nature machine intelligence

Article

https://doi.org/10.1038/s42256-025-01083-x

Towards generalizable and interpretable three-dimensional tracking with inverse neural rendering

Received: 3 September 2024

Accepted: 23 June 2025

Published online: 04 August 2025

Check for updates

Julian Ost[1,3], Tanushree Banerjee[1,3], Mario Bijelic[1,2] & Felix Heide[1,2]

Today, the most successful methods for image-understanding tasks rely on feed-forward neural networks. Although this approach offers empirical accuracy, efficiency and task adaptation through fine-tuning, it also comes with fundamental disadvantages. Existing networks often struggle to generalize across different datasets, even on the same task. By design, these networks ultimately reason about high-dimensional scene features, which are challenging to analyse. This is true especially when attempting to predict three-dimensional (3D) information based on two-dimensional images. We propose to recast vision problems with RGB inputs as an inverse rendering problem by optimizing through a differentiable rendering pipeline over the latent space of pretrained 3D object representations and retrieving latents that best represent object instances in a given input image. Specifically, we solve the task of 3D multi-object tracking by optimizing an image loss over generative latent spaces that inherently disentangle shape and appearance properties. Not only do we investigate an alternative take on tracking but our method also enables us to examine the generated objects, reason about failure situations and resolve ambiguous cases. We validate the generalization and scaling capabilities of our method by learning the generative prior exclusively from synthetic data and assessing camera-based 3D tracking on two large-scale autonomous robot datasets. Both datasets are completely unseen to our method and do not require fine-tuning.

Inverse rendering offers a new perspective on computer vision by combining differentiable rendering pipelines[1] and generative models[2] as a prior for spatial reasoning. Forward rendering describes the synthesis of two-dimensional images from a three-dimensional (3D) scene description. By contrast, inverse rendering is the process of inferring a 3D scene description solely from two-dimensional image observations of the given scenes[1]. Existing image-understanding methods almost exclusively use feed-forward neural networks for performing vision tasks, including segmentation[3–5], object detection[6–8], object tracking[9,10] and pose estimation[11]. Typically, these approaches learn network weights using large, labelled datasets. At inference time, the trained network layers sequentially process a given two-dimensional image to make a prediction. Despite being a successful approach across disciplines from robotics to health and being effective in operating at real-time rates, this approach also comes with several limitations: (1) Networks trained on data captured with a specific camera and geography generalize poorly. (2) They typically rely on high-dimensional internal feature representations, which are often not interpretable, making it hard to identify and reason about failure cases. (3) It is challenging to enforce 3D geometrical constraints and priors during inference.

[1]Department of Computer Science, Princeton University, Princeton, NJ, USA. [2]Torc Robotics, Blacksburg, VA, USA. [3]These authors contributed equally: Julian Ost, Tanushree Banerjee. e-mail: fheide@princeton.edu

We focus on multi-object tracking as a task at the heart of autonomous robotics that must tackle all these challenges. Accurate multi-object tracking is essential for safe robotic planning. Although approaches using lidar point clouds (and camera image input) are successful because of the explicitly measured depth[12–18], camera-based approaches to 3D multi-object tracking have been studied only recently[9,19–26]. Monocular tracking methods, typically consisting of independent detection, 3D dynamic models and matching modules, often struggle, as the errors in the distinct modules tend to accumulate. Moreover, wrong poses in the detections can lead to ID switches in the matching process.

We propose an alternative approach that recasts visual inference problems as inverse rendering tasks, jointly solving them at test time by optimizing over the latent space of a generative object representation. Specifically, we combine object retrieval through the inversion of a rendering pipeline and a learned object model with a 3D object-tracking pipeline (Fig. 1a). This approach allows us to reason about the 3D shape, appearance and 3D trajectory of an object simultaneously from only a monocular image input. The location, pose, shape and appearance parameters corresponding to the anchor objects are then iteratively refined with test-time optimization to minimize the distance between their corresponding generated objects and the given input image. Rather than directly predicting scene and object attributes, we optimize over a latent object representation to synthesize image regions that best explain the observed image. Then, we match the inverse rendered objects by comparing their optimized representations. As the proposed method relies on image renderings of all tracked objects, it also provides a new tool for interpretable debugging and analysis, for example, when the association between tracked object instances in adjacent frames fails.

Our method hinges on an efficient rendering pipeline and generative object representation at its core. Although the approach is not tied to a specific object representation, we adopt GET3D (ref. 2) as the generative object prior. It is is trained only on synthetic data[27] to synthesize textured meshes and corresponding images with an efficient differentiable rendering pipeline[28]. Note that popular implicit shape and object representations either do not support class-specific priors[29,30] or require expensive volume sampling[31].

The proposed method builds on the inductive geometry priors embedded in our rendering forward model by solving several different tasks simultaneously. Our method refines the object pose as a by-product merely by learning to represent objects of a given class. Recovering object attributes with inverse rendering also provides interpretability 'for free', once our proposed method detects an object at test time. It can extract the parameters of the corresponding representation alongside the rendered input view, which is human-interpretable and, as such, offers insights into the tracking process. This structured representation facilitates reasoning about failure cases and contributes to the explainability of the tracking decision.

We validate that the method naturally exploits 3D geometry priors and generalizes across unseen domains and datasets within the context of 3D multi-object tracking in driving scenes, without requiring retraining or fine-tuning on new data. To do this, we combine the proposed inverse rendering approach with an object dynamics model and matching strategy across adjacent frames (Fig. 1a). We match all objects in adjacent time steps by computing similarity across all available state parameters, including inverse rendered object shapes, textures and optimized poses. After training solely on simulated object appearance data, we test on nuScenes[32] and Waymo driving[33] datasets. Note that this setting is like offline auto-annotation for large-scale datasets[34,35], which requires generalizable methods that often do not have access to training data.

Although untrained, our method outperforms both existing dataset-agnostic multi-object tracking approaches and dataset-specific learned approaches[20] when operating on the same inputs. Although we evaluate the method for single-class vehicle tracking as a representative and well-explored task in driving, we confirm that the method generalizes to several classes for broader tracking scenarios. The approach achieves a 57.8% higher recall than existing learning-based methods transferred to unseen datasets. Moreover, we report an average multi-object tracking accuracy (AMOTA) of 0.413 which is a 6.5% improvement over the next best generalizing method. See Supplementary Video or https://light.princeton.edu/inverse-rendering-tracking.

Results

Single-shot object retrieval with inverse rendering

In the following, we assess the proposed approach, which is described in Methods in detail. Having trained our generative scene model solely on simulated data[27], we test the generalization capabilities on the nuScenes[32] and Waymo[33] datasets, both of which are unseen by the method. We analyse generative outputs of the test-time optimization and compare them against existing 3D multi-object trackers[9,20,24,26,36] on camera-only data.

Although trained only on general object-centric synthetic data, ShapeNet[27], our method is capable of fitting a sample from the generative prior to observed objects in real datasets that match the vehicle type, colour and overall appearance closely, effectively making our method dataset-agnostic. We analyse the generations during optimization in the following.

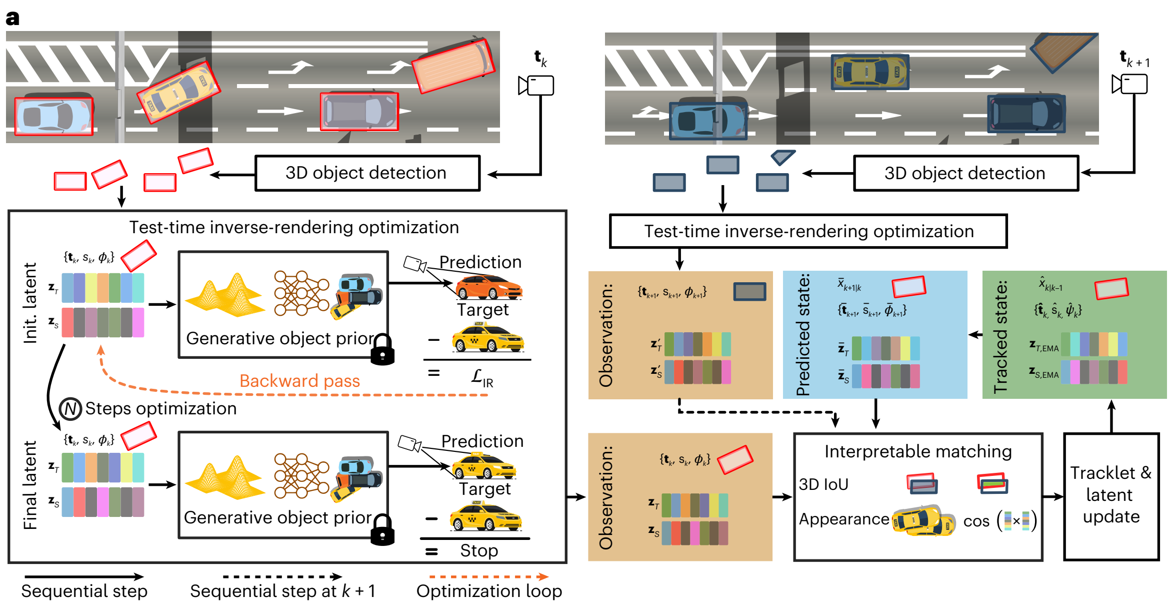
Given an image observation and coarse detections, our method aims to find the best 3D representation, including pose and appearance, solely with inverse rendering. In Fig. 1b we analyse this iterative optimization process, following scheduled optimization as described. We observe that the colour of an object is inferred in only two steps. Further, we can observe that even though the initial pose is incorrect, the rotation and translation are optimized jointly with inverse rendering together with the shape and scale of the objects, thus recovering from the suboptimal initial guesses. The shape representation close to the observed object is reconstructed in only five steps. Quantitative metrics for reconstruction show improving quality of optimized objects. We provide a numerical evaluation in Supplementary Note 13.

Generalization

To provide a fair comparison of 3D multi-object tracking methods using monocular inputs, we compare against existing methods by running all our evaluations with the method reference code. We evaluate only methods that consider past frames but have no knowledge about future frames, which is a different task. Although our method does not store the full history length of all images, we allow such memory techniques for other methods. We consider only purely mono-camera-based tracking methods. In contrast to our method, most existing methods we compare to are fine-tuned on the respective training set. For all two-stage detect-and-track methods, we use CenterPoint[13] as the detection method. We compare with CenterTrack[20] as an established learning-based baseline and present results from the very recent PF-Track[25], a transformer-based tracking method, QTrack[24] as a metric learning method, and QD-3DT (ref. 9) as a state tracker based on long short-term memory combined with image feature matching. Of all learning-based methods, only CenterTrack[20] allows us to evaluate tracking performance with identical detections. Finally, we compare with AB3DMOT[36], which builds on an arbitrary 3D detection algorithm and combines it with a modified Kalman filter[37] to track the state of each object. AB3DMOT[36] and the proposed method are the only methods that are data-agnostic in the sense that they have not seen the training dataset. For a fair evaluation of these generalization capabilities in learning-based methods, we include another version of QD-3DT solely trained on the Waymo Open Dataset[33] and evaluate on nuScenes[32]. We discuss the findings in the following.

Table 1 reports quantitative results on the test split of the nuScenes tracking dataset[32] on the car and motorcycle object class for all six

https://doi.org/10.1038/s42256-025-01083-x



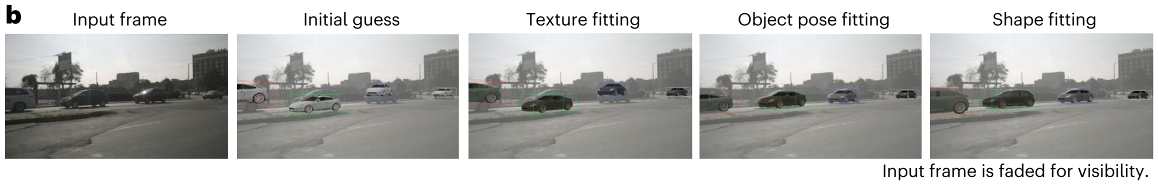


Fig. 1 | Inverse rendering for monocular multi-object tracking and inverse rendering optimization. a, We initialize the embedding codes of an object generator $\mathbf{z}_S$ for shape and $\mathbf{z}_T$ for texture for each detected object k. The generative object prior (for example, GET3D (ref. 2), pretrained on synthetic data) is frozen. Only embedding codes for an object's geometry $\mathbf{z}_S$ and texture $\mathbf{z}_T$, location $\mathbf{t}_k$, rotation ψ_k and size $\mathbf{s}_k$ for each object instance k are optimized with inverse rendering to fit the image observation best. The inverse rendering loss ($\mathcal{L}_{IR}$) quantifies the discrepancy between the observed and rendered images to guide the optimization. Solid connectors denote sequential processing steps, and dashed connectors indicate iterative feedback loops for optimization. The process terminates after a maximum fixed number of steps or when $\mathcal{L}_{IR}$ converges. Inverse rendered texture and shape embeddings and refined object locations are provided to the matching stage to match predicted states of tracked objects of past with their historical, exponentially moving averaged (EMA) texture and shape embeddings $\mathbf{z}_{T,EMA}$ and $\mathbf{z}_{S,EMA}$, and new observations. Matched and new tracklets are updated, and unmatched tracklets are ultimately discarded before predicting states in the next step (data from ref. 33). **b**, An example of this test-time optimization method with zoomed-in views of rendered objects. From left to right: the observed image, the rendering predicted by the initial starting point latent embeddings, the predicted rendered objects after the texture code is optimized, the predicted rendered objects after the translation, scale and rotation are optimized, and the predicted rendered objects after the shape latent code is optimized. The ground-truth images are faded to show our rendered objects clearly. Our proposed method effectively refines the predicted texture, pose and shape over several optimization steps, even if initialized with poses or appearances far from the target, all found at test time with inverse rendering. Init., initial; IoU, intersection over union.

cameras (Fig. 2). We list results for the multi-object tracking accuracy (MOTA)[38] metric, the AMOTA[36] metric, average multi-object tracking precision (AMOTP)[36] and recall of all methods. First, we evaluate a version of QD-3DT (ref. 9) that has been trained on the Waymo Open Dataset[33] but tested on nuScenes. This experiment is reported in row four of Table 1 and confirms that recent end-to-end detection and tracking methods do not perform well on unseen data (see qualitative results in Supplementary Note 9) on cars and fail on sparse object classes. Moreover, perhaps surprisingly, even when using the same vision-only detection backbone as in our approach, the established end-to-end trained baseline CenterTrack[20], which has seen the dataset, performs worse than our method. Our inverse rendering method outperforms the general tracker AB3DMOT[36] on the car class and shows higher precision and recall and comparable accuracy performance on the rare motorcycle class. When other methods are given access to the dataset, recent learning-based methods such as the end-to-end method based on long short-term memory QD-3DT (ref. 9) perform on par across all classes. Only the most recent transformer-based methods such as PF-Track[25] and QTrack[24], which use a quality-based association model on a large set of learned metrics, such as heat maps and depth, achieve higher scores. Note again, that these methods, in contrast to the proposed method, have been trained on this dataset and cannot be evaluated independently of their detector performance.

In Fig. 2, we ablate the optimization objective, which is composed of the RGB mean-squared-error loss, a learned perceptual loss (Supplementary Note 5) and equation (8) as well as the proposed schedule, and we provide the design choices. The absence of an optimization schedule led to less robust matching, as the quantitative and qualitative results in Supplementary Fig. 3 reveal. However, the core efficacy of our tracking method remained intact, as indicated in the last row of Supplementary Table 2. This nuanced understanding underscores the importance of component interplay in our approach.

We visualize the rendered objects predicted by our tracking method in Fig. 3. We show an observed image from a single camera at time step $k = 0$, followed by rendered objects overlaid over the observed image at time steps $k = 0, 1, 2$ and 3 along with their respective bounding boxes, with colour-coded tracklets. Our method does not lose any tracks in challenging scenarios in diverse scenes shown here from dense urban areas to suburban traffic crossings, and it handles occlusions and clutter effectively.

We additionally provide qualitative results from the 3D tracking on the validation set in the Waymo Open Dataset[33] in Fig. 3. The only public

Table 1 | Quantitative evaluation for camera-only multi-object tracking

Training data unseen	Method	Car				Motorcycle				Modality
		AMOTA ↑	AMOTP (m) ↓	Recall ↑	MOTA ↑	AMOTA ↑	AMOTP (m) ↓	Recall ↑	MOTA ↑	
×	PF-Track	0.622	0.916	0.719	0.558	0.448	1.245	0.457	0.384	Camera
×	QTrack	0.692	0.753	0.760	0.596	0.531	1.098	0.861	0.500	Camera
×	QD-3DT	0.425	1.258	0.563	0.358	0.253	1.543	0.437	0.243	Camera
✓	QD-3DT (trained on WOD)	0.000	1.893	0.226	0.000	0.000	2.000	0.000	0.000	Camera
× (CP)	CenterTrack	0.202	1.195	0.313	0.134	0.011	1.636	0.141	0.033	Camera
✓ (CP)	AB3DMOT	<u>0.387</u>	**1.158**	<u>0.506</u>	<u>0.284</u>	**0.254**	<u>1.549</u>	<u>0.360</u>	**0.232**	Camera
✓ (CP)	Inverse neural rendering (ours)	**0.402**	<u>1.213</u>	**0.521**	**0.315**	<u>0.244</u>	**1.479**	**0.389**	<u>0.220</u>	Camera

Bold entries denotes best and underlined second best scores for methods that did not train on the dataset or use the same detection backbone.

a

Ablation study of loss components

Method	AMOTA ↑	Recall ↑	MOTA ↑
$\mathcal{L}_{IR}$ & $\mathcal{L}_{embed}$	0.112	0.264	0.113
$\mathcal{L}_{IR}$	0.103	0.236	0.112
$\mathcal{L}_{perceptual}$	0.100	0.251	0.101
$\mathcal{L}_{RGB}$	NA	NA	NA
No schedule	0.102	0.224	0.110

b

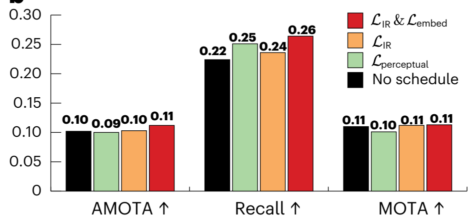


Fig. 2 | Quantitative Evaluation and Ablation Experiments for Camera-only Multi-Object Tracking. Evaluation on 'cars' and 'motorcycles' in the test split of the nuScenes tracking dataset[33] in **a**. Our IR-based tracker outperforms the recent AB3DMOT[37] on all metrics and CenterTrack[21] on accuracy on 'cars' and shows competitive and better performance on 'motorcycles' respectively. All three methods use the same detection backbone for fair comparison, while only CenterTrack requires end-to-end training on the dataset. Even when allowing other methods (but not ours) to train on nuScenes[33], the proposed method performs on par with QD-3DT[19]. We note that QD-3DT trained on the Waymo Open Dataset (WOD)[34] does *not generalize* to nuScenes and does not achieve competitive results on cars and fails on motorcycles. Only very recent transformer-based methods, such as PF-Track[26] and the metric learning approach of Q-Track[25] achieve a higher score and require end-to-end training on each dataset. In **a**, 'CP' denotes the vision-only version of CenterPoint[21] was used for object detection. **Bold** denotes best and <u>underlined</u> second best for methods that did not train on the dataset or use the same detection backbone. In **b**, we report ablation experiments on a small subset of the nuScenes[33] validation set. We analyse the proposed optimization scheme including the loss components and optimization schedule. Here $\mathcal{L}_{IR}$ is the sum of the RGB MSE and learned perceptual loss, as described in Eq. S2 and Eq. S3 in Supplementary Note 5 in Supplementary Information. The INR loss function components $\mathcal{L}_{RGB}$ fail due to the optimizer fitting objects to the background instead, increasing the size of each object and resulting in an out-of-memory error.

results on the provided test set are presented for QD-3DT (ref. 9), which may indicate that it fails on this dataset. Although the size of the dataset and its variety is of high interest for all autonomous driving tasks, ref. 9 concludes that vision-only test set evaluation is not representative of a test set developed for surround-view lidar data on partial unobserved camera images only. As such, we provide qualitative results in Fig. 3 that validate that the method achieves tracking of similar quality on all datasets, thus providing a generalizable tracking approach. This is further verified by the results of experiments on tracking presented in Fig. 3, which show similar performance across several classes (car and motorcycle). Although the proposed method is limited to rigid objects, we outline potential future extensions to deformable classes, such as humans and animals, in Supplementary Note 11. We demonstrate the potential of this direction with qualitative single-shot object retrieval for the pedestrian class in an extended multi-class experiment. For all experiments, we show that our method does not lose tracks on both Waymo[33] and nuScenes[32] scenes in diverse conditions.

Analysis and interpretation

By visualizing the rendered objects and analysing the matching and loss components, our method allows us to reason about and explain success and failure cases effectively. The rendered output images provide interpretable inference results that explain successful or failed matching due to shadows, appearance, shape or pose. For example, the blue car in the inverse rendered inference (column 2 in the top row of Fig. 3b) was incorrectly matched due to an appearance mismatch in a shadow region. Note that a future rendering model including ambient illumination may resolve this ambiguity; see the discussion in Supplementary Note 3.

Figure 4 shows inverse rendered scene graphs in isolation and bird's-eye-view (BEV) tracking outputs showing the layout. Combined with rendered object instance masks $M_{c,p}$, this method can be directly leveraged for free-space detection in the image space without requiring explicit segmentation (Supplementary Fig. 7). Our method accurately recovers the object pose, instance type, appearance and scale. As such, our approach directly outputs a 3D model of the full scene, that is, layout and object instances, along with the temporal history of the scene recovered through tracking. This rich scene representation can be directly ingested by downstream planning and control tasks or simulation methods to train downstream tasks. As such, the method also allows us to reason about the scene by leveraging the 3D information provided by our predicted 3D representations. The 3D locations, object orientations and sizes recovered from such visualizations can not only enable us to explain the predictions of our object-tracking method, especially in the presence of occlusions or ID switches, but can also be used in other downstream tasks that require a rich 3D understanding, such as planning.

Discussion

In this work, we investigate inverse neural rendering as an alternative to existing feed-forward tracking methods. Specifically, we recast

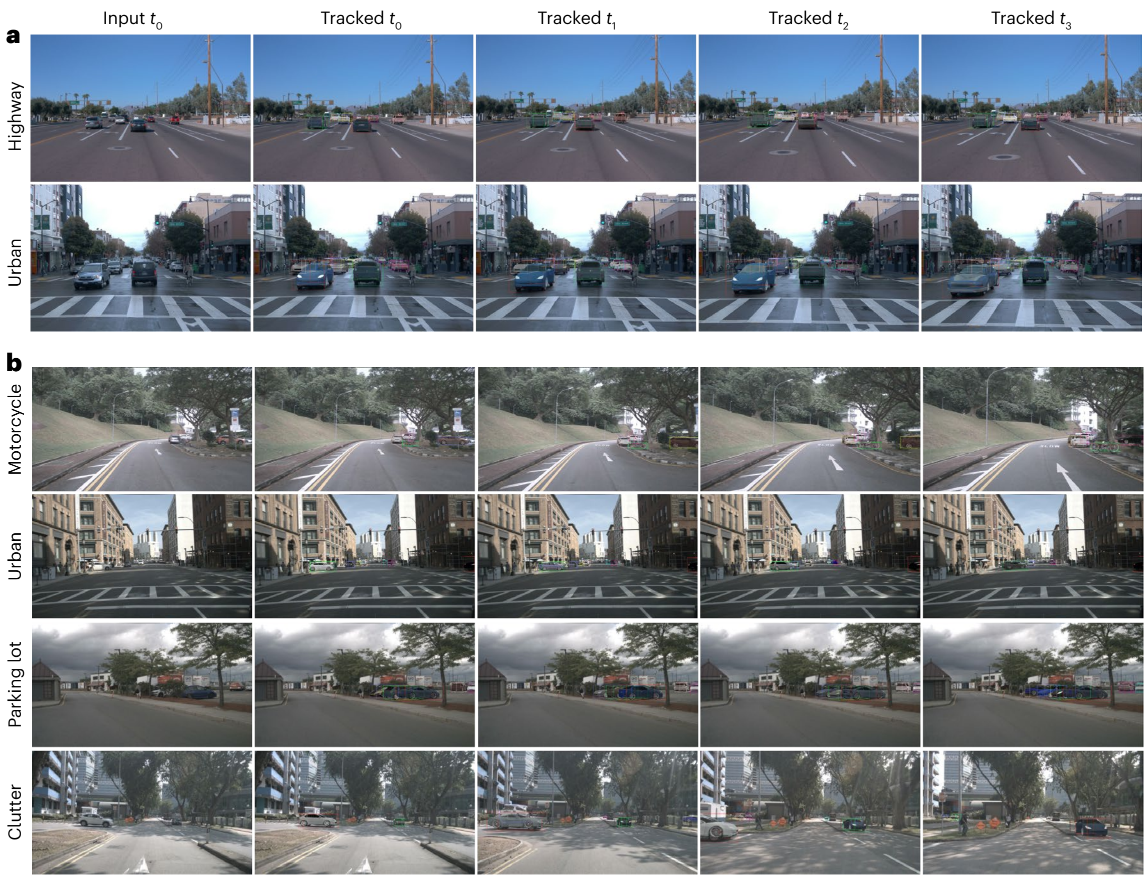


Fig. 3 | Generalizing multi-object tracking results. a,**b**, Tracking with inverse neural rendering using the Waymo open driving dataset[33] (**a**) and nuScenes[32] (**b**). The proposed method can generalize to unseen datasets. From left to right, we show observed images from diverse scenes at time step $k = 0$ and the optimized generated object and its 3D bounding boxes at time steps $k = 0, 1, 2$ and 3 overlaid over the input frame, which is faded for visibility. The colour of the bounding boxes for each object corresponds to the predicted tracklet ID. Even in such diverse scenarios, our method does not lose any tracks and performs robustly across all scenarios, although the dataset is unseen, validating that the approach generalizes. All tracked frames show images with objects in classes car and motorcycle (top row in **b**) overlaid over the ground-truth images, which are faded to show our rendered objects clearly. Panel **a** adapted with permission from ref. 33, CVPR.

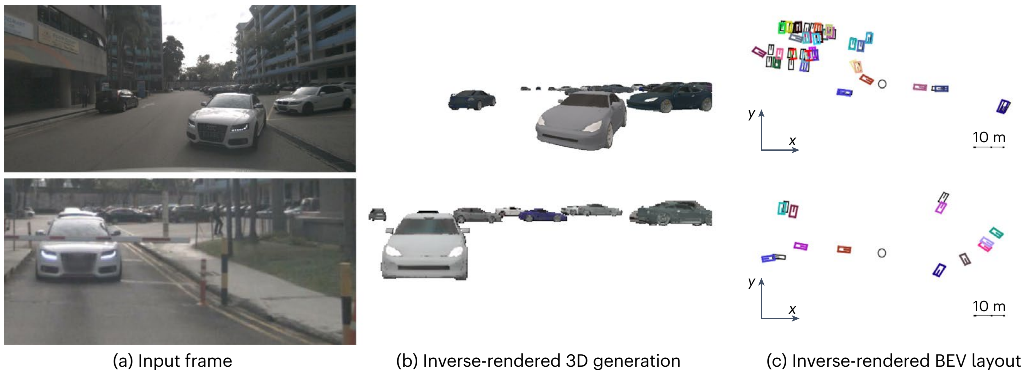

y
x
10 m
y
x
10 m
(a) Input frame
(b) Inverse-rendered 3D generation
(c) Inverse-rendered BEV layout

Fig. 4 | Layout generation. a, Images for two scenes observed by a single camera. **b**, Test-time optimized inverse rendered objects. **c**, BEV layouts of the scenes. In the BEV layout (a common representation for autonomous driving tasks), black boxes represent the ground truth and coloured boxes represent predicted BEV boxes. The bottom shows a zoomed-in region at 60 m distance. The complete set of tracked objects can be seen in the BEV layout, confirming that the method recovers the accurate appearance, shape, pose and size of the objects.

3D multi-object tracking from RGB cameras as an inverse test-time optimization problem over the latent space of pretrained 3D object representations that, when rendered, best represent object instances in a given input image. This approach to tracking also enables us to examine the reconstructed objects, reason about failure situations and resolve ambiguous cases. The rendering object layouts and loss function values provide interpretability 'for free'. We analyse the single-shot capabilities and the interpretability of our method using the image generated by our method during test-time optimization. Given a single image observation, our findings validate the potential for interpretable inverse rendering in safety-critical downstream tasks, such as 3D occupancy-based planning and free-space prediction in autonomous systems. This and other natural downstream extensions to our approach include cost-effective general offline auto-annotation and multi-sensor extensions (see also Supplementary Note 12). Trained only on synthetic data, we validate the generalization capabilities of our method by evaluating it on unseen automotive datasets. Our method achieves a 57.8% higher recall score compared with a learning-based method transferred to unseen datasets.

We investigate not only object detection with inverse rendering but broad, in-the-wild object class identification with conditional generation methods, thus unlocking analysis-by-synthesis in vision with generative neural rendering. By design, our approach allows a retrospective analysis of perception failure cases, specifically in scenarios where the association of tracked object instances in adjacent frames fails. Although it facilitates the inverse rendering, the iterative optimization in our method makes it slower than classical object-tracking methods based on feed-forward networks. We hope to address this limitation in the future by accelerating the forward and backward passes with adaptive level-of-detail rendering techniques. Although generative object models trained on synthetic data show wide dataset, domain and object class generalization capabilities in our work, we also see failure cases under adverse weather and lighting. Further improving the generative model on a wide set of real data, including surface materials and a more sophisticated rendering pipeline, will be a promising next step in improving the robustness and explainability of inverse rendering perception pipelines.

Methods

Overview

In this work, we leverage inverse rendering and generative object priors to infer and track 3D multi-object scenes by jointly optimizing object pose, geometry and appearance. Our approach focuses on scenarios where an accurate scene understanding is crucial for downstream decision-making, such as autonomous driving. Specifically, we cast object tracking as a test-time inverse rendering and synthesis problem that we solve by searching for latent object representations of all scene objects that match the image observations across time. We achieve this by optimizing a 3D object latent for each instance to the observed image frames with inverse rendering to minimize the visual distance between the rendered 3D representation and observed images. Therefore, we first structure a complex multi-object scene as a scene graph representation that describes individually generated object 3D models as its leaf nodes. This representation enables efficient gradient computation in both the object and camera coordinate systems.

Given a differentiable forward-rendering pipeline and observation image loss (Fig. 1b), we find the best set of generated objects for the scene with inverse rendering by minimizing the difference between the view generations of each observed object instance and the observation. Using a differentiable rasterized rendering pipeline, we directly unlock access to scene gradients, which is key to making our approach both efficient and interpretable.

We formulate a tracking pipeline based on the inverse rendered multi-object scenes in Fig. 1a to track objects through time with inverse neural rendering. We provide a detailed definition of our end-to-end tracking algorithm as Algorithm 1 in Supplementary Note 8.

Object generation

We employ an object-centric scene representation and model the underlying 3D scene for a frame observation as a composition of all object instances. To represent a large, diverse set of instances per class, we define each object instance o as a sample from a distribution O over all objects in a class:

$$(\mathbf{z}_S, \mathbf{z}_T) \sim O, \tag{1}$$

where O is a learned representation of a known prior object distribution. Here, the prior distribution is modelled by a differentiable generative 3D object model:

$$o_p = G(\mathbf{z}_{S,p}, \mathbf{z}_{T,p}), \tag{2}$$

that maps a latent embeddings $\mathbf{z}_{S,p}$ and $\mathbf{z}_{T,p}$ to an object instance o_p. In particular, the latent space comprises two disentangled spaces $\mathbf{z}_S \in \mathbb{R}^{d_S}$ and $\mathbf{z}_T \in \mathbb{R}^{d_T}$ for shape S and texture T.

Multi-object scene rendering

We model a multi-object scene as a differentiable scene graph[39] composed of affine transformations in the edges and object instances in the leaf nodes. The scene graph models object relationships and occlusions, including camera and scene objects, for differentiable coordinate system conversions to enable efficient gradient computation. The transformation in a render view for camera c is defined as

$$T_{c,p} = \mathrm{diag}\left(\frac{1}{s_p}\right) T_p T_c^{-1}, \tag{3}$$

where the factor s_p is a scaling factor along all axes to allow a shared object representation of a unified scale. This canonical object scale is necessary for representing objects of various sizes, independent of the learned prior on shape and texture. Further, the object-centric projection $P_{c,p} = K_c T_{c,p}$ is used to render the RGB image $I_{c,p} \in \mathcal{R}^{H\times W\times 3}$ and mask $M_{c,p} \in [0,1]^{H\times W}$ for each individual object/camera pair with the forward-rendering operator, which is a differentiable rasterization function R, as

$$I_p, M_p = R\left(G(\mathbf{z}_{S,p}, \mathbf{z}_{T,p}), P_{c,p}\right). \tag{4}$$

Individual rendered RGB images are ordered by object distance $|\mathbf{t}_{c,p}|$, such that $p = 1$ is the shortest distance to c. We define individual occlusion-aware alpha masks:

$$\boldsymbol{\gamma}_p = \max\left(\left(M_{c,p} - \sum_{q=1}^{p} M_{c,q}\right), 0^{H\times W}\right). \tag{5}$$

We then compose the final image of the multi-object scene $\hat{I}_c$ for all N_o objects by alpha-masking occluded pixels of occluded objects using the Hadamard product of the respective mask as

$$\hat{I}_c = \sum_{k=1}^{N_o} I_k \circ \boldsymbol{\gamma}_k, \tag{6}$$

which is, thus, a method for rendering and composing several generated objects into a single view image output corresponding to the camera model. This involves ordering objects by distance from the camera and sequentially rendering them while accounting for occlusions using masks. Instance masks are generated similarly using the same occlusion-aware composition process.

Inverse rendering and object generation

We invert the described differentiable rendering model defined in equation (4) by optimizing the set of all object representations in a given image I_c with gradient-based optimization. We assume that, initially, each object o_p is placed at a pose $\hat{T}_{c,p}$ and scaled with $\hat{s}_p$ near its underlying location. We represent object orientations in their respective Lie algebraic form $\mathfrak{so}(3)$. We sample an object embedding $\hat{\mathbf{z}}_{S,p}$ and $\hat{\mathbf{z}}_{T,p}$ in the respective latent embedding space.

For in-the-wild images, I_c is composed of sampled object instances, other objects and the scene background, which poses a challenge for the prior.

As our goal for tracking is to reconstruct all object instances of specific object classes, a naive ℓ_2 image matching objective of the form $\|I_c - \hat{I}_c\|_2$ is noisy and challenging to solve with vanilla stochastic gradient descent methods. To tackle this issue, we optimize visual similarity in the generated object regions inside $M_{I_c} = \sum^{N_o} M_{c,p}$ instead of the full image consisting of an RGB pixel loss and a learned perpetual similarity metric[40] (LPIPS) as

$$\mathcal{L}_{\mathrm{IR}} = \mathcal{L}_{\mathrm{RGB}} + \lambda \mathcal{L}_{\mathrm{perceptual}} = \|(I_c - \hat{I}_c) \circ \hat{M}_{I_c}\|_2 + \lambda_1 \,\mathrm{LPIPS}_{\mathrm{patch}}(I_c, \hat{I}_{c,p}, \hat{M}_{I_c}). \quad (7)$$

See Supplementary Note 5 for a detailed description of this loss component.

Instead of using vanilla gradient descent methods, we propose an alternating optimization schedule with distinct properties that includes aligning $\mathbf{z}_S$ before $\mathbf{z}_T$ to reduce the number of optimization steps. See Supplementary Note 6 for the details of this optimization schedule. Initial object proposals are realized at the bounding-box centroid locations of the upstream object detector. We initialize all shape and texture embeddings with the same fixed values inside the embedding space. We then apply two optimization steps solely based on colour using the described loss and freeze the colour for the joint optimization of the pose. We add shape and scale only in the last steps (Fig. 1b). We regularize out-of-distribution generations averaged across all objects with

$$\mathcal{L}_{\mathrm{embed}} = \|\alpha_T \mathbf{z}_T + (1-\alpha_T)\mathbf{z}_T^{\mathrm{avg}}\| + \|\alpha_S \mathbf{z}_S + (1-\alpha_S)\mathbf{z}_S^{avg}\|, \quad (8)$$

which minimizes a weighted distance for each dimension of $\mathbf{z}_S$ or $\mathbf{z}_T$ with respect to the average embedding. For optimization, we use the Adam optimizer[41]. The values $\mathbf{z}_S^{\mathrm{avg}}$ and $\mathbf{z}_T^{\mathrm{avg}}$ are computed as the mean of the embeddings for shape and texture of the prior distribution of G. The final loss objective sums the RGB, perceptual cost $\mathcal{L}_{\mathrm{IR}}$ and the regularization with the balancing factor $\alpha_T = 0.7$ and $\alpha_S = 0.7$ between the texture and shape instance, and respective mean embeddings in equation (8).

3D multi-object tracking by inverse rendering

Finally, we use the described inverse rendering approach to track objects in the proposed representation across video frames, which is illustrated in Fig. 1a. For readability, we omit p and the split of $\mathbf{z}$ into $\mathbf{z}_S$ and $\mathbf{z}_T$ in the following.

Common to tracking methods, we initialize observation $\mathbf{y}_k$ with a given initial 3D detection on image $I_{c,k}$, and we set object location $\mathbf{t}_k = [x, y, z]_k$ in all three dimensions and scale $s_k = \max(w_k, h_k, l_k)$ using the detected bounding-box width, length and height and heading ψ_k in frame k. We then find an optimal latent shape and texture representation $\mathbf{z}_k$ and a refined location and rotation of each object o with the inverse rendering pipeline for multi-object scenes. The resulting location, rotation and scale lead to the updated observation vector $\mathbf{y}_k = [\mathbf{t}_k, s_k, \psi_k]$. Although we are not tied to a specific dynamics model, we use a linear state-transition model $\mathbf{A}$ for the object state $x_k = [x, y, z, s, \psi, w, h, l, x', y', z']_k$, and a forward prediction using a Kalman filter[37], a vanilla approach in 3D object tracking[36]. The derivates x', y', z' are the respective velocities in all three dimensions of object k.

Matching between all objects in adjacent time steps is facilitated by computing the similarity across all available states. This includes the centroid distances and the 3D bounding-box intersection over union and places an additional focus on information about the appearance of the object and geometry embeddings ($\mathbf{z}_T$, $\mathbf{z}_S$), which improves the interpretability of such models. For all tracked states in $\mathbf{x}_k$, we follow the traditional Kalman filter match, update and predict design (Fig. 1). Supplementary Algorithm 1 and the derivations in Supplementary Note 8 provide a detailed pseudo-algorithm and mathematical derivation of all steps. Only embeddings are updated through an exponential moving average $\mathbf{z}_{k,\mathrm{EMA}}$ over the past observations of the object.

Implementation details

We describe the implementation of all design choices, including the composition of the loss term, the proposed optimization schedule, the heuristics applied in the matching stage of the multi-object tracker and details about the generative object model, in Supplementary Information.

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

The data used to generate the findings of this study are accessible through the respective public dataset download pages. The nuScenes dataset can be downloaded from https://www.nuscenes.org/nuscenes#download and access to the Waymo Open Dataset can be requested at https://waymo.com/open/download/. We have included instructions on how to run the supplementary code on the nuScenes dataset in the supplemental code repository. Source data are provided with this paper.

Code availability

The code used to generate the findings of this study is available via Zenodo at https://doi.org/10.5281/zenodo.15659175 (ref. 42) or GitHub at https://github.com/princeton-computational-imaging/INRTracker.

References

1. Spielberg, A. et al. Differentiable visual computing for inverse problems and machine learning. *Nat. Mach. Intell.* **5**, 1189–1199 (2023).
2. Gao, J. et al. Get3D: a generative model of high quality 3D textured shapes learned from images. In *Proc. Advances In Neural Information Processing Systems* Vol. 35 (eds Koyejo, S. et al.) 31841–31854 (Curran Associates, 2022).
3. Almalioglu, Y., Turan, M., Trigoni, N. & Markham, A. Deep learning-based robust positioning for all-weather autonomous driving. *Nat. Mach. Intell.* **4**, 749–760 (2022).
4. Zhang, B. et al. Segvit: semantic segmentation with plain vision transformers. In *Proc. Advances in Neural Information Processing Systems* Vol. 35 (eds Koyejo, S. et al.) 4971–4982 (Curran Associates, 2022).
5. Long, J., Shelhamer, E. & Darrell, T. Fully convolutional networks for semantic segmentation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Grauman, K. et al.) 3431–3440 (IEEE, 2015).
6. Huang, S. et al. Perspectivenet: 3D object detection from a single RGB image via perspective points. In *Proc. Advances in Neural Information Processing Systems* Vol. 32 (eds Wallach, H. et al.) 8905–8917 (Curran Associates, 2019).
7. Ren, S., He, K., Girshick, R. & Sun, J. Faster R-CNN: towards real-time object detection with region proposal networks. In *Proc. Advances in Neural Information Processing Systems* Vol. 28 (eds Cortes, C. et al.) 91–99 (Curran Associates, 2015).
8. Li, Y. et al. Unifying voxel-based representation with transformer for 3D object detection. In *Proc. Advances in Neural Information Processing Systems* Vol. 35 (eds Koyejo, S. et al.) 18442–18455 (Curran Associates, 2022).
9. Hu, H.-N. et al. Monocular quasi-dense 3D object tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**, 1992–2008 (2021).
10. Ke, L. et al. Prototypical cross-attention networks for multiple object tracking and segmentation. In *Proc. Advances in Neural Information Processing Systems* Vol. 34 (eds Ranzato, M. et al.) 1192–1203 (Curran Associates, 2021).
11. Wang, C. et al. Densefusion: 6D object pose estimation by iterative dense fusion. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Gupta, A. et al.) 3343–3352 (IEEE, 2019).

12. Pang, Z., Li, Z. & Wang, N. Simpletrack: understanding and rethinking 3D multi-object tracking. In *Proc. European Conference on Computer Vision (ECCV)* (eds Avidan, S. et al.) 680–696 (Springer, 2022).
13. Yin, T., Zhou, X. & Krahenbuhl, P. Center-based 3D object detection and tracking. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Forsyth, D. et al.) 11784–11793 (IEEE, 2021).
14. Kim, A., Ošep, A. & Leal-Taixé, L. Eagermot: 3D multi-object tracking via sensor fusion. In *Proc. 2021 IEEE International Conference on Robotics and Automation (ICRA)* (eds Howard, A. et al.) 11315–11321 (IEEE, 2021).
15. Liu, Z. et al. Bevfusion: multi-task multi-sensor fusion with unified bird's-eye view representation. In *Proc. 2023 IEEE International Conference on Robotics and Automation (ICRA)* (eds O'Malley, M. et al.) 2774–2781 (IEEE, 2023).
16. Weng, X., Wang, Y., Man, Y. & Kitani, K. M. GNN3DMOT: graph neural network for 3D multi-object tracking with 2D-3D multi-feature learning. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Liu, C. et al.) 6499–6508 (IEEE, 2020).
17. Chen, Y. et al. FocalFormer3D: focusing on hard instance for 3D object detection. In *Proc. IEEE/CVF International Conference on Computer Vision (ICCV)* (eds Agapito, L. et al.) 8394–8405 (IEEE, 2023).
18. Bai, X. et al. TransFusion: robust lidar-camera fusion for 3D object detection with transformers. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Dana, K. et al.) 1090–1099 (IEEE, 2022).
19. Wu, J. et al. Track to detect and segment: an online multi-object tracker. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Forsyth, D. et al.) 12352–12361 (IEEE, 2021).
20. Zhou, X., Koltun, V. & Krähenbühl, P. Tracking objects as points. In *Proc. European Conference on Computer Vision (ECCV)* (eds Bischof, H. et al.) 474–490 (Springer, 2020).
21. Marinello, N., Proesmans, M. & Van Gool, L. Triplettrack: 3D object tracking using triplet embeddings and LSTM. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Dana, K. et al.) 4500–4510 (IEEE, 2022).
22. Nguyen, P. et al. Multi-camera multiple 3D object tracking on the move for autonomous vehicles. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Dana, K. et al.) 2569–2578 (IEEE, 2022).
23. Gladkova, M. et al. Directtracker: 3D multi-object tracking using direct image alignment and photometric bundle adjustment. In *Proc. 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (eds Asfouret, T. et al.) 3777–3784 (IEEE, 2022).
24. Yang, J., Yu, E., Li, Z., Li, X. & Tao, W. QTrack: embracing quality clues for robust 3D multi-object tracking. In *Proc. 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (eds Laugier, C. et al.) 4904–4911 (IEEE, 2024).
25. Pang, Z. et al. Standing between past and future: spatio-temporal modeling for multi-camera 3D multi-object tracking. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Brown, M. S et al.) 17928–17938 (IEEE, 2023).
26. Wang, S., Liu, Y., Wang, T., Li, Y. & Zhang, X. Exploring object-centric temporal modeling for efficient multi-view 3D object detection. In *Proc. IEEE/CVF International Conference on Computer Vision (ICCV)* (eds Agapito, L. et al.) 3621–3631 (IEEE, 2023).
27. Chang, A. X. et al. ShapeNet: an information-rich 3D model repository. Preprint at https://arxiv.org/abs/1512.03012 (2015).
28. Munkberg, J. et al. Extracting triangular 3D models, materials, and lighting from images. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Dana, K. et al.) 8280–8290 (IEEE, 2022).
29. Park, J. J., Florence, P., Straub, J., Newcombe, R. & Lovegrove, S. DeepSDF: learning continuous signed distance functions for shape representation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Gupta, A. et al.) 165–174 (IEEE, 2019).
30. Mildenhall, B. et al. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* **65**, 99–106 (2021).
31. Shen, B. et al. Gina-3D: learning to generate implicit neural assets in the wild. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Brown, M. S. et al.) 4913–4926 (IEEE, 2023).
32. Caesar, H. et al. nuScenes: a multimodal dataset for autonomous driving. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Liu, C. et al.) 11621–11631 (IEEE, 2020).
33. Sun, P. et al. Scalability in perception for autonomous driving: Waymo Open Dataset. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Liu, C. et al.) 2446–2454 (IEEE, 2020).
34. Elezi, I., Yu, Z., Anandkumar, A., Leal-Taixe, L. & Alvarez, J. M. Not all labels are equal: rationalizing the labeling costs for training object detection. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Dana, K. et al.) 14492–14501 (IEEE, 2022).
35. Wang, B.-L., King, C.-T. & Chu, H.-K. A semi-automatic video labeling tool for autonomous driving based on multi-object detector and tracker. In *Proc. 2018 6th International Symposium on Computing and Networking (CANDAR)* (eds Bordim, J. et al.) 201–206 (IEEE, 2018).
36. Weng, X., Wang, J., Held, D. & Kitani, K. 3D multi-object tracking: a baseline and new evaluation metrics. In *Proc. 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (eds Zhang, H. et al.) 10359–10366 (IEEE, 2020).
37. Kalman, R. E. A new approach to linear filtering and prediction problems. *J. Basic Eng.* **82**, 35–45 (1960).
38. Bernardin, K., Elbs, A. & Stiefelhagen, R. Multiple object tracking performance metrics and evaluation in a smart room environment. In *Proc. 6th IEEE International Workshop on Visual Surveillance, in conjunction with ECCV* Vol. 90 (Citeseer, 2006).
39. Ost, J., Mannan, F., Thuerey, N., Knodt, J. & Heide, F. Neural scene graphs for dynamic scenes. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Forsyth, D. et al.) 2856–2865 (IEEE, 2021).
40. Zhang, R., Isola, P., Efros, A. A., Shechtman, E. & Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (eds Forsyth, D. et al.) 586–595 (IEEE, 2018).
41. Kingma, D. P. & Ba, J. Adam: a method for stochastic optimization. Preprint at https://arxiv.org/abs/1412.6980v5 (2017).
42. Ost, J., Banerjee, T., Bijelic, M. & Heide, F. Towards generalizable and interpretable 3D tracking with inverse neural rendering: data and code. *Zenodo* https://doi.org/10.5281/zenodo.15659175 (2025).

Acknowledgements

F.H. was supported by an NSF CAREER Award (Grant No. 2047359), a Packard Foundation Fellowship, a Sloan Research Fellowship, a Sony Young Faculty Award, a Project X Innovation Award, an Amazon Science Research Award and a Bosch Research Award.

https://doi.org/10.1038/s42256-025-01083-x

Author contributions

J.O. and F.H. conceived the method and experimental evaluation. J.O. and T.B. performed the experiments. J.O. and F.H. led the manuscript writing. J.O., T.B. and M.B. performed the analysis. F.H. supervised the project.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42256-025-01083-x.

Correspondence and requests for materials should be addressed to Felix Heide.

Peer review information *Nature Machine Intelligence* thanks Jinwei Ye and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

Reprints and permissions information is available at www.nature.com/reprints.