconstruction accuracy using the nearest-neighbor distance between the input point clouds and the reconstructed scene at each timestamp. We report the average distance and two accuracy metrics to characterize the distribution of errors. Specifically, we compute the percent of points less than 10cm and 5cm for the relaxed and strict metrics, respectively

| Annotation Rate | 1Hz | 0.5Hz | 0.25Hz |
|---|---|---|---|
| Interpolation | 0.29m | 0.40m | 0.74m |
| Ours | **0.20m** | **0.22m** | **0.52m** |

Table 2. Pose accuracy evaluation on nuScenes (using NuScene's default ATE metric), measured by comparing the bounding box locations predicted by our method to held-out ground truth labels provided at 2Hz. We compare our method to linearly interpolating the poses as is commonly done to create scene-flow labels [1].

aggregation errors, leading to poor surface reconstructions. Note that this shows that the human-provided labels are not accurate enough for high-quality reconstruction. Motion distortion from dynamic objects also contributes to the poor quality of the ground truth reconstructions. In Fig. 2, we show how accounting for this distortion can significantly improve the reconstructions.

Background reconstructions from nuScenes and Argoverse are shown in Fig. 8. For these "objects" the quality improvement comes from refining the ego-pose of the vehicle. As with foreground objects, ego-pose errors cause misalignment of the LiDAR sweeps, which become surface artifacts. However, the comparison is with a state-of-the-art LiDAR odometry method instead of the ground truth since we find odometry is generally superior.

## 7. Conclusion

In this work, we brought dense, dynamic reconstruction to the large-scale in-the-wild autonomous vehicle setting. We developed an optimization framework for understanding this problem and provided a simple yet effective solution based on decomposing the problem into well-studied sub-components. This solution yields high-quality reconstructions of both the foreground and background and can even account for subtle distortions in the input point clouds. We hope that this method will not only be useful for creating training and evaluation data for other perception tasks but will also promote active research in this challenging setting.

| Method | Chamfer Dist. ↓ | Depth ↓ |
|---|---|---|
| Ours | **0.26** | **0.0002** |
| NeuRAD[33] | 3.26 | 0.0053 |
| NeuRAD[33] w. Our Poses | 2.19 | 0.002 |

Table 3. Our reconstructions sigificantly outperform the state-of-the-art in LiDAR Novel View Synthesis [33] in terms of chamfer distance ($m^2$) and median depth error ($m^2$). Such methods can benefit from our optimized poses, suggesting that SMORE produces more accurate poses and benefits from an explicit surface model (Fig. 7).

| Pose Source | PSNR ↑ | SSIM ↑ | LPIPS ↓ | CD ↓ | Depth ↓ |
|---|---|---|---|---|---|
| Nuscenes-GT | 26.37 | 0.791 | 0.283 | 4.22 | 0.017 |
| KISS-ICP[38] | 24.99 | 0.76 | 0.328 | 2.58 | 0.016 |
| Ours | **27.52** | **0.821** | **0.238** | **2.19** | **0.002** |

Table 4. Ego-pose evaluation by fitting a SOTA NeRF method [33] to: the poses provided by nuScenes, those from a LiDAR odometry method, and the ones produced by our method. We see that our poses yield a superior scene model.

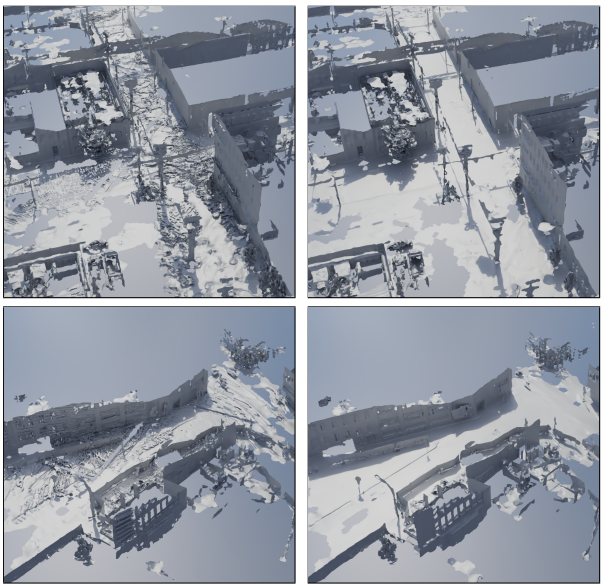

Figure 8. Map reconstruction comparisons on Argoverse (**top**) and nuScenes (**bottom**). The **left** column uses poses from a high-quality LiDAR odometry method[38] and **right** shows the result from our method's poses. The ground truth poses are reasonably accurate, but small errors in orientation and position lead to artifacts on flat surfaces and missing small details.

## References


[1] Baur, S.A., Emmerichs, D.J., Moosmann, F., Pinggera, P., Ommer, B., Geiger, A.: Slim: Self-supervised lidar scene flow and motion segmentation. In: ICCV. pp. 13126–13136 (2021) 2, 8

[2] Bibby, C., Reid, I.: Simultaneous localisation and mapping in dynamic environments (slamide) with reversible data association. In: Proceedings of Robotics: Science and Systems. vol. 66, p. 81 (2007) 3

[3] 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: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11621–11631 (2020) 5, 7

[4] Chodosh, N., Simon, L., Deva, R.: Re-evaluating lidar scene flow (2024) 1, 2, 7

[5] Dellenbach, P., Deschaud, J.E., Jacquet, B., Goulette, F.: Ct-icp: Real-time elastic lidar odometry with loop closure. In: 2022 International Conference on Robotics and Automation (ICRA). pp. 5580–5586. IEEE (2022) 2, 3

[6] Deng, K., Liu, A., Zhu, J.Y., Ramanan, D.: Depth-supervised nerf: Fewer views and faster training for free. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12882–12891 (2022) 5

[7] Dewan, A., Caselitz, T., Tipaldi, G.D., Burgard, W.: Motion-based detection and tracking in 3d lidar scans. In: 2016 IEEE international conference on robotics and automation (ICRA). pp. 4508–4513. IEEE (2016) 3

[8] Droeschel, D., Behnke, S.: Efficient continuous-time slam for 3d lidar-based online mapping. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 5000–5007. IEEE (2018) 2, 3

[9] Gojcic, Z., Litany, O., Wieser, A., Guibas, L.J., Birdal, T.: Weakly supervised learning of rigid 3d scene flow. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5692–5703 (2021) 1

[10] Hähnel, D., Schulz, D., Burgard, W.: Map building with mobile robots in populated environments. In: IROS. pp. 496–501 (2002) 3

[11] Henein, M., Zhang, J., Mahony, R., Ila, V.: Dynamic slam: The need for speed. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 2123–2129. IEEE (2020) 3

[12] Huang, J., Gojcic, Z., Atzmon, M., Litany, O., Fidler, S., Williams, F.: Neural kernel surface reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4369–4379 (2023) 4, 5, 6, 8, 13

[13] Huang, S., Gojcic, Z., Wang, Z., Williams, F., Kasten, Y., Fidler, S., Schindler, K., Litany, O.: Neural lidar fields for novel view synthesis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 18236–18246 (2023) 5, 6

[14] Jiang, C., Fougerolle, Y., Fofi, D., Demonceaux, C.: Dynamic 3d scene reconstruction and enhancement. In: Image Analysis and Processing-ICIAP 2017: 19th International Conference, Catania, Italy, September 11-15, 2017, Proceedings, Part I 19. pp. 518–529. Springer (2017) 3

[15] Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the fourth Eurographics symposium on Geometry processing. vol. 7, p. 0 (2006) 4

[16] Le Gentil, C., Vidal-Calleja, T.: Continuous latent state preintegration for inertial-aided systems. The International Journal of Robotics Research **42**(10), 874–900 (2023) 2, 3

[17] Li, H., Yu, J., Ye, Y., Bregler, C.: Realtime facial animation with on-the-fly correctives. ACM Trans. Graph. **32**(4), 42–1 (2013) 3

[18] Li, X., Kaesemodel Pontes, J., Lucey, S.: Neural scene flow prior. NeurIPS **34**, 7838–7851 (2021) 2

[19] Lin, C.H., Ma, W.C., Torralba, A., Lucey, S.: Barf: Bundle-adjusting neural radiance fields. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 5741–5751 (2021) 6

[20] Lv, J., Hu, K., Xu, J., Liu, Y., Ma, X., Zuo, X.: Clins: Continuous-time trajectory estimation for lidar-inertial system. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 6657–6663. IEEE (2021) 2, 3

[21] Manivasagam, S., Wang, S., Wong, K., Zeng, W., Sazanovich, M., Tan, S., Yang, B., Ma, W.C., Urtasun, R.: Lidarsim: Realistic lidar simulation by leveraging the real world. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11167–11176 (2020) 3, 5

[22] Mitra, N.J., Flöry, S., Ovsjanikov, M., Gelfand, N., Guibas, L.J., Pottmann, H.: Dynamic geometry registration. In: Symposium on geometry processing. pp. 173–182 (2007) 3

[23] Newcombe, R.A., Fox, D., Seitz, S.M.: Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 343–352 (2015) 1, 3

[24] Ost, J., Mannan, F., Thuerey, N., Knodt, J., Heide, F.: Neural scene graphs for dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2856–2865 (June 2021) 1, 3

[25] Palafox, P., Božič, A., Thies, J., Nießner, M., Dai, A.: Npms: Neural parametric models for 3d deformable shapes. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12695–12705 (2021) 3

[26] Park, C., Moghadam, P., Kim, S., Elfes, A., Fookes, C., Sridharan, S.: Elastic lidar fusion: Dense map-centric continuous-time slam. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 1206–1213. IEEE (2018) 2

[27] Peri, N., Dave, A., Ramanan, D., Kong, S.: Towards long-tailed 3d detection. In: Conference on Robot Learning. pp. 1904–1915. PMLR (2023) 6, 7, 8, 13

[28] Qian, C., Sun, X., Wei, Y., Tang, X., Sun, J.: Realtime and robust hand tracking from depth. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1106–1113 (2014) 3

[29] Runz, M., Buffier, M., Agapito, L.: Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. In: 2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR). pp. 10–20. IEEE (2018) 1

[30] Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al.: Scalability in perception for autonomous driving: Waymo open dataset. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2446–2454 (2020) 6

[31] Taylor, J., Shotton, J., Sharp, T., Fitzgibbon, A.: The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 103–110. IEEE (2012) 3

[32] Tian, R., Zhang, Y., Cao, Z., Zhang, J., Yang, L., Coleman, S., Kerr, D., Li, K.: Object slam with robust quadric initialization and mapping for dynamic outdoors. IEEE Transactions on Intelligent Transportation Systems (2023) 3

[33] Tonderski, A., Lindström, C., Hess, G., Ljungbergh, W., Svensson, L., Petersson, C.: Neurad: Neural rendering for autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14895–14904 (2024) 1, 2, 3, 6, 7, 8, 12

[34] Turki, H., Zhang, J.Y., Ferroni, F., Ramanan, D.: Suds: Scalable urban dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12375–12385 (2023) 1, 3

[35] Uhrig, J., Schneider, N., Schneider, L., Franke, U., Brox, T., Geiger, A.: Sparsity invariant cnns. In: 2017 international conference on 3D Vision (3DV). pp. 11–20. IEEE (2017) 2

[36] Vedder, K., Peri, N., Chodosh, N., Khatri, I., Eaton, E., Jayaraman, D., Liu, Y., Ramanan, D., Hays, J.: ZeroFlow: Scalable Scene Flow via Distillation (2024) 2

[37] Vizzo, I., Chen, X., Chebrolu, N., Behley, J., Stachniss, C.: Poisson surface reconstruction for lidar odometry and mapping. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). pp. 5624–5630. IEEE (2021) 3, 6

[38] Vizzo, I., Guadagnino, T., Mersch, B., Wiesmann, L., Behley, J., Stachniss, C.: KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way **8**(2), 1–8 (2023). https://doi.org/10.1109/LRA.2023.3236571 7, 8, 12, 13

[39] Wang, C.C., Thorpe, C., Thrun, S., Hebert, M., Durrant-Whyte, H.: Simultaneous localization, mapping and moving object tracking. The International Journal of Robotics Research **26**(9), 889–916 (2007) 3

[40] Wang, J., Manivasagam, S., Chen, Y., Yang, Z., Bârsan, I.A., Yang, A.J., Ma, W.C., Urtasun, R.: Cadsim: Robust and scalable in-the-wild 3d reconstruction for controllable sensor simulation. In: Conference on Robot Learning. pp. 630–642. PMLR (2023) 3

[41] Wang, Y., Dai, Y., Liu, Q., Yang, P., Sun, J., Li, B.: Cu-net: Lidar depth-only completion with coupled u-net. IEEE Robotics and Automation Letters **7**(4), 11476–11483 (2022) 2

[42] 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) 2, 5, 6

[43] Xie, Z., Zhang, J., Li, W., Zhang, F., Zhang, L.: S-nerf: Neural radiance fields for street views. In: International Conference on Learning Representations (ICLR) (2023) 1, 3

[44] Yang, G., Vo, M., Neverova, N., Ramanan, D., Vedaldi, A., Joo, H.: Banmo: Building animatable 3d neural models from many casual videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2863–2873 (2022) 1

[45] Yang, S., Scherer, S.: Cubeslam: Monocular 3-d object slam. IEEE Transactions on Robotics **35**(4), 925–938 (2019) 3

[46] Yang, Z., Manivasagam, S., Chen, Y., Wang, J., Hu, R., Urtasun, R.: Reconstructing objects in-the-wild for realistic sensor simulation 3

[47] Yin, T., Zhou, X., Krahenbuhl, P.: Center-based 3d object detection and tracking. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11784–11793 (2021) 7

[48] Yu, T., Zheng, Z., Guo, K., Liu, P., Dai, Q., Liu, Y.: Function4d: Real-time human volumetric capture from very sparse consumer rgbd sensors. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5746–5756 (2021) 1

[49] Zhang, J., Yang, G., Tulsiani, S., Ramanan, D.: Ners: Neural reflectance surfaces for sparse-view 3d reconstruction in the wild. Advances in Neural Information Processing Systems **34**, 29835–29847 (2021) 6

[50] Zhao, Y., Bai, L., Zhang, Z., Huang, X.: A surface geometry model for lidar depth completion. IEEE Robotics and Automation Letters **6**(3), 4457–4464 (2021) 2

[51] Zhou, Q.Y., Park, J., Koltun, V.: Open3D: A modern library for 3D data processing. arXiv:1801.09847 (2018) 6