PointX: Multi-View Consistent Intrinsics by Querying Surface Points

Miguel Farinha[1,2,*] Sam Sartor[1] Iliyan Georgiev[1] Giuseppe Vecchio[1] Ronald Clark[2] Michael Fischer[1]

[1]Adobe Research [2]University of Oxford

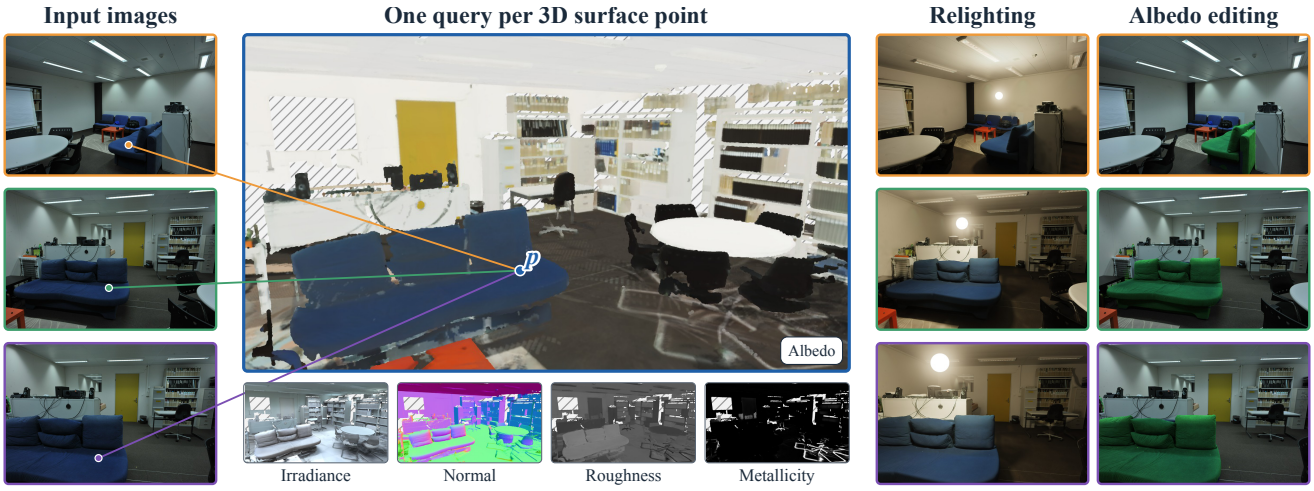

Figure 1. **PointX** is a feed-forward model for multi-view inverse rendering. From a set of images, it predicts the albedo, irradiance, roughness, metallicity and normal of any queried 3D surface point, here the points of a laser scan [74] registered to the images. Each surface point is decoded once, so the prediction is consistent across views by construction, enabling view-consistent relighting and editing.

Abstract

The appearance of a scene is determined by the albedo, roughness, metallicity and normal of every point on its surface, and by the light that reaches it, its irradiance. Recovering these intrinsics from images is ambiguous, and methods that predict them per pixel give the same surface point a different decomposition in every view. We introduce PointX, a simple feed-forward model that predicts intrinsics per 3D surface point instead. A multi-view transformer encodes a set of unposed images, and for any queried surface point, PointX pools the point's tokens across the views in which it appears and decodes the point's intrinsics once. Each point of a 3D reconstruction of the scene, such as a point cloud or a mesh, thus receives a single prediction, which is exactly consistent across views by construction and enables view-consistent relighting and editing. The same model also predicts per-view maps, from one image or many, by querying the point seen at every pixel. These maps agree across views up to the precision of the estimated geometry and correspondences. PointX is significantly more consistent than existing methods, while remaining competitive in accuracy from a single image and improving as views are added.
Project page: https://pointx-paper.github.io/

1. Introduction

Inverse rendering decomposes an image into the inputs of a renderer: the albedo, normal, roughness and metallicity of every surface point, and the light that reaches it, its irradiance. With these intrinsic properties [2], a captured scene can be relit, its materials edited, or new objects inserted into it. However, the decomposition is ambiguous, since the same pixel color has several valid explanations: a wall that appears red may be white under red light or red under white light. While recent methods [6, 25, 36, 40, 52, 67, 72, 103, 104] achieve accurate single-view predictions, a scene is usually captured in several images, or a video. The same surface point appears in several of them, yet per-image methods resolve the ambiguity in each image independently, so the point receives

*Work done during an internship at Adobe Research.

a different decomposition in every view. Applications using these intrinsics across multiple views then inherit this disagreement: a relit scene flickers from view to view, an edit made in one image does not carry over to the next, and materials texturing a reconstruction show seams.

One line of work addresses this inconsistency by attaching the intrinsics to a 3D representation of the scene, fitted per scene [23, 30, 56, 65, 106, 111] or predicted by a network [35, 53, 73, 109], and rendering them into every view, so that all views show the same underlying intrinsics. However, every view then inherits the errors of this representation, and per-scene fits take minutes to hours. A second line of work drops the 3D representation. Multi-view networks [14, 54, 98] compute the features of each view from all images through cross-view attention, and video models do the same across frames [20, 55, 85, 104]. Their predictions are more consistent than those of single-view networks, but each view is still decoded independently, so corresponding pixels of different views can still disagree. A third line of work keeps the per-view predictions and reconciles them afterwards, through a multi-view consensus at test time [11] or a per-scene optimization [41], which has to resolve the disagreement of separately predicted views. Thus far, consistency has either required an explicit 3D representation, been encouraged through attention and training, or been enforced after the predictions are made.

Our key insight is to predict intrinsic properties per surface point rather than per pixel. We introduce **PointX**, a feed-forward model that takes one or several unposed images of a static scene and predicts the intrinsics of a query surface point using evidence from all views in which it appears. A multi-view transformer, initialized from a feed-forward reconstruction model [90], first computes a feature map for each image. Given a query point in 3D or pixel space, we then find its correspondences across the input images and sample the feature maps at these locations. A 3D point is projected into each image with cameras and depth, estimated by a frozen reconstruction model [90] or taken from any other source, and a pixel can also be matched across the images with a dense matcher [66]. We pool these features into a single representation of the surface point, weighting each view by the confidence that the point is visible in it, and small MLP heads decode the point's intrinsic properties from it only once. The resulting prediction belongs to the point rather than to a view, making it consistent by construction. The same model can predict properties directly for the points of a reconstruction, without having to fuse separate per-view predictions, or produce per-view maps by querying every pixel, which then agree across views as far as their correspondences do. PointX also benefits directly from more accurate geometry and correspondences, such as those of a dense matcher, without retraining. In Fig. 1, PointX predicts the intrinsic properties of the points of a laser-scanned scene [74], so that a light placed once or a material edited once produces consistent results across the input views.

We evaluate PointX in three experiments. First, with the geometry of the scene given, we query the surface points of rendered scenes [70, 117] directly: PointX is competitive with or more accurate than baselines whose per-view predictions are averaged at each point, and has no disagreement between views to resolve. Second, we measure the consistency of per-view maps at exact correspondences on laser-scanned real scenes [74], where PointX is the most consistent. Third, we measure prediction accuracy on rendered scenes and real photographs [96]: PointX is competitive with the state of the art from a single image, and becomes more accurate with every added image. Ablations show that pooling the features of all views into a single representation per point brings this consistency at no cost in accuracy: decoding every pixel from its own features is equally accurate, but far less consistent.

In summary, our contributions are: (1) a formulation that predicts intrinsic properties once per surface point rather than per pixel, and is consistent across views by construction; (2) PointX, a feed-forward model that implements it from one or several unposed images, and serves both per-point and per-view prediction; and (3) an evaluation of intrinsics on surface points, and of their consistency at exact correspondences on real scenes.

2. Related Work

Intrinsic decomposition and inverse rendering. Intrinsic images [2, 46] separate an image into reflectance and shading. Early methods relied on physical assumptions and hand-designed priors [1, 24, 28, 86, 95]; later ones learned priors from human judgments [3, 64, 116], rendered scenes [49, 70, 117] and time-lapse video [50]. Inverse rendering widens the output to materials and lighting for whole indoor scenes [52, 75, 94, 117, 118]. Generative models have recently raised single-image accuracy [6, 8, 15, 36, 40, 67, 72, 85, 103], since they can resolve the ambiguity with a plausible sample. All of these methods predict single-view, per-pixel intrinsics.

Consistency across views and time. The disagreement between views has been addressed in three ways. Intrinsic video and multi-view intrinsic images tie the reflectance along correspondences with a penalty, solved per sequence [4, 16, 22, 42–44, 63, 100]; later work used such sequences as a training signal for single-image networks [48, 50, 102]. Reconstruction-based inverse rendering attaches materials and lighting to a 3D representation, optimized per scene [5, 23, 29, 30, 41, 56, 65, 81, 97, 99, 106, 107, 111, 112] or predicted by a network [35, 53, 73,

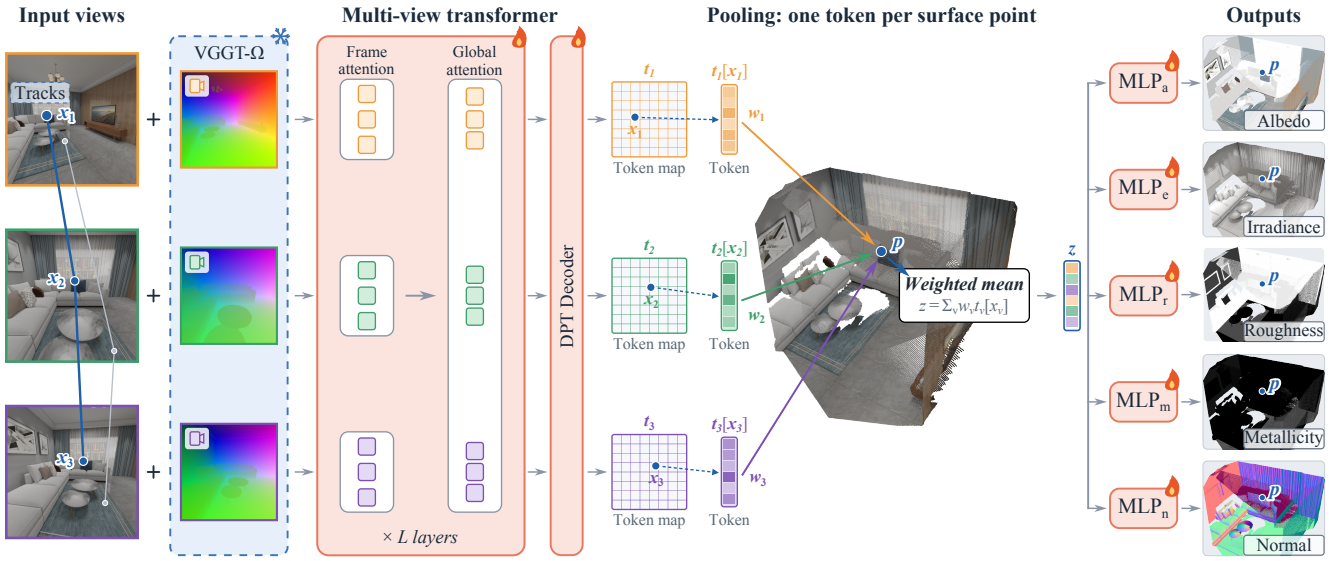

Figure 2. **PointX overview**. A multi-view transformer, conditioned on the geometry estimated by frozen VGGT-Ω [90], turns each view v into a token map t_v. For a queried surface point p, the tokens at its locations x_v in the views are pooled into a single token z by a confidence-weighted mean, and one MLP per property decodes the point's intrinsics from z, consistent across views by construction.

109, 110], and renders it into every view; this is consistent by design, but every view inherits its errors. Most recently, multi-view and video networks exchange information across views through attention [14, 20, 54, 55, 98], project predictions between frames [11], or fine-tune with a consistency objective [98]. They are more consistent than single-image networks, but the maps of each view are still decoded independently, so corresponding pixels can disagree. We instead decode the intrinsics once per surface point.

Feed-forward reconstruction, matching and tracking. Feed-forward reconstruction models [34, 37, 47, 57, 61, 82, 89, 90, 92, 93] predict depth and cameras for a set of images or a video with a transformer that attends across views. We use one as our multi-view transformer, and a frozen copy to estimate the geometry that conditions it. Dense matchers [17–19, 58, 66, 84, 87, 113] estimate a correspondence and its certainty for every pixel of an image pair without recovering 3D geometry, and point trackers [12, 13, 26, 32, 33, 45, 79] follow a pixel's surface point through a video. In our framework, a track is the pixel at which one surface point is visible in each view, obtained by projecting a 3D point with this geometry or by matching a pixel across the images, which is more precise and what our per-view maps rely on.

Pooling the observations of a point. Pooling per-view features at a 3D point is a staple of generalizable view synthesis and reconstruction: pixelNeRF [101] averages them, IBRNet [91] and NeuRay [59] weight them by learned visibility, MVSNeRF [7] fuses them in a cost volume, transformers attend along epipolar lines [83] or over all view tokens without an explicit point [27, 31, 71, 108], and GenRecon [73] conditions a generative mesh model on their statistics. Our pooling follows these works: a visibility-weighted mean of the tokens of a surface point, decoded into intrinsic properties. A second line lifts per-view features or labels into a 3D field and consolidates their disagreements there [21, 22, 38, 39, 62, 76, 88, 114, 115], at the cost of a per-scene optimization. We pool the tokens before decoding, so there are no per-view predictions to consolidate.

3. Method

We introduce PointX, illustrated in Fig. 2, a feed-forward transformer that takes a set of images of a scene and predicts the intrinsic properties of any queried surface point. Section 3.1 states the problem and how a surface point is located in the images, Sec. 3.2 describes the architecture, and Sec. 3.3 the training setup.

3.1. Querying Surface Points

Problem. Let $I_1, \dots, I_N \in \mathbb{R}^{3\times H\times W}$ be $N \geq 1$ images of a scene and $p \in \mathbb{R}^3$ a point on one of its surfaces. PointX predicts the intrinsic properties of p,

$$(a, e, r, m, n) = f\big(p;\, I_1, \dots, I_N\big), \tag{1}$$

where $a \in [0,1]^3$ is the albedo, $e \in \mathbb{R}^3_{\geq 0}$ the irradiance, $r \in [0,1]$ the roughness, $m \in [0,1]$ the metallicity and $n \in \mathbb{S}^2$ the normal. These are properties of 3D surface

points instead of images, and thus should be identical across views. We therefore let the surface point, rather than a pixel of an image, be the unit of prediction. Since f depends only on the point and the images, and on no reference view, every surface point receives a single prediction, consistent across views by construction.

Geometry. A 3D point is only meaningful relative to the geometry of the scene. We assume a camera and a depth map D_v for every view v. They can be given, for example by a scan or a multi-view reconstruction, or, for unposed images, estimated from the images themselves by a reconstruction model. Unprojecting D_v yields the point map X_v, the 3D point observed at each pixel of view v.

Tracks. To predict the properties of p, we first locate it in the input images. The *track* of p is the set of locations x_v at which it appears in the views, which we obtain by projecting p with the camera of each view v. Projection returns a location even where p is occluded, so each location x_v carries a confidence $c_v \in [0, 1]$ that p is visible in view v. We estimate it from the distance between p and the surface point imaged at x_v,

$$c_v = \frac{1}{1 + \delta_v/\tau}, \qquad \delta_v = \frac{\|X_v[x_v] - p\|}{d_v}, \tag{2}$$

where $X_v[x_v]$ is the point map sampled bilinearly at x_v, d_v is the depth of p in view v, and τ is a relative tolerance, which we set to 1%. If p lies behind the camera ($d_v \leq 0$) or does not project into the image, $c_v = 0$. Where p is visible, $\delta_v \approx 0$ and the confidence is close to one. The farther the imaged surface lies from p, the lower the confidence. PointX needs only the track of a query, so its locations can equally come from any other source of correspondences.

Pixel queries. A pixel query is a special case of Eq. (1): a pixel q of view i maps to a single surface point, $p = X_i[q]$. Querying every pixel of an image yields its intrinsic maps, while querying the points of a point cloud or the texels of a mesh yields them directly on that representation. The track of a pixel can be obtained by projection or directly in image space with a dense matcher such as RoMa-Ω [66]; c_v of Eq. (2), still computed with the point maps and cameras at x_v, is then multiplied by the dense matcher's confidence.

3.2. Architecture

PointX consists of three parts (Fig. 2): a multi-view transformer turns every image into a map of tokens, a pooling step reads one token per view along the track of the query and averages them into a single token, and MLP heads decode the intrinsic properties of the point from this token.

Input geometry. We run VGGT-Ω [90] (frozen) once on the input images to estimate a camera and a depth map with per-pixel confidence for every view. This geometry is an input of our multi-view transformer, but we first change the frame in which it is expressed: VGGT-Ω expresses its camera poses relative to the first view, whereas our model should treat all views symmetrically. We therefore express all geometry in a common frame, the *mean-frame*: its origin is the mean of the camera centers, its forward axis the normalized mean of the viewing directions, its horizontal axis the mean of the cameras' horizontal axes made orthogonal to the forward axis, and its vertical axis the cross product of the two. No view is privileged in this frame, and for a single view it coincides with the camera frame. The predicted normal n is expressed in the mean-frame as well.

Multi-view transformer. The transformer computes one feature map per view from all images jointly, so that the features of a view combine evidence from all views. We use the transformer of VGGT-Ω [90], initialized from its released weights and fine-tuned, which alternates self-attention within each view (frame attention) and across all views (global attention) [89]. Since a normal is decoded from a token pooled over several views, the features of every view must encode the pose of its camera in the mean-frame, which the image alone cannot provide. We therefore give every pixel of a view ten geometry channels, all expressed in the mean-frame: its viewing ray in 6D Plücker coordinates [80], the 3D point that VGGT-Ω estimates at that pixel, and the scalar log-confidence of this estimate. An additional, zero-initialized patch embedding embeds these channels, and its output is added to the patch embedding of the image, so that the pretrained features are unchanged at initialization. Further, VGGT-Ω marks its first view with separate camera and register tokens. We drop the camera tokens and use the same register tokens for every view, making our transformer, unlike the frozen VGGT-Ω that estimates its input geometry, equivariant to the order of views.

Token map. The transformer outputs feature maps at $1/16$ of the image resolution, too coarse for per-pixel detail, which a DPT decoder [68] upsamples to the full image resolution (see Sec. A.1). The *token map* t_v of view v then holds one token per pixel, built from two sources. The first is a linear projection of the upsampled feature at that pixel, which carries the multi-view evidence of the transformer. Inspired by D4RT [105], we add a linear projection of the 9×9 RGB window of I_v around the pixel. Both projections are shared by all views.

Pooling. To decode the intrinsics of p once, from all the views in which it is visible, rather than once per view, we

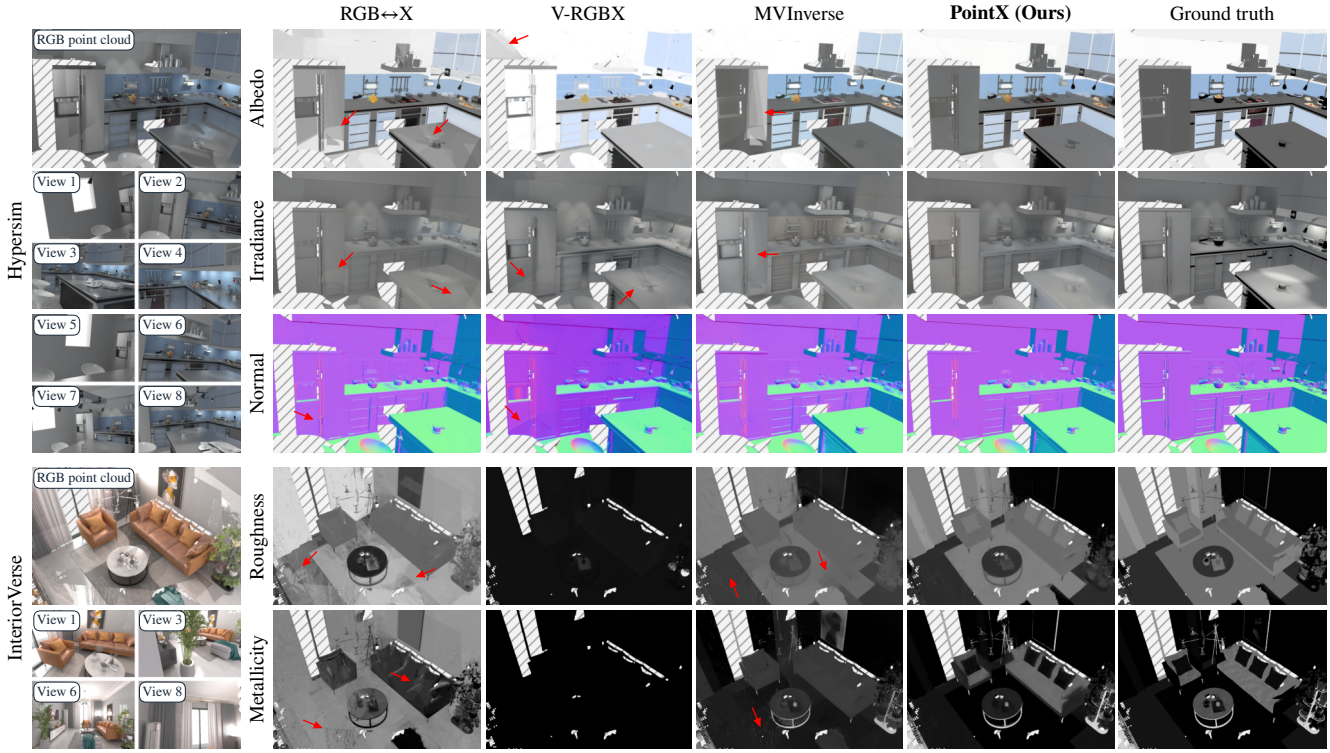

Figure 3. **Intrinsics of surface points.** Point clouds of a Hypersim [70] (top) and an InteriorVerse [117] (bottom) scene, built from the ground-truth depth of the input views and rendered from a new viewpoint (regions without ground truth are hatched). The baselines average their per-view maps over the views in which a point is visible; where these views disagree, the average shows seams and patches (red arrows). PointX predicts every point once and needs no averaging, yielding uniform surfaces.

merge the tokens along its track into a single token. Every view contributes the token at its location on the track, $t_v[x_v]$, sampled bilinearly, and we average these tokens with their confidence values along the track as weights,

$$z = \sum_v w_v \, t_v[x_v], \qquad w_v = \frac{c_v}{\sum_u c_u}. \tag{3}$$

A view in which p is occluded, or in which x_v falls on the background beside a thin structure, has a low confidence and contributes little to z. The weights have no learned parameters and depend only on the point and the views. For a pixel query, the queried view i has $\delta_i = 0$ and thus full confidence. The weighted mean is invariant to the order of the views and defined for any number of them, provided $\sum_u c_u > 0$; for a single image, z is simply the token at x_1.

Heads. Each intrinsic is decoded from z by an individual two-layer MLP, followed by a sigmoid for albedo, roughness and metallicity, a softplus for irradiance which is predicted directly as $\log(1+e)$, and a unit-length normalization for the normal. To obtain the normal map of a view, we rotate n from the mean-frame into that view's camera frame.

Consistency. Nothing in Eq. (3) refers to the view from which a point is queried. With a track obtained by projection, the locations, the weights and hence z depend only on the point, the geometry and the images. A surface point thus receives a single prediction from whichever view it is queried: its disagreement across views is zero by construction, without a consistency loss or fusion of per-view predictions. Per-view maps are consistent up to a small disagreement, because corresponding pixels of two views do not query exactly the same point. Even with exact geometry, the two pixels unproject to neighboring points, since each view samples the surface on its own pixel grid. Errors in the estimated depth and cameras move them further apart and tilt the normals. With a matcher, the two tracks also differ slightly, as each is matched from its own view.

3.3. Training

Since the ground truth of our datasets comes as per-view maps, we train with pixel queries. At each step, we sample query pixels in one view of every scene and supervise their intrinsics with the ground truth. The tracks of these queries are drawn at random from three sources: projection with the ground-truth or VGGT-Ω's geometry, and the dense matcher RoMa-Ω [66]. We apply an L_1 loss on albedo,

Table 1. **Intrinsics of surface points.** Accuracy of the per-point predictions, and disagreement between a method's per-view predictions before they are averaged; PointX predicts each point once, so its disagreement is zero by construction (not ranked). The first row applies the same protocol to the ground-truth maps, each rendered on its own pixel grid. PSNR_s: PSNR after a per-channel scale; RMSE_r: RMSE between two views relative to random pairs. Best, second and third best are highlighted. †Trained on the Hypersim test scenes, not ranked.

Method	Hypersim [70]							InteriorVerse [117]							
	Accuracy				Disagreement			Accuracy					Disagreement			
	Albedo		Irrad.	Normal	Albedo	Irrad.	Normal	Albedo		Normal	Rough.	Metal.	Albedo	Normal	Rough.	Metal.
	PSNR ↑	PSNR_s ↑	PSNR_s ↑	Mean° ↓	RMSE_r ↓	RMSE_r ↓	Mean° ↓	PSNR ↑	PSNR_s ↑	Mean° ↓	RMSE ↓	RMSE ↓	RMSE_r ↓	Mean° ↓	RMSE_r ↓	RMSE_r ↓
Ground-truth maps	43.47	43.53	42.58	0.7	0.089	0.145	3.1	45.12	45.14	0.6	0.006	0.008	0.093	3.1	0.083	0.173
Intrinsic Image Diffusion [40]	12.37	14.21	–	–	0.483	–	–	16.11	19.12	–	0.201	0.130	0.439	–	0.562	0.493
RGB↔X† [103]	24.09	24.89	24.17	8.0	0.196	0.289	8.3	9.57	18.96	12.3	0.304	0.407	0.499	18.1	0.582	0.837
DiffusionRenderer [55]	14.50	16.82	–	13.6	0.312	–	12.2	10.82	17.92	11.4	0.292	0.192	0.513	14.0	0.821	0.954
V-RGBX [20]	11.98	15.57	18.92	23.3	0.239	0.278	11.9	11.42	14.48	22.4	0.407	0.193	0.431	16.6	0.789	0.959
MVInverse [98]	18.62	26.01	19.22	8.2	0.141	0.302	6.4	18.41	19.33	7.8	0.152	0.111	0.218	8.8	0.241	0.316
RGB↔X + GeoID† [11]	23.62	24.58	–	–	0.186	–	–	10.89	18.99	–	0.266	0.419	0.481	–	0.553	0.815
UniD [85]	17.34	18.65	20.72	14.9	0.189	0.283	11.0	9.63	16.26	15.7	–	–	0.341	17.7	–	–
Marigold V2 [67]	21.16	22.64	–	8.4	0.209	–	5.9	11.57	17.47	8.7	–	–	0.382	10.5	–	–
MVID [14]	26.22	27.01	25.19	8.9	0.147	0.228	4.9	24.64	25.15	24.3	–	–	0.157	12.7	–	–
MVID-Ω [14]	17.72	19.53	19.34	11.5	0.172	0.252	5.0	19.32	21.03	25.4	–	–	0.174	9.0	–	–
PointX w/o pooling, maps averaged	20.33	22.78	22.66	8.6	0.136	0.212	4.1	25.41	25.89	5.7	0.083	0.071	0.158	4.8	0.144	0.244
PointX (Ours)	20.32	22.76	22.69	8.3	**0** by construction			25.50	25.97	5.2	0.083	0.072	**0** by construction			

roughness, metallicity and log-irradiance, the cosine distance on the normal, and a multi-scale gradient loss [51, 89] on queries sampled as 16×16 windows. As rendered frames are exposed individually, we bring the irradiance labels of a set of views to one exposure, so that all views learn the same irradiance. We neither compare the predictions of different views nor use visibility labels: the pooling of Eq. (3) has no learned parameters, so consistency holds by construction, not by learning. We train on rendered scenes of Hypersim [70], InteriorVerse [117] and three in-house datasets (1,400 in- and outdoor scenes), and on 1M real photographs pseudo-labeled by a single-image model (see Sec. A.3).

4. Experiments

We compare PointX with recent state-of-the-art methods in two settings. In Sec. 4.1, the geometry of the scene is given and we query its surface points directly. In Secs. 4.2 and 4.3, only the images are given: we query every pixel of every image and evaluate the consistency and the accuracy of the resulting per-view maps. Finally, we present a set of key ablations in Sec. 4.4 and discuss applications in Sec. 4.5.

Baselines. We compare with single-view methods [40, 67, 103], video models [20, 55, 85], the multi-view models MVInverse [98], MVID and MVID-Ω [14], and GeoID [11], a test-time multi-view consensus on RGB↔X [103]. All use their released weights, and the video and multi-view models process several images jointly. Methods that need posed images [73] or per-scene optimization [41] are not part of the main comparison. We evaluate GenRecon [73] and Intrinsic Image Fusion [41], on the task of Sec. 4.1 in Sec. C.2. We omit methods that target single objects [54] or have no released model [35].

Datasets and metrics. Every dataset is evaluated on sets of up to eight views of a scene. Hypersim [70] and InteriorVerse [117] are rendered and come with ground-truth intrinsics and depth, MAW [96] provides measured albedo for real indoor photographs, and ETH3D [74] provides real photographs with laser-scanned geometry. For accuracy, we report the PSNR of the albedo, as predicted and after fitting a per-channel scale to the ground truth (PSNR_s); the PSNR_s of the irradiance, which is defined only up to the exposure of an image; the mean angular error of the normals; and the RMSE of roughness and metallicity. The disagreement between two views is measured at corresponding points: we report the mean angle between their normals and, for the other properties, the RMSE between corresponding points divided by the RMSE between randomly paired ones (RMSE_r, see appendix), so that a constant map is not rewarded. An RMSE_r of 0 means that the views agree, and an RMSE_r of 1 that they agree no better than at random.

4.1. Intrinsics of Surface Points

We first evaluate the task of Eq. (1): predicting the intrinsics of a surface point from all the images in which it is visible.

Protocol. We use Hypersim and InteriorVerse, whose ground-truth depth gives the surface of every scene. InteriorVerse provides no camera poses, so we register its views from the depth maps (see Sec. B). For every scene, we unproject the depth maps and sample surface points that are visible from at least two views. We define the true value of a point to be the median of the ground-truth maps at its projections. PointX is queried once per point, with its track and the confidence of Eq. (2) obtained by projection with the ground-truth geometry. The baselines predict one map per view, so we read these maps at the projections of the point and take their unweighted mean over the views in which it

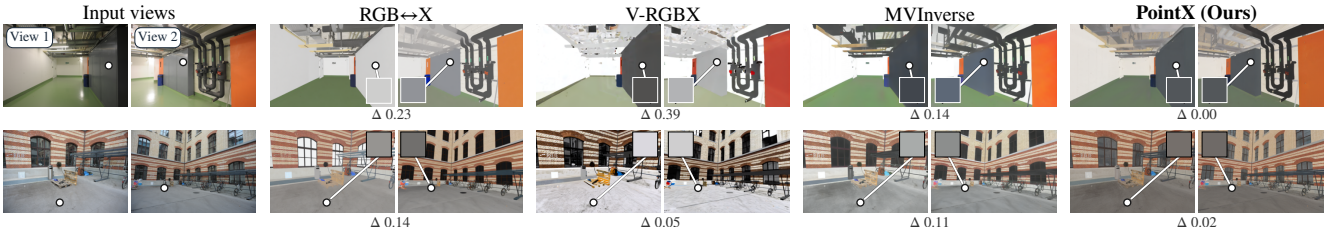

Figure 4. **Consistency of per-view maps on ETH3D [74].** Albedo predictions for two views of a scene (PointX: RoMa-Ω tracks). The marked surface point is the same in both views; the insets show the value each method predicts for it, and Δ their mean absolute difference.

is visible, using the same geometry; a weighted mean or a median gives similar results (see Sec. C.1).

Results. Table 1 shows that PointX is the most accurate on every property of InteriorVerse and competitive with the most accurate baselines on Hypersim. Averaging the maps that PointX predicts for each view without pooling gives nearly the same accuracy, but at the cost of consistency. Every baseline assigns different values to the same surface point in different views, so its predictions must be fused before they can be attached to the scene. As observed in Fig. 3, where there is disagreement, seams appear on the baselines' intrinsic point clouds. PointX decodes the intrinsics of every point once, which yields uniform surfaces.

4.2. Consistency of Per-View Maps

The geometry of a scene is often unknown, and many applications expect one map per image. We hence measure next whether a method's per-view maps agree at ground-truth correspondences.

Protocol. We apply the reprojection protocol of MVInverse [98] to the real photographs of ETH3D, whose laser scans give exact correspondences. For every pair of views, the pixels of one view are projected into the other with the ground-truth depth and kept where the depth of the other view confirms that they are visible. The maps of the two views are compared at these corresponding pixels, with the normals rotated into the world frame by the ground-truth cameras. Additional results are in the appendix.

Results. Table 2 shows that PointX is the most consistent on all five properties with the tracks of a dense matcher, RoMa-Ω. In Fig. 4, the marked surface point looks different in the two views, and the baselines assign it a different albedo in each view, whereas PointX's predictions look the most similar and have the lowest difference. The video and multi-view baselines process the views jointly but still decode each view separately, which does not ensure consistency. In PointX, corresponding pixels are decoded from tokens pooled along nearly identical tracks, so their predictions can differ only as much as their tracks do (Sec. 3.2). Its consistency therefore improves with the precision of the tracks, without retraining. With tracks projected through the coarser geometry of VGGT-Ω, it already leads on irradiance, normals and roughness. The matcher extends the lead to all four properties. When ground-truth geometry is available, its exact tracks make PointX highly consistent, with by far the lowest disagreement on every property.

Table 2. **Consistency of per-view maps on ETH3D [74].** Disagreement between the maps of two views at ground-truth correspondences. Gray: ground-truth geometry at test time, not ranked. Best, second and third best are highlighted.

Method	Track Source	Albedo RMSE$_r$ ↓	Irrad. RMSE$_r$ ↓	Normal Mean° ↓	Rough. RMSE$_r$ ↓	Metal. RMSE$_r$ ↓
Single-view methods						
Intrinsic Image Diffusion [40]	–	0.481	–	–	0.759	0.581
RGB↔X [103]	–	0.378	0.448	13.5	0.562	0.799
Marigold V2 [67]	–	0.210	–	7.4	–	–
Video & Multi-view methods						
DiffusionRenderer [55]	–	0.291	–	11.6	0.568	0.733
V-RGBX [20]	–	0.216	0.302	10.4	0.334	0.870
MVInverse [98]	–	0.161	0.318	9.6	0.382	0.270
RGB↔X + GeoID [11]	–	0.374	–	–	0.556	0.783
UniD [85]	–	0.200	0.327	11.1	–	–
MVID [14]	–	0.202	0.335	11.0	–	–
MVID-Ω [14]	–	0.181	0.303	6.8	–	–
PointX (Ours)	VGGT-Ω [90]	0.177	0.214	4.6	0.217	0.353
	RoMa-Ω [66]	0.095	0.146	2.9	0.136	0.222
	Ground truth	0.058	0.079	1.0	0.078	0.125

4.3. Accuracy of Per-View Maps

We now evaluate the accuracy of per-view maps, predicted from one image and from several images of a scene.

Protocol. We evaluate on InteriorVerse, Hypersim and the real photographs of MAW, and additionally report the LPIPS of the albedo. On MAW, we follow the protocol of Wu et al. [96] and report the intensity, chromaticity and WHDR errors of the albedo.

Results. As shown in Tab. 3 (top), from a single image, PointX is the most accurate on every property of InteriorVerse, and ranks among the top three on most metrics of Hypersim and of MAW, real photographs on which no

Table 3. **Accuracy of per-view maps.** Top: from one image. Bottom: video and multi-view models on several images (PointX: RoMa-Ω tracks). MAW intensity $\times 10^2$. Best, second and third best of each block are highlighted. †Trained on the Hypersim test scenes, not ranked.

Method	InteriorVerse [117]						Hypersim [70]					MAW [96]		
	Albedo			Normal	Rough.	Metal.	Albedo			Irrad.	Normal	Albedo		
	PSNR ↑	PSNR_s ↑	LPIPS ↓	Mean° ↓	RMSE ↓	RMSE ↓	PSNR ↑	PSNR_s ↑	LPIPS ↓	PSNR_s ↑	Mean° ↓	Int. ↓	Chrom. ↓	WHDR ↓
One image														
Intrinsic Image Diffusion [40]	15.84	19.02	0.163	–	0.211	0.141	11.67	14.71	0.455	–	–	1.36	4.16	0.212
RGB↔X† [103]	9.78	18.22	0.320	19.3	0.317	0.412	22.92	24.07	0.143	23.90	11.5	0.70	4.08	0.219
DiffusionRenderer [55]	13.36	18.74	0.281	10.8	0.281	0.199	12.64	16.84	0.385	–	15.1	0.42	3.44	0.170
V-RGBX [20]	9.78	15.57	0.434	20.7	0.566	0.197	12.57	16.06	0.393	17.88	22.9	0.52	4.67	0.263
MVInverse [98]	18.02	19.00	0.177	8.7	0.155	0.117	17.05	24.28	0.240	19.17	10.6	0.34	3.93	0.223
UniD [85]	8.74	14.28	0.351	12.4	–	–	17.11	18.75	0.352	20.08	14.3	0.36	4.77	0.229
Marigold V2 [67]	11.84	17.27	0.222	9.7	–	–	19.90	21.92	0.220	–	10.3	0.24	3.07	0.233
MVID [14]	23.54	24.31	0.121	11.3	–	–	25.06	25.83	0.235	23.91	10.5	0.42	3.71	0.290
MVID-Ω [14]	18.93	20.57	0.170	11.4	–	–	17.21	19.45	0.372	19.39	13.3	0.37	3.67	0.203
PointX (Ours)	24.09	24.87	0.101	6.3	0.100	0.085	18.87	21.25	0.192	21.25	12.3	0.29	3.10	0.208
Several images of a scene (up to eight)														
DiffusionRenderer [55]	10.92	17.51	0.365	13.1	0.293	0.196	13.85	16.76	0.379	–	16.5	0.44	3.56	0.186
V-RGBX [20]	11.50	14.41	0.389	24.1	0.395	0.197	11.80	16.03	0.397	18.23	24.8	0.63	4.94	0.300
MVInverse [98]	18.50	19.39	0.173	8.6	0.152	0.115	17.23	24.61	0.237	19.19	10.4	0.31	3.82	0.233
RGB↔X + GeoID† [11]	11.16	18.27	0.279	–	0.277	0.423	21.99	23.59	0.151	–	–	0.78	3.90	0.216
UniD [85]	9.99	16.38	0.318	17.9	–	–	16.51	18.44	0.426	19.76	17.3	0.41	4.76	0.237
MVID [14]	24.23	24.89	0.118	25.8	–	–	25.60	26.22	0.230	24.14	10.7	0.41	3.58	0.295
MVID-Ω [14]	19.15	20.81	0.168	26.4	–	–	17.40	19.64	0.370	19.46	13.8	0.35	3.54	0.208
PointX (Ours)	25.13	25.71	0.103	5.7	0.084	0.077	19.65	22.02	0.191	21.42	10.7	0.30	2.87	0.214

method was trained. Table 3 (bottom) further shows that, when given several images of a scene, PointX becomes more accurate on all three datasets and leads on most metrics, whereas MVInverse improves only slightly and the video models do not improve consistently. Figure 5 shows how the accuracy of a view changes as more views of its scene are added. Every property predicted by PointX becomes more accurate, while MVInverse gains considerably less from the same images. Most of this gain remains when every pixel is decoded from its own token, without pooling (dashed line): it comes from our multi-view transformer, which gathers more evidence from the additional views. The same trend holds on the other datasets (see appendix).

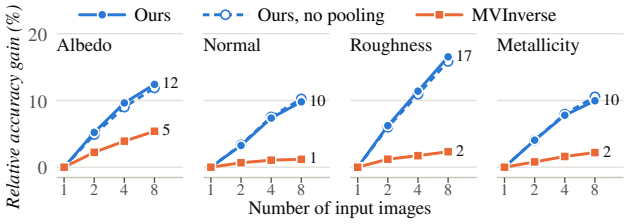

Figure 5. **Accuracy with more input images.** Reduction of the error on InteriorVerse [117] relative to a single image (RMSE; mean angle for normals). PointX uses RoMa-Ω tracks.

4.4. Ablation Studies

Pooling. Removing the pooling step and decoding every pixel from its own token is as accurate, but makes the views disagree notably more, as shown in Tab. 4a. Pooling brings consistency at no cost in accuracy.

Table 4. **Ablation studies** on per-view maps. (a) Without one component, from one and eight images (*not retrained). (b) With and without the confidence, from eight images. Best in bold.

(a) Without one component (RoMa-Ω tracks)

	Accuracy (InteriorVerse [117])				Disagreement (ETH3D [74])			
	Albedo PSNR ↑		Normal Mean° ↓		Albedo	Normal	Rough.	Metal.
	1 image	8 images	1 image	8 images	RMSE_r ↓	Mean° ↓	RMSE_r ↓	RMSE_r ↓
PointX (Ours)	**24.09**	**25.13**	6.29	5.68	**0.095**	**2.91**	**0.136**	**0.222**
w/o pooling*	**24.09**	25.12	6.29	**5.65**	0.141	5.24	0.256	0.355
w/o input geometry	22.92	23.83	7.53	7.10	0.098	3.09	0.149	0.278
w/o global attention	23.90	24.31	**6.22**	6.13	0.118	3.92	0.248	0.316

(b) Confidence of Eq. (2), by source of tracks

	Accuracy (Hypersim [70], 8 images)				Disagreement (ETH3D [74])			
	Albedo PSNR ↑		Normal Mean° ↓		Albedo RMSE_r ↓		Normal Mean° ↓	
	with	without	with	without	with	without	with	without
Ground truth	**20.52**	18.01	**9.23**	12.26	0.058	0.058	0.99	0.99
VGGT-Ω [90]	**19.49**	16.99	**10.60**	14.87	**0.177**	0.245	**4.63**	5.31
RoMa-Ω [66]	**20.37**	19.81	**9.56**	10.27	**0.095**	0.096	2.91	**1.87**

Input geometry. Without the geometry channels, we observe in Tab. 4a that every property becomes less accurate and less consistent. The normals degrade the most: they are expressed in the mean-frame, so the model must infer the orientation of each camera from the images alone, and any error in it tilts all the normals of that view (see Sec. C.5).

Global attention. With global attention restricted to each view, the views interact only through pooling. In Tab. 4a, this variant remains as accurate from one image, but improves far less with more images, and its maps disagree more. Encoding the views jointly thus turns additional views into accuracy gains (Fig. 5) and aligns the tokens of a surface point across views, which pooling alone does not.

Confidence. The confidence of Eq. (2) down-weights the views in which a point is occluded. As observed in Tab. 4b, this improves accuracy with every source of tracks. The gain is largest with projected tracks, where nothing else rejects the occluded views, and smaller with RoMa-Ω, whose own confidence already rejects most of them.

Runtime. On an A100 GPU, PointX predicts all maps of eight 512×336 images in 1.6 s with ground-truth tracks, 2.0 s with VGGT-Ω tracks and 5.8 s with RoMa-Ω tracks.

4.5. Applications

Since PointX predicts one set of intrinsics per surface point, its predictions attach directly to a 3D representation of the scene, with no per-view maps to fuse. In Fig. 1, we query every point of the laser scan of an ETH3D scene once to obtain an intrinsic point cloud. A new light or an albedo edit applied in 3D then changes every photograph consistently, since each surface point is modified once (see Sec. D). In Fig. 6, we instead query one point per texel of a mesh, which bakes the predictions into its textures. The result is a standard asset for physically based rendering: a path tracer renders it under a new light with the predicted materials, and an inserted chrome ball reflects the textured scene.



Figure 6. **Texturing, relighting and object insertion.** The mesh of an ETH3D [74] scene, reconstructed from its laser scans and textured by one PointX query per texel, rendered from the camera of the input photograph: its albedo texture, the mesh path traced under a new light with the predicted albedo, roughness, metallicity and normals, and a chrome ball inserted into the relit scene.

5. Conclusion

We introduce PointX, a simple feed-forward model that predicts the intrinsic properties of any queried surface point from one or more images of a scene. Our core insight is that intrinsics belong to the surface points of a scene, not to the pixels of an image. PointX therefore pools the evidence of all views in which a point is visible into a single token and decodes its intrinsics once, making each point's prediction consistent across views by construction. Its per-view maps achieve state-of-the-art consistency on real scenes, and their accuracy, competitive from a single image, improves with every added view to lead on most metrics. PointX also predicts the intrinsics of point clouds and meshes, for view-consistent relighting, editing and object insertion.

LLM Usage Statement

We used LLMs to revise parts of the text, and to check grammar and improve the flow of writing. The authors edited the text, verified its content, and take full responsibility for it. The list of papers to discuss was compiled by the authors together with LLMs. The authors checked and manually entered all references. The research direction and ideas came from the authors. Under the authors' direction, LLMs helped implement the model, training and evaluation code, and set up the baselines and benchmarks from their public repositories. LLMs were used to check that the method as described in the paper matches the codebase. The authors verified all results and claims.

References

[1] Jonathan T Barron and Jitendra Malik. Shape, illumination, and reflectance from shading. *IEEE transactions on pattern analysis and machine intelligence*, 37(8):1670–1687, 2014. 2

[2] Harry Barrow, J Tenenbaum, A Hanson, and E Riseman. Recovering intrinsic scene characteristics. *Comput. vis. syst*, 2(3-26):2, 1978. 1, 2

[3] Sean Bell, Kavita Bala, and Noah Snavely. Intrinsic images in the wild. *ACM Transactions on Graphics (TOG)*, 33(4): 1–12, 2014. 2

[4] Nicolas Bonneel, Kalyan Sunkavalli, James Tompkin, Deqing Sun, Sylvain Paris, and Hanspeter Pfister. Interactive intrinsic video editing. *ACM Transactions on Graphics (TOG)*, 33(6):1–10, 2014. 2

[5] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12684–12694, 2021. 2

[6] Chris Careaga and Yağız Aksoy. Colorful diffuse intrinsic image decomposition in the wild. *ACM Transactions on Graphics (ToG)*, 43(6):1–12, 2024. 1, 2

[7] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pages 14104–14113. IEEE, 2021. 3

[8] Xi Chen, Sida Peng, Dongchen Yang, Yuan Liu, Bowen Pan, Chengfei Lv, and Xiaowei Zhou. Intrinsicanything: Learning diffusion priors for inverse rendering under unknown illumination. In *European Conference on Computer Vision*, pages 450–467. Springer, 2024. 2

[9] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992. 17

[10] Robert L Cook and Kenneth E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics (ToG)*, 1(1):7–24, 1982. 23

[11] Alara Dirik and Stefanos Zafeiriou. Geo-id: Test-time geometric consensus for cross-view consistent intrinsics. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2026. 2, 3, 6, 7, 8, 17, 21

[12] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adria Recasens, Lucas Smaira, Yusuf Aytar, Joao Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 35:13610–13626, 2022. 3

[13] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. Tapir: Tracking any point with per-frame initialization and temporal refinement. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10027–10038. IEEE, 2023. 3

[14] Kang Du, Duotun Wang, Wanling Li, Yirui Guan, and Zeyu Wang. MVID: Feed-forward multi-view intrinsic image decomposition. In *SIGGRAPH Asia 2026 Conference Papers*, New York, NY, USA, 2026. Association for Computing Machinery. arXiv:2512.23667. 2, 3, 6, 7, 8, 16, 17, 21

[15] Xiaodan Du, Nicholas Kolkin, Greg Shakhnarovich, and Anand Bhattad. Generative models: What do they know? do they know things? let's find out! *arXiv preprint arXiv:2311.17137*, 2023. 2

[16] Sylvain Duchêne, Clement Riant, Gaurav Chaurasia, Jorge Lopez-Moreno, Pierre-Yves Laffont, Stefan Popov, Adrien Bousseau, and George Drettakis. Multi-view intrinsic images of outdoors scenes with an application to relighting. *ACM Transactions on Graphics*, page 16, 2015. 2

[17] Johan Edstedt, Ioannis Athanasiadis, Mårten Wadenbäck, and Michael Felsberg. DKM: Dense kernelized feature matching for geometry estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2023. 3

[18] Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. Roma: Robust dense feature matching. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19790–19800. IEEE, 2024.

[19] Johan Edstedt, David Nördstrom, Yushan Zhang, Georg Bökman, Jonathan Astermark, Viktor Larsson, Anders Heyden, Fredrik Kahl, Mårten Wadenbäck, and Michael Felsberg. Roma v2: Harder better faster denser feature matching. In *European Conference on Computer Vision*, pages 664–682. Springer, 2026. 3

[20] Ye Fang, Tong Wu, Valentin Deschaintre, Duygu Ceylan, Iliyan Georgiev, Chun-Hao Paul Huang, Yiwei Hu, Xuelin Chen, and Tuanfeng Yang Wang. V-rgbx: Video editing with accurate controls over intrinsic properties. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23182–23192, 2026. 2, 3, 6, 7, 8, 16, 17, 21

[21] Michael Fischer, Zhengqin Li, Thu Nguyen-Phuoc, Aljaz Bozic, Zhao Dong, Carl Marshall, and Tobias Ritschel. Nerf analogies: Example-based visual attribute transfer for nerfs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4640–4650, 2024. 3

[22] Michael Fischer, Iliyan Georgiev, Thibault Groueix, Vladimir G Kim, Tobias Ritschel, and Valentin Deschaintre. Sama: Material-aware 3d selection and segmentation. In *2026 International Conference on 3D Vision (3DV)*, pages 1812–1822. IEEE, 2026. 2, 3

[23] Jian Gao, Chun Gu, Youtian Lin, Zhihao Li, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3d gaussians: Realistic point cloud relighting with brdf decomposition and ray tracing. In *European conference on computer vision*, pages 73–89. Springer, 2024. 2

[24] Roger Grosse, Micah K Johnson, Edward H Adelson, and William T Freeman. Ground truth dataset and baseline evaluations for intrinsic image algorithms. In *2009 IEEE 12th international conference on computer vision*, pages 2335–2342. Ieee, 2009. 2

[25] Zheng Gu, Rui Huang, Xilu Zhang, Jingbo Zhang, Min Lu, Zhida Sun, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Intrinsync: Joint intrinsic decomposition and reciprocal rendering. *arXiv preprint arXiv:2610.11138*, 2026. 1

[26] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *ECCV*, 2022. 3

[27] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. In *International Conference on Learning Representations*, pages 50678–50702, 2024. 3

[28] Berthold KP Horn. Determining lightness from an image. *Computer graphics and image processing*, 3(4):277–299, 1974. 2

[29] Yingwenqi Jiang, Jiadong Tu, Yuan Liu, Xifeng Gao, Xiaoxiao Long, Wenping Wang, and Yuexin Ma. Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5322–5332. IEEE, 2024. 2

[30] Haian Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. Tensoir: Tensorial inverse rendering. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 165–174. IEEE, 2023. 2

[31] Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, Sai Bi, Tianyuan Zhang, Fujun Luan, Noah Snavely, and Zexiang Xu. Lvsm: A large view synthesis model with minimal 3d inductive bias. In *The Thirteenth International Conference on Learning Representations*, 2025. 3

[32] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In *European conference on computer vision*, pages 18–35. Springer, 2024. 3

[33] Nikita Karaev, Yuri Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labeling real videos. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1–10. IEEE, 2025. 3

[34] Jay Karhade, Nikhil Keetha, Yuchen Zhang, Tanisha Gupta, Akash Sharma, Sebastian Scherer, and Deva Ramanan. Any4d: Unified feed-forward metric 4d reconstruction. *arXiv preprint arXiv:2512.10935*, 2025. 3

[35] Polina Karpikova, Wenjing Bian, Haofei Xu, Hendrik Lensch, and Andreas Geiger. Invsplat: Inverse feed-forward scene splatting. *arXiv preprint arXiv:2607.02301*, 2026. 2, 6

[36] Bingxin Ke, Kevin Qu, Tianfu Wang, Nando Metzger, Shengyu Huang, Bo Li, Anton Obukhov, and Konrad Schindler. Marigold: Affordable adaptation of diffusion-based image generators for image analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025. 1, 2

[37] Nikhil Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zauss, Ethan Weber, Nelson Antunes, Jonathon Luiten, Manuel Lopez-Antequera, Samuel Rota Bulò, Christian Richardt, Deva Ramanan, Sebastian Scherer, and Peter Kontschieder. MapAnything: Universal feed-forward metric 3D reconstruction. In *International Conference on 3D Vision (3DV)*. IEEE, 2026. 3

[38] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19672–19682. IEEE, 2023. 3

[39] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing nerf for editing via feature field distillation. *Advances in neural information processing systems*, 35:23311–23330, 2022. 3

[40] Peter Kocsis, Vincent Sitzmann, and Matthias Nießner. Intrinsic image diffusion for indoor single-view material estimation. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 5198–5208, 2024. 1, 2, 6, 7, 8, 17, 21

[41] Peter Kocsis, Lukas Höllein, and Matthias Nießner. Intrinsic image fusion for multi-view 3d material reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22571–22580, 2026. 2, 6, 19, 20

[42] Naejin Kong, Peter V Gehler, and Michael J Black. Intrinsic video. In *European Conference on Computer Vision*, pages 360–375. Springer, 2014. 2

[43] Pierre-Yves Laffont, Adrien Bousseau, and George Drettakis. Rich intrinsic image decomposition of outdoor scenes from multiple views. *IEEE transactions on visualization and computer graphics*, 19(2):210–224, 2012.

[44] Pierre-Yves Laffont, Adrien Bousseau, Sylvain Paris, Frédo Durand, and George Drettakis. Coherent intrinsic images from photo collections. *ACM Transactions on Graphics*, 31 (6), 2012. 2

[45] Zihang Lai, Eldar Insafutdinov, Edgar Sucar, and Andrea Vedaldi. Cowtracker: Tracking by warping instead of correlation. *arXiv preprint arXiv:2602.04877*, 2026. 3

[46] Edwin H Land and John J McCann. Lightness and retinex theory. *Journal of the Optical society of America*, 61(1): 1–11, 1971. 2

[47] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European conference on computer vision*, pages 71–91. Springer, 2024. 3

[48] Louis Lettry, Kenneth Vanhoey, and Luc Van Gool. Unsupervised deep single-image intrinsic decomposition using illumination-varying image sequences. In *Computer graphics forum*, pages 409–419. Wiley Online Library, 2018. 2

[49] Zhengqi Li and Noah Snavely. Cgintrinsics: Better intrinsic image decomposition through physically-based rendering. In *European conference on computer vision*, pages 381–399. Springer, 2018. 2

[50] Zhengqi Li and Noah Snavely. Learning intrinsic image decomposition from watching the world. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9039–9048. IEEE, 2018. 2

[51] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2041–2050. IEEE, 2018. 6

[52] Zhengqin Li, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and svbrdf from a single image. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2472–2481. IEEE, 2020. 1, 2

[53] Zhengqin Li, Dilin Wang, Ka Chen, Zhaoyang Lv, Thu Nguyen-Phuoc, Milim Lee, Jia-Bin Huang, Lei Xiao, Yufeng Zhu, Carl S Marshall, et al. Lirm: Large inverse rendering model for progressive reconstruction of shape, materials and view-dependent radiance fields. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 505–517. IEEE, 2025. 2

[54] Zhibing Li, Tong Wu, Jing Tan, Mengchen Zhang, Jiaqi Wang, and Dahua Lin. Idarb: Intrinsic decomposition for arbitrary number of input views and illuminations. In *International Conference on Learning Representations*, pages 36830–36854, 2025. 2, 3, 6

[55] Ruofan Liang, Zan Gojcic, Huan Ling, Jacob Munkberg, Jon Hasselgren, Chih-Hao Lin, Jun Gao, Alexander Keller, Nandita Vijaykumar, Sanja Fidler, et al. Diffusionrenderer: Neural inverse and forward rendering with video diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26069–26080. IEEE, 2025. 2, 3, 6, 7, 8, 16, 17, 21

[56] Zhihao Liang, Qi Zhang, Ying Feng, Ying Shan, and Kui Jia. Gs-ir: 3d gaussian splatting for inverse rendering. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21644–21653. IEEE, 2024. 2

[57] Haotong Lin, Sili Chen, Junhao Liew, Donny Y Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025. 3

[58] Philipp Lindenberger, Paul-Edouard Sarlin, Ondrej Miksik, and Marc Pollefeys. FlexGlue: Sparse-to-Dense Feature Matching and Visual Disambiguation. *arXiv preprint arXiv:2610.32976*, 2026. 3

[59] Yuan Liu, Sida Peng, Lingjie Liu, Qianqian Wang, Peng Wang, Christian Theobalt, Xiaowei Zhou, and Wenping Wang. Neural rays for occlusion-aware image-based rendering. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 7814–7823. IEEE, 2022. 3

[60] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 16

[61] Yihang Luo, Shangchen Zhou, Yushi Lan, Xingang Pan, and Chen Change Loy. 4rc: 4d reconstruction via conditional querying anytime and anywhere. In *ICML*, 2026. 3

[62] Juliette Marrie, Romain Ménégaux, Michael Arbel, Diane Larlus, and Julien Mairal. Ludvig: Learning-free uplifting of 2d visual features to gaussian splatting scenes. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7440–7450. IEEE, 2025. 3

[63] Abhimitra Meka, Michael Zollhöfer, Christian Richardt, and Christian Theobalt. Live intrinsic video. *ACM Transactions on Graphics (TOG)*, 35(4):1–14, 2016. 2

[64] Takuya Narihira, Michael Maire, and Stella X Yu. Direct intrinsics: Learning albedo-shading decomposition by convolutional regression. In *Proceedings of the IEEE international conference on computer vision*, pages 2992–2992, 2015. 2

[65] Jiahao Niu, Rongjia Zheng, Wenju Xu, Wei-Shi Zheng, and Qing Zhang. Sgs-intrinsic: Semantic-invariant gaussian splatting for sparse-view indoor inverse rendering. *arXiv preprint arXiv:2603.27516*, 2026. 2

[66] David Nordström, Xinyue Zhang, Thibaut Loiseau, Vincent Lepetit, and Fredrik Kahl. Roma-ω: What feed-forward 3d models know about image matching. *arXiv preprint arXiv:2609.09507*, 2026. 2, 3, 4, 5, 7, 8, 15, 21, 23

[67] Igor Pavlovic, Thiemo Wandel, Anton Obukhov, Luca Bartolomei, Andrey Davydov, Fabio Tosi, Matteo Poggi, Sabine Süsstrunk, and Dengxin Dai. Marigold v2: Revisiting diffusion transformers for monocular depth estimation. *ACM Trans. Graph.*, 45(6), 2026. 1, 2, 6, 7, 8, 17, 21

[68] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12159–12168. IEEE, 2021. 4, 15

[69] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. In *International Conference on Learning Representations*, pages 28085–28128, 2025. 20

[70] Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10892–10902. IEEE, 2021. 2, 5, 6, 8, 16, 17, 18, 19, 20, 21, 22

[71] Mehdi SM Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lučić, Daniel Duckworth, Alexey Dosovitskiy, et al. Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6219–6228. IEEE, 2022. 3

[72] Sam Sartor and Pieter Peers. Teamwork: Collaborative diffusion with low-rank coordination and adaptation. In *ACM SIGGRAPH Asia Conference Proceedings*, 2025. 1, 2

[73] Katharina Schmid, Nicolas von Lützow, Jozef Hladkỳ, Angela Dai, and Matthias Nießner. Genrecon: Bridging generative priors for multi-view 3d scene reconstruction. *arXiv preprint arXiv:2605.23888*, 2026. 2, 3, 6, 19, 20

[74] Thomas Schops, Johannes L Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3260–3269, 2017. 1, 2, 6, 7, 8, 9, 16, 21

[75] Soumyadip Sengupta, Jinwei Gu, Kihwan Kim, Guilin Liu, David W Jacobs, and Jan Kautz. Neural inverse rendering of an indoor scene from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8598–8607, 2019. 2

[76] Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Buló, Norman Müller, Matthias Nießner, Angela Dai, and Peter Kontschieder. Panoptic lifting for 3d scene understanding with neural fields. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9043–9052. IEEE, 2023. 3

[77] Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025. 15

[78] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 15

[79] Assaf Singer, Maroun Khriesh, Noam Rotstein, Yoni Kasten, and Or Litany. ReLift: Dense 3D tracking as lifted trajectory repair, 2026. 3

[80] Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. *Advances in neural information processing systems*, 34: 19313–19325, 2021. 4

[81] Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *2021 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 7491–7500. IEEE, 2021. 2

[82] Edgar Sucar, Eldar Insafutdinov, Zihang Lai, and Andrea Vedaldi. V-DPM: 4d video reconstruction with dynamic point maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. 3

[83] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Generalizable patch-based neural rendering. In *European Conference on Computer Vision*, pages 156–174. Springer, 2022. 3

[84] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. *CVPR*, 2021. 3

[85] Yihong Sun, Seoung Wug Oh, Jiahui Huang, Bharath Hariharan, and Joon-Young Lee. Unified video dense prediction from disjoint data. In *European Conference on Computer Vision*, pages 453–473. Springer, 2026. 2, 6, 7, 8, 16, 17, 21

[86] Marshall Tappen, William Freeman, and Edward Adelson. Recovering intrinsic images from a single image. *Advances in neural information processing systems*, 15, 2002. 2

[87] Prune Truong, Martin Danelljan, Radu Timofte, and Luc Van Gool. Pdc-net+: Enhanced probabilistic dense correspondence network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8):10247–10266, 2023. 3

[88] Vadim Tschernezki, Iro Laina, Diane Larlus, and Andrea Vedaldi. Neural feature fusion fields: 3d distillation of self-supervised 2d image representations. In *2022 International Conference on 3D Vision (3DV)*, pages 443–453. IEEE, 2022. 3

[89] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5294–5306. IEEE, 2025. 3, 4, 6

[90] Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-Ω. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. 2, 3, 4, 7, 8, 15, 21, 23

[91] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2021. 3

[92] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20697–20709. IEEE, 2024. 3

[93] Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *International Conference on Learning Representations*, pages 10481–10497, 2026. 3

[94] Zian Wang, Jonah Philion, Sanja Fidler, and Jan Kautz. Learning indoor inverse rendering with 3d spatially-varying lighting. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12518–12527. IEEE, 2021. 2

[95] Yair Weiss. Deriving intrinsic images from image sequences. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, pages 68–75. IEEE, 2001. 2

[96] Jiaye Wu, Sanjoy Chowdhury, Hariharmano Shanmugaraja, David Jacobs, and Soumyadip Sengupta. Measured albedo in the wild: Filling the gap in intrinsics evaluation, 2023. 2, 6, 7, 8, 16

[97] Liwen Wu, Rui Zhu, Mustafa B Yaldiz, Yinhao Zhu, Hong Cai, Janarbek Matai, Fatih Porikli, Tzu-Mao Li, Manmohan Chandraker, and Ravi Ramamoorthi. Factorized inverse path tracing for efficient and accurate material-lighting estimation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3825–3835. IEEE, 2023. 2

[98] Xiangzuo Wu, Chengwei Ren, Jun Zhou, Xiu Li, and Yuan Liu. Mvinverse: Feed-forward multiview inverse rendering in seconds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 37343–37357, 2026. 2, 3, 6, 7, 8, 17, 21

[99] Yao Yao, Jingyang Zhang, Jingbo Liu, Yihang Qu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. Neilf: Neural incident light field for physically-based material estimation. In *European conference on computer vision*, pages 700–716. Springer, 2022. 2

[100] Genzhi Ye, Elena Garces, Yebin Liu, Qionghai Dai, and Diego Gutierrez. Intrinsic video and applications. *ACM Transactions on Graphics (ToG)*, 33(4):1–11, 2014. 2

[101] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4576–4585. IEEE, 2021. 3

[102] Ye Yu and William AP Smith. Inverserendernet: Learning single image inverse rendering. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3150–3159. IEEE, 2019. 2

[103] Zheng Zeng, Valentin Deschaintre, Iliyan Georgiev, Yannick Hold-Geoffroy, Yiwei Hu, Fujun Luan, Ling-Qi Yan, and Miloš Hašan. Rgb↔x: Image decomposition and synthesis using material-and lighting-aware diffusion models. In *ACM SIGGRAPH 2024 conference papers*, pages 1–11, 2024. 1, 2, 6, 7, 8, 17, 20, 21

[104] Zheng Zeng, Marco Salvi, Lifan Wu, Jan Novák, Daqi Lin, Saeed Hadadan, Yichen Sheng, Robert Pottorff, Shiqiu Liu, Ravi Ramamoorthi, et al. Rgbx-next: Towards realistic generative rendering from g-buffers. *arXiv preprint arXiv:2608.13929*, 2026. 1, 2

[105] Chuhan Zhang, Guillaume Le Moing, Skanda Koppula, Ignacio Rocco, Liliane Momeni, Junyu Xie, Shuyang Sun, Rahul Sukthankar, Joëlle K Barral, Raia Hadsell, et al. Efficiently reconstructing dynamic scenes one d4rt at a time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7382–7392, 2026. 4

[106] Jingyang Zhang, Yao Yao, Shiwei Li, Jingbo Liu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. Neilf++: Inter-reflectable light fields for geometry and material estimation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3578–3587. IEEE, 2023. 2

[107] Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *2021 IEEE/CVF Conference on Computer Vision*

and Pattern Recognition (CVPR), pages 5449–5458. IEEE, 2021. 2

[108] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. In *European Conference on Computer Vision*, pages 1–19. Springer, 2024. 3

[109] Mengchen Zhang, Yang Li, Chunshi Wang, Tong Wu, and Chunchao Guo. Worldrecon: Generative multi-view mesh reconstruction for large-scale 3d world. *arXiv preprint arXiv:2610.18846*, 2026. 2, 3

[110] Tianyuan Zhang, Zhengfei Kuang, Haian Jin, Zexiang Xu, Sai Bi, Hao Tan, He Zhang, Yiwei Hu, Milos Hasan, William Freeman, et al. Relitlrm: Generative relightable radiance for large reconstruction models. In *International Conference on Learning Representations*, pages 11366–11388, 2025. 3

[111] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (ToG)*, 40(6):1–18, 2021. 2

[112] Yuanqing Zhang, Jiaming Sun, Xingyi He, Huan Fu, Rongfei Jia, and Xiaowei Zhou. Modeling indirect illumination for inverse rendering. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18622–18631. IEEE, 2022. 2

[113] Yuchen Zhang, Nikhil Keetha, Chenwei Lyu, Bhuvan Jhamb, Yutian Chen, Yuheng Qiu, Jay Karhade, Shreyas Jha, Yaoyu Hu, Deva Ramanan, Sebastian Scherer, and Wenshan Wang. Ufm: A simple path towards unified dense correspondence with flow. In *arXiV*, 2025. 3

[114] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J Davison. In-place scene labelling and understanding with implicit scene representation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15818–15827. IEEE, 2021. 3

[115] Shijie Zhou, Haoran Chang, Sicheng Jiang, Zhiwen Fan, Zehao Zhu, Dejia Xu, Pradyumna Chari, Suya You, Zhangyang Wang, and Achuta Kadambi. Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21676–21685. IEEE, 2024. 3

[116] Tinghui Zhou, Philipp Krahenbuhl, and Alexei A Efros. Learning data-driven reflectance priors for intrinsic image decomposition. In *Proceedings of the IEEE international conference on computer vision*, pages 3469–3477, 2015. 2

[117] Jingsen Zhu, Fujun Luan, Yuchi Huo, Zihao Lin, Zhihua Zhong, Dianbing Xi, Rui Wang, Hujun Bao, Jiaxiang Zheng, and Rui Tang. Learning-based inverse rendering of complex indoor scenes with differentiable monte carlo raytracing. In *Siggraph asia 2022 conference papers*, pages 1–8, 2022. 2, 5, 6, 8, 16, 17, 18, 19, 20, 21, 22

[118] Rui Zhu, Zhengqin Li, Janarbek Matai, Fatih Porikli, and Manmohan Chandraker. Irisformer: Dense vision transformers for single-image inverse rendering in indoor scenes. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2812–2821. IEEE, 2022. 2

PointX: Multi-View Consistent Intrinsics by Querying Surface Points

Appendix

A. Model and Training Details

We complement Sec. 3 with further details on the architecture, the training procedure and the training data, in the notation of the main paper.

A.1. Architecture

Transformer and decoder. The transformer is initialized from the released 1B-parameter checkpoint of VGGT-Ω [90]: a DINOv3 ViT-L/16 patch encoder [77] followed by 24 frame-attention and 24 global-attention blocks of width 1024 in alternation. The DPT decoder [68] follows the dense head of VGGT-Ω without its output layer and is trained from scratch. It fuses the tokens of four transformer blocks into a feature map at $1/4$ of the image resolution, and two further upsampling stages bring it to the full resolution, each with a skip connection from the first two convolution blocks of an ImageNet-pretrained VGG19 [78]. The decoder feature has 128 channels and the tokens have 256 dimensions.

Tracks. For the tracks of RoMa-Ω [66], we use its `fast` setting and take its predicted overlap probability as the confidence of the matcher.

Heads and model size. Each head is a two-layer MLP with a hidden width of 1024 and a GELU between its two layers; its output passes through the activation given in Sec. 3.2. In total, PointX has 999M trainable parameters: 911M in the transformer, 87M in the decoder and 1.3M in the heads. The frozen VGGT-Ω that estimates the input geometry and RoMa-Ω are not counted.

A.2. Training

Training sets. Each GPU holds 24 images per step, as $\lfloor 24/N \rfloor$ sets of N views, with N drawn from 2 to 24; with probability 0.1, a step instead holds 24 unrelated photographs, each a set of a single view. A multi-view set consists of frames of one camera trajectory: either a run of frames at a random stride, or a random frame together with frames that are close to it in camera pose.

Augmentation. Images are resized and cropped to a width of 512 px and to a random aspect ratio between 0.33 and 1 that is shared by all images of a step, and each set is mirrored horizontally with probability 0.5. With probability 0.5, the input images of a set are degraded by a random combination of blur, downsampling, noise and JPEG compression, while their labels are left unchanged.

Queries. Each step draws 16,384 query pixels per GPU, all in one randomly chosen view of each set. Half of them form the 16×16 windows of the gradient loss. We draw 10% of the queries on metallic surfaces, 27% near depth discontinuities and the rest uniformly. The source of the tracks is drawn uniformly among the three sources of Sec. 3.3 at every step; where a dataset provides no ground-truth cameras or depth, the geometry of VGGT-Ω takes the place of the ground truth.

Losses. The losses of the five properties are summed with equal weights. The gradient loss is an L_1 loss on the finite differences of prediction and label within each window, at four scales that each subsample the window by two. Metallic surfaces are rare, so within the metallicity loss, the queries on them are up-weighted by their inverse frequency, capped at 2. In addition to the queries, the pixels of every view on a grid of stride 4 are decoded from their own token alone and supervised with the same losses, with weight 0.5. Each loss is applied only where its label is valid: the dataset must provide the property (Tab. 5) and define it at the pixel, which excludes, *e.g.*, the normal of the sky and the irradiance of saturated pixels.

Irradiance exposure. Where frames are exposed individually, the irradiance label of a surface point differs between two views by the ratio of their exposures. We estimate this ratio as the median ratio of the labels over the surface points that both views observe, which we find with the ground-truth geometry, solve for the exposure of every view of a set by least squares, and divide its labels by it. The common exposure of a set remains arbitrary: we fit one scale between prediction and label per set in closed form before the L_1 loss, and penalize its logarithm with weight 0.1. Where the views cannot be related, as for single photographs, the scale is fitted per image.

Optimization. We use AdamW [60] ($\beta = (0.9, 0.95)$, weight decay 0.05), bfloat16 mixed precision, gradient clipping at an L^2 norm of 10, and an exponential moving average of the parameters with decay 0.999, which is the model we evaluate. PointX is trained with the losses above for 70k steps, with 5k warm-up steps to a learning rate of 1×10^{-4} and a cosine decay to 10^{-5}. Training takes approximately 5 days to complete on 8 A100 80GB GPUs.

A.3. Training Data

Table 5 lists the training data. For Hypersim [70] and InteriorVerse [117], we follow the official scene splits and train on the training and validation scenes; for InteriorVerse we use both of its lens versions. The three in-house datasets contain rendered indoor and outdoor scenes. The photographs are real images with pseudo-labels from a single-image model and form the single-image steps. The rendered datasets are sampled with equal probability.

Table 5. **Training data.** Training scenes and images of each dataset, and the labels it provides: albedo a, irradiance e, normal n, roughness r and metallicity m. *Geometry* marks the datasets whose cameras and metric depth we use for ground-truth tracks and to relate the exposures of a set of views. *Exposed per image, *i.e.* known up to one scale per image. †For the scenes rendered with a pinhole camera.

Dataset	Scenes	Images	Resolution	a	e	n	r	m	Geometry
Hypersim [70]	411	67k	1024×768	✓	✓*	✓	–	–	✓†
InteriorVerse [117]	3,966	100k	640×480	✓	–	✓	✓	✓	–
In-house A	125	162k	1820×1024	✓	✓*	✓	✓	–	✓
In-house B	761	617k	1820×1024	✓	–	✓	✓	✓	–
In-house C	479	81k	1920×1080	✓	✓	✓	✓	✓	✓
Photographs (pseudo-labels)	–	1.06M	768 (short side)	✓	✓*	✓	–	–	–

B. Evaluation Details

Test sets. On Hypersim [70], we use the 46 scenes of the official test split with up to two camera trajectories each, and form 69 sets of eight views: a random frame of a trajectory and the seven frames closest to it in camera pose. On InteriorVerse [117], we form sets of up to eight views in the same way, one for each of 105 random scenes of the official test split (773 images). On MAW [96], we use all 874 photographs of its 38 scenes, in sets of up to eight in capture order. On ETH3D [74], we use the 13 scenes with laser scans of the high-resolution multi-view benchmark, in 50 sets of up to eight overlapping views (395 images). The photographs of MAW and ETH3D are downsampled to a width of 1024 px.

Baselines. We run every baseline with its released code and weights and follow its conventions for the color space of its inputs and outputs. The video models [20, 55, 85] receive the views of a set as the frames of one clip; for the single-image results of Tab. 3, every method processes each image separately. MVID [14] predicts the normals of a set in the camera frame of its first view. We rotate them into the frame of each view with the ground-truth cameras. All predictions are resized to the resolution of the ground truth before they are scored.

Accuracy. Albedo errors are computed in linear RGB, and LPIPS on gamma-encoded images. For PSNR_s, we fit one scale per color channel to the ground truth by least squares; the irradiance is then compared after gamma encoding. All metrics are computed over the pixels with a valid label.

Disagreement. For two views $u \neq v$ of a set, let $\mathcal{C}_{uv}$ be the pixels q of view u whose surface point is also visible in view v, at the location $x_v(q)$, and let M_v be the map that a method predicts for view v. We define

$$\mathrm{RMSE_r} = \left(\frac{\sum_{u \neq v} \sum_{q \in \mathcal{C}_{uv}} \|M_u[q] - M_v[x_v(q)]\|^2}{\sum_{u \neq v} \sum_{q \in \mathcal{C}_{uv}} \|M_u[q] - M_v[x_v(\pi_{uv}(q))]\|^2} \right)^{1/2}, \tag{4}$$

where π_{uv} is a random permutation of $\mathcal{C}_{uv}$, so that the denominator pairs every pixel with the correspondence of another pixel. It is computed per set and averaged over the sets. In Sec. 4.2, $\mathcal{C}_{uv}$ holds the pixels whose projection with the ground-truth depth matches the depth map of view v within 5%; in Sec. 4.1, the projections of the surface points that both views see. For the irradiance, M_v is first scaled to the exposure of M_u by the median ratio of the two maps over $\mathcal{C}_{uv}$, and both maps are gamma-encoded.

Surface points. We complete the protocol of Sec. 4.1. From the unprojected depth maps of a set, we keep one point per voxel, with voxels the size of a pixel at the median depth, and draw up to 500k of the points that at least two views see. A view sees a point if $c_v > 1/2$ in Eq. (2), *i.e.*, if $\delta_v < \tau$. All maps are read at the projections of a point by bilinear interpolation, the albedo in linear RGB and the normals in the world frame. Irradiance maps are first brought to a common exposure per set as in Sec. A.2, separately for the ground truth and for each method. PointX predicts the normal of a point in the mean-frame, which we map to the world frame by aligning the cameras of VGGT-Ω with the ground-truth cameras. On Hypersim, we use the 41 sets rendered with a pinhole camera, as the others cannot be reprojected.

Registration of InteriorVerse. InteriorVerse provides depth maps and intrinsics but no camera poses. We recover the poses of a set by point-to-plane ICP [9] on the ground-truth depth maps, optimizing all views jointly and starting from the cameras of VGGT-Ω. With the initial cameras of VGGT-Ω, 41.9% of the pixels of a view that are visible in another view match its depth map within 1%; after registration, 96.1% do. For reference, the ground-truth cameras of Hypersim reach 96.9%.

C. Additional Results

C.1. Intrinsics of Surface Points

In Sec. 4.1, the per-view predictions of a baseline are combined by their unweighted mean. Table 6 compares this rule with a mean weighted by the confidence of Eq. (2) and with a median. The rule has little effect on any method. Figures 7 to 9 show the intrinsic point clouds for further scenes of Hypersim [70] and InteriorVerse [117].

Table 6. **Fusion of per-view predictions at surface points.** Accuracy on the task of Sec. 4.1 when the per-view predictions of a method are combined by their mean, by their mean weighted with the confidence of Eq. (2), or by their median. PointX predicts every point once and needs no fusion, so its value is the same under each rule. The other metrics of Tab. 1 behave alike. Best, second and third best are highlighted. †Trained on the Hypersim test scenes, not ranked.

Method	Hypersim [70]						InteriorVerse [117]					
	Albedo PSNR ↑			**Normal** Mean° ↓			**Albedo** PSNR ↑			**Normal** Mean° ↓		
	Mean	Weighted	Median	Mean	Weighted	Median	Mean	Weighted	Median	Mean	Weighted	Median
Intrinsic Image Diffusion [40]	12.37	12.37	12.36		–		16.11	16.12	16.10		–	
RGB↔X† [103]	24.09	24.09	24.33	8.0	8.0	7.8	9.57	9.57	9.55	12.3	12.2	11.4
DiffusionRenderer [55]	14.50	14.49	14.49	13.6	13.5	13.0	10.82	10.82	10.83	11.4	11.3	10.7
V-RGBX [20]	11.98	11.99	11.93	23.3	23.3	23.3	11.42	11.42	11.34	22.4	22.4	22.2
MVInverse [98]	18.62	18.61	18.62	8.2	8.2	7.9	18.41	18.43	18.41	7.8	7.7	7.5
RGB↔X + GeoID† [11]	23.62	23.63	23.82		–		10.89	10.89	10.87		–	
UniD [85]	17.34	17.34	17.33	14.9	14.8	14.6	9.63	9.63	9.65	15.7	15.7	15.3
Marigold V2 [67]	21.16	21.17	21.31	8.4	8.4	8.2	11.57	11.58	11.60	8.7	8.6	8.4
MVID [14]	26.22	26.24	26.38	8.9	8.8	8.6	24.64	24.67	24.67	24.3	24.3	24.3
MVID-Ω [14]	17.72	17.73	17.73	11.5	11.5	11.3	19.32	19.33	19.31	25.4	25.4	25.3
PointX w/o pooling	20.33	20.33	20.34	8.6	8.6	8.4	25.41	25.44	25.47	5.7	5.6	5.4
PointX (Ours)	20.32	20.32	20.32	8.3	8.3	8.3	25.50	25.50	25.50	5.2	5.2	5.2

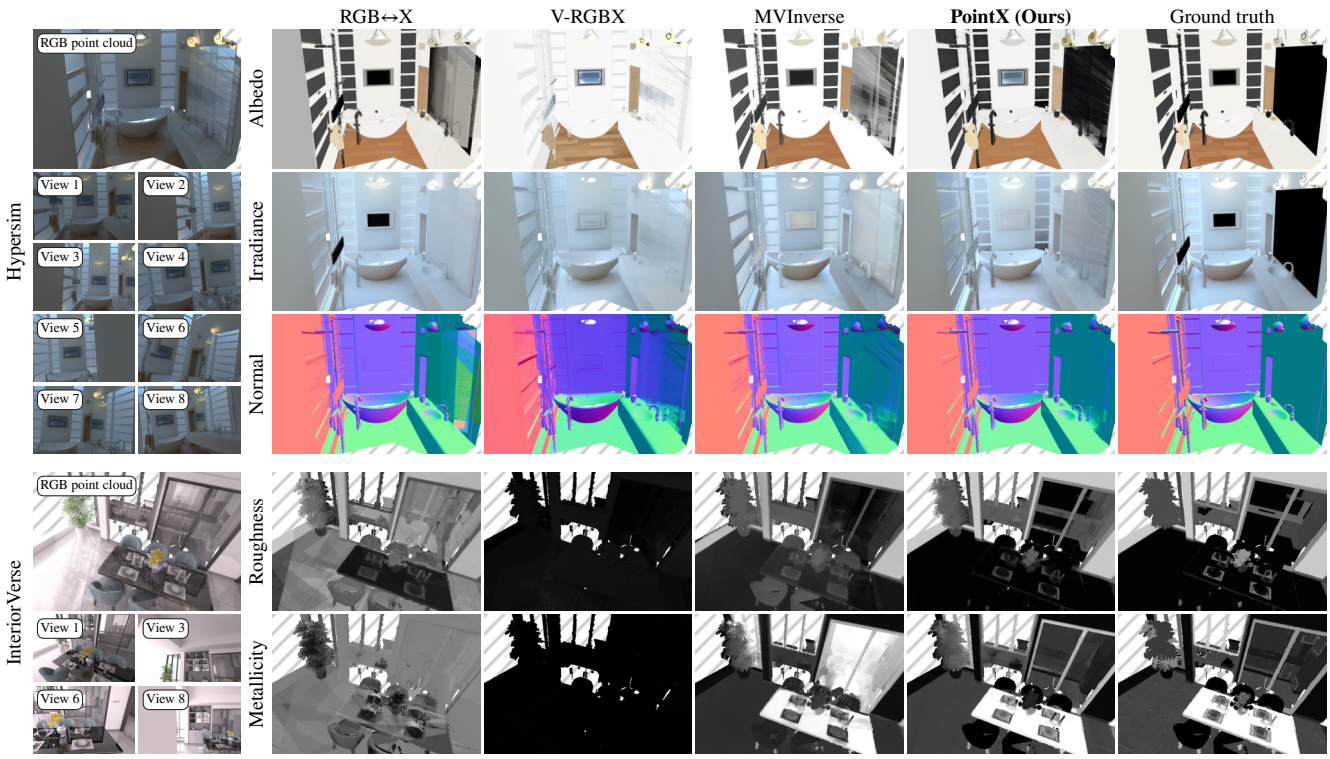

Figure 7. **Intrinsics of surface points: additional scenes.** A Hypersim [70] (top) and an InteriorVerse [117] (bottom) scene, shown as in Fig. 3. Where the views of a baseline disagree, their average shows seams and patches, *e.g.*, on the glass panel (top) and on the floor (bottom).

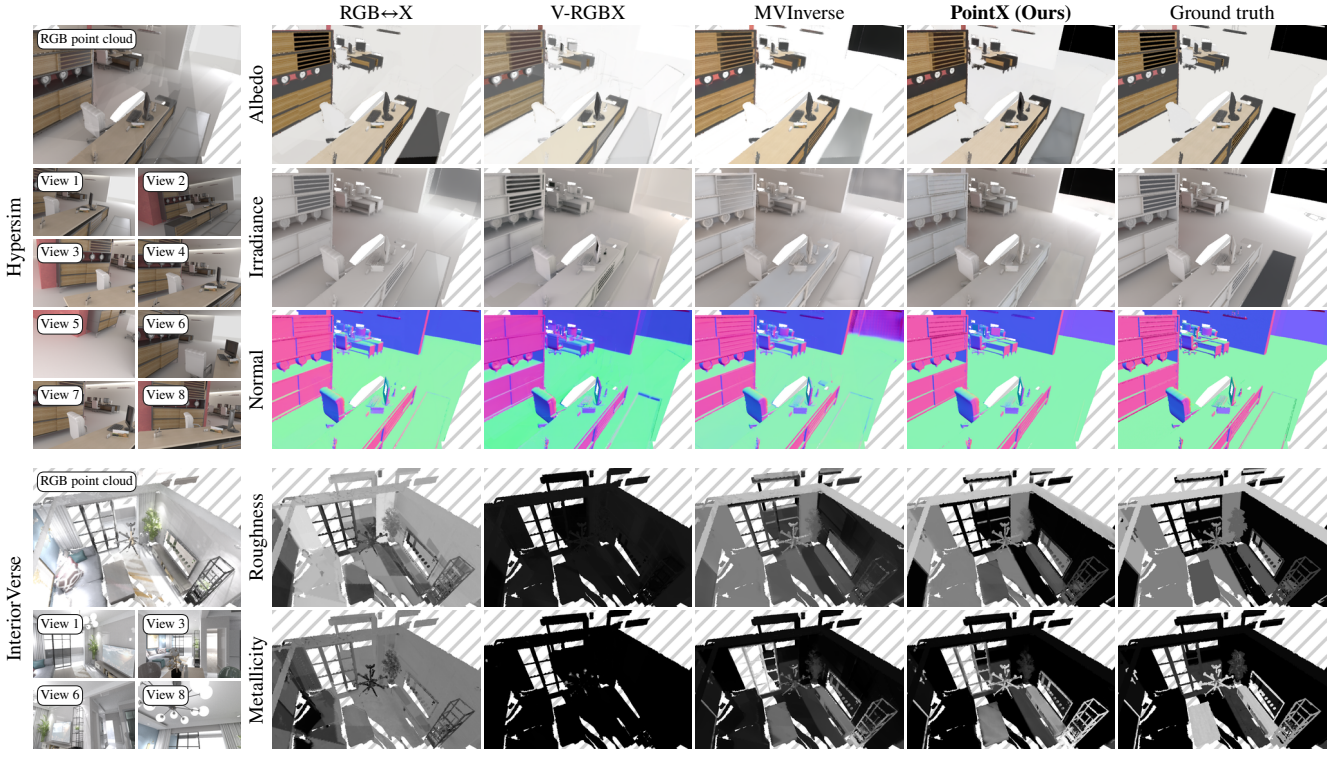

Figure 8. **Intrinsics of surface points: additional scenes (continued).** Two further scenes, shown as in Fig. 7.

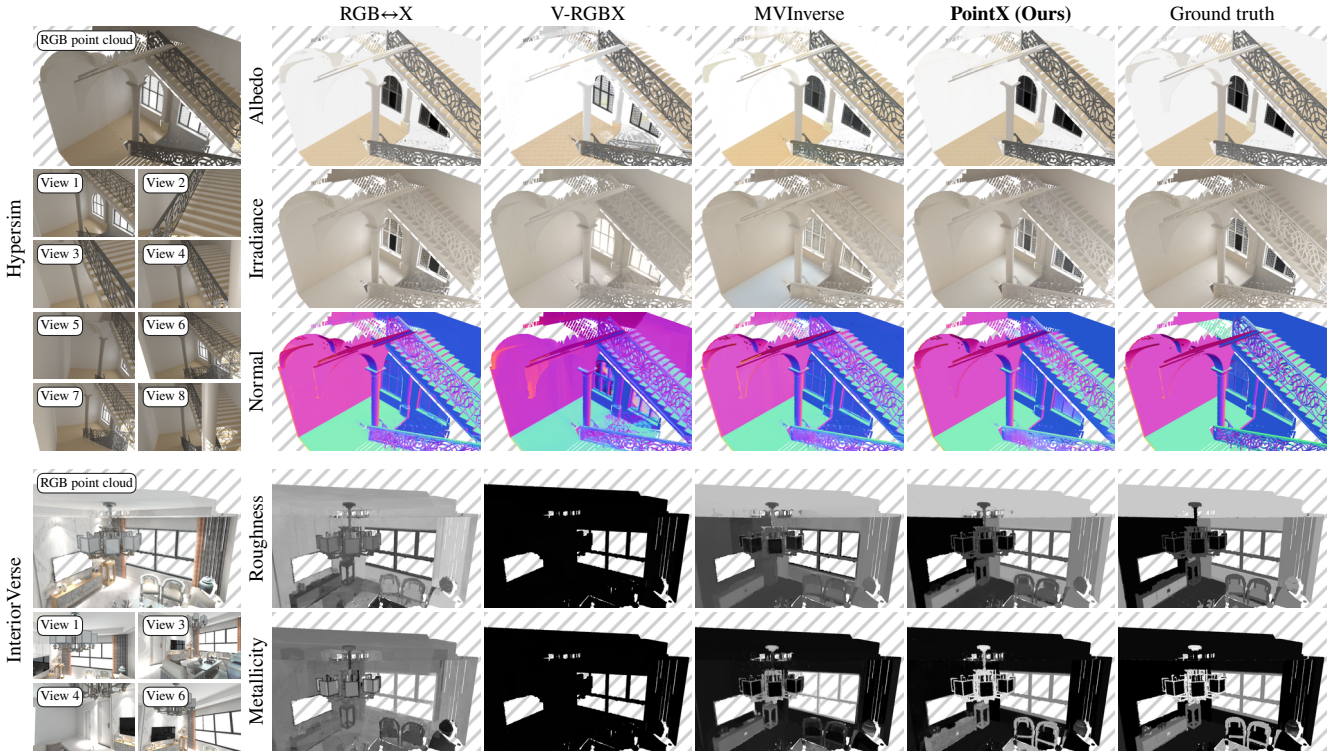

Figure 9. **Intrinsics of surface points: additional scenes (continued).** Two further scenes, shown as in Fig. 7.

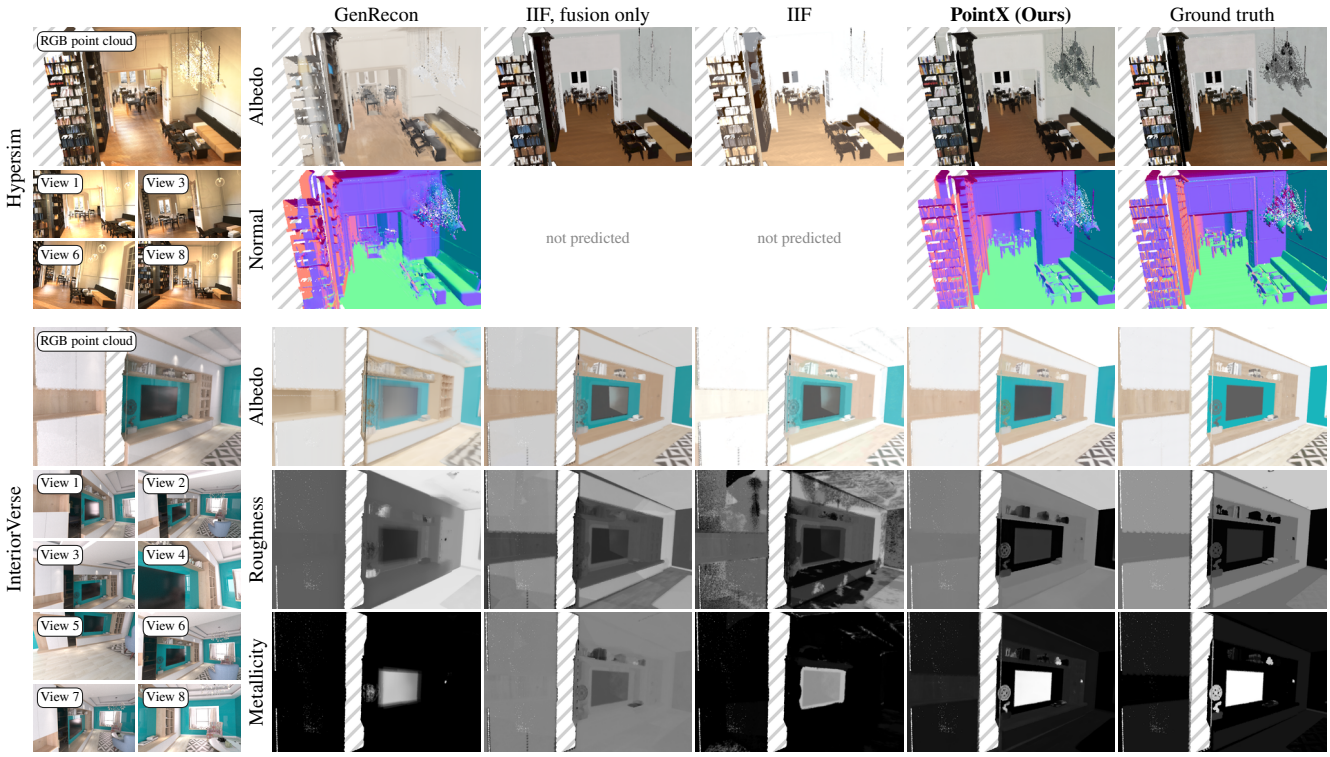

Figure 10. **Additional baselines.** A Hypersim [70] (top) and an InteriorVerse [117] (bottom) scene, shown as in Fig. 3, for GenRecon [73] and Intrinsic Image Fusion (IIF) [41] with and without its inverse path tracing. The materials of GenRecon look plausible but differ from the true ones. The inverse path tracing of IIF brightens the albedo, since much of the light in the scene comes from surfaces that the eight views do not show.

C.2. Additional Baselines

In Sec. 4, we omit methods that need posed images or per-scene optimization. Since the task of Sec. 4.1 provides the cameras and the geometry of every scene, we evaluate two of them on it: Intrinsic Image Fusion [41] and GenRecon [73]. Table 7 shows that PointX is considerably more accurate than both on every property of InteriorVerse. On Hypersim, it is more accurate than GenRecon, while Intrinsic Image Fusion builds on RGB↔X [103], which was trained on the test scenes. Figure 10 shows two scenes.

Setup. Intrinsic Image Fusion draws samples of RGB↔X for every view, fuses them into one material field on a mesh of the scene, and refines this field by inverse path tracing. We give it the ground-truth cameras, the ground-truth depth maps as its mesh, and the instance segmentation it requires, from SAM 2 [69] masks that we link across views with the ground-truth depth. GenRecon generates a mesh with material textures from posed images and sparse scene points, for which we use the ground-truth cameras and points of the ground-truth depth maps. The result of each method is rendered into the views of a set and scored like the maps of the other baselines. Intrinsic Image Fusion optimizes a set for more than one hour on an A100 GPU and GenRecon takes about ten minutes, whereas PointX takes seconds (Sec. 4.4).

Intrinsic Image Fusion. The fusion makes the samples of RGB↔X about as consistent as the ground-truth maps (Tab. 1), but it does not correct them: its accuracy remains that of RGB↔X. The inverse path tracing explains the images by the light that the surfaces of the mesh exchange. Its authors evaluate it on scenes covered by about 200 views, whereas our sets hold eight views of a part of a scene, so much of this light comes from surfaces that no view shows. The optimization compensates for the missing light with a brighter albedo, which saturates on Hypersim. On InteriorVerse, it improves the metallicity but degrades the roughness and the PSNR_s of the albedo. On one of the scenes of its authors, its released pipeline raises the albedo PSNR of the fusion from 15.0 to 22.6 dB with all 202 views, and not at all with eight of them.

GenRecon. GenRecon is a generative reconstruction model, and its authors evaluate its geometry only. Its materials are sampled from a 3D generative prior conditioned on the images: they look plausible but are far from the true ones. Its mesh also deviates from the true surface, in particular on Hypersim, where a set often shows only a part of a large room, so the renders of this one mesh still disagree at the true surface points.

Table 7. **Additional baselines.** The protocol of Tab. 1 for two methods that require posed images, GenRecon [73] and Intrinsic Image Fusion [41]. Intrinsic Image Fusion fuses samples of RGB↔X, listed for reference, and refines the result by inverse path tracing; *fusion only* omits this refinement. Neither method predicts irradiance, and Intrinsic Image Fusion predicts no normals. The best result is highlighted. †Based on RGB↔X, which was trained on the Hypersim test scenes; not ranked.

Method	Hypersim [70]					InteriorVerse [117]								
	Accuracy			Disagreement		Accuracy					Disagreement			
	Albedo		Normal	Albedo	Normal	Albedo		Normal	Rough.	Metal.	Albedo	Normal	Rough.	Metal.
	PSNR ↑	PSNR_s ↑	Mean° ↓	RMSE_r ↓	Mean° ↓	PSNR ↑	PSNR_s ↑	Mean° ↓	RMSE ↓	RMSE ↓	RMSE_r ↓	Mean° ↓	RMSE_r ↓	RMSE_r ↓
GenRecon [73]	11.85	13.61	23.4	0.472	13.9	14.66	15.30	12.5	0.446	0.186	0.360	9.8	0.299	0.487
RGB↔X† [103]	24.09	24.89	8.0	0.196	8.3	9.57	18.96	12.3	0.304	0.407	0.499	18.1	0.582	0.837
Intrinsic Image Fusion† [41], fusion only	24.63	25.52	–	0.068	–	10.61	19.10	–	0.281	0.432	0.106	–	0.096	0.191
Intrinsic Image Fusion† [41]	8.18	15.26	–	0.097	–	13.78	16.73	–	0.359	0.204	0.124	–	0.107	0.137
PointX (Ours)	20.32	22.76	8.3	**0** by construction		25.50	25.97	5.2	0.083	0.072	**0** by construction			

C.3. Consistency of Per-View Maps

Table 8 applies the protocol of Sec. 4.2 to the sets of Hypersim and InteriorVerse used in Sec. 4.1, with their ground-truth depth and, for InteriorVerse, the registered cameras. The results agree with those on ETH3D. Figure 11 compares the albedo that each method predicts for the same surface point in two views, on all three datasets.

Table 8. **Consistency of per-view maps on rendered scenes.** The protocol of Tab. 2 on Hypersim and InteriorVerse: disagreement between the maps of two views at ground-truth correspondences. Gray: ground-truth geometry at test time, or †trained on the Hypersim test scenes; not ranked. Best, second and third best are highlighted.

Method	Track Source	Hypersim [70]					InteriorVerse [117]				
		Albedo	Irrad.	Normal	Rough.	Metal.	Albedo	Irrad.	Normal	Rough.	Metal.
		$\text{RMSE}_r \downarrow$	$\text{RMSE}_r \downarrow$	Mean° ↓	$\text{RMSE}_r \downarrow$	$\text{RMSE}_r \downarrow$	$\text{RMSE}_r \downarrow$	$\text{RMSE}_r \downarrow$	Mean° ↓	$\text{RMSE}_r \downarrow$	$\text{RMSE}_r \downarrow$
Single-view methods											
Intrinsic Image Diffusion [40]	–	0.469	–	–	0.641	0.596	0.453	–	–	0.582	0.512
RGB↔X† [103]	–	0.228	0.321	10.8	0.409	0.643	0.515	0.434	21.8	0.605	0.838
Marigold V2 [67]	–	0.238	–	7.6	–	–	0.403	–	11.6	–	–
Video & Multi-view methods											
DiffusionRenderer [55]	–	0.326	–	13.5	0.514	0.892	0.520	–	15.2	0.813	0.948
V-RGBX [20]	–	0.264	0.324	13.4	0.484	0.906	0.471	0.386	18.5	0.805	0.961
MVInverse [98]	–	0.174	0.326	8.0	0.244	0.244	0.249	0.340	10.1	0.250	0.349
RGB↔X + GeoID† [11]	–	0.221	–	–	0.368	0.622	0.505	–	–	0.573	0.823
UniD [85]	–	0.221	0.303	12.2	–	–	0.364	0.330	19.2	–	–
MVID [14]	–	0.174	0.261	6.7	–	–	0.196	0.285	14.1	–	–
MVID-Ω [14]	–	0.201	0.277	6.6	–	–	0.217	0.278	10.2	–	–
	VGGT-Ω [90]	0.195	0.198	5.7	0.246	0.317	0.195	0.214	4.9	0.145	0.285
PointX (Ours)	RoMa-Ω [66]	0.135	0.155	4.0	0.190	0.257	0.161	0.200	4.3	0.126	0.233
	Ground truth	0.076	0.080	1.5	0.083	0.118	0.110	0.132	2.1	0.075	0.155

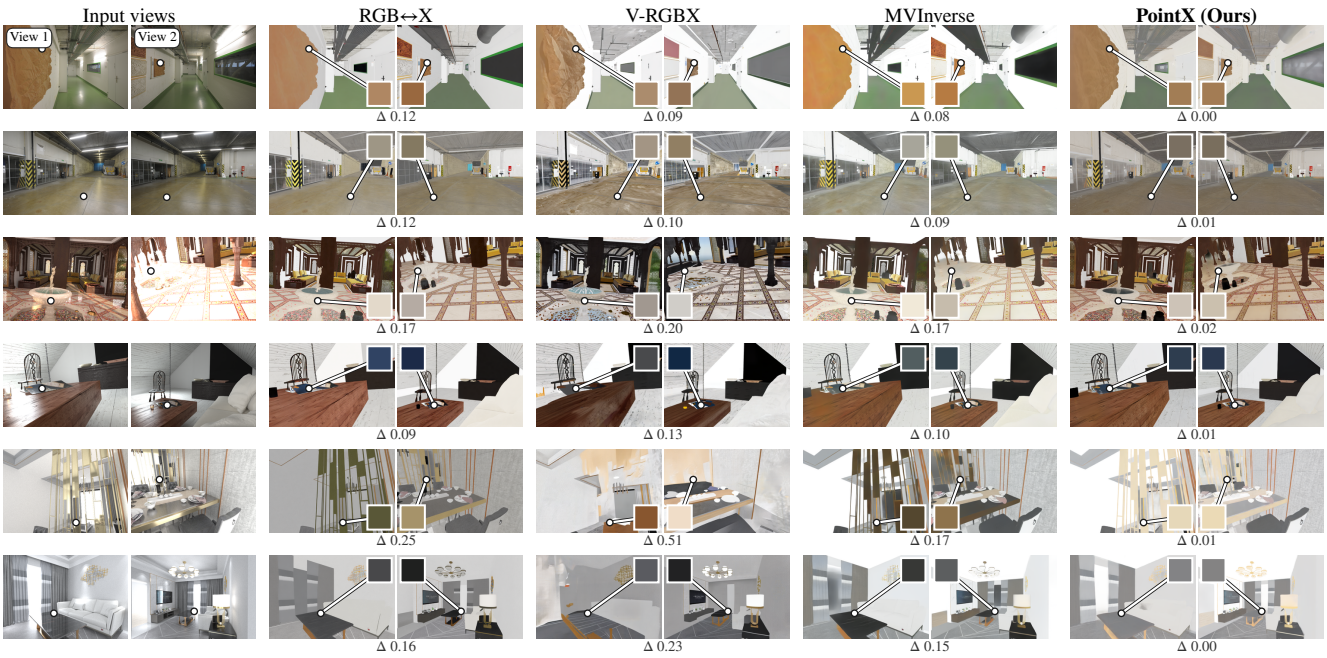

Figure 11. **Consistency of per-view maps: additional scenes.** Albedo predictions for two views of a scene on ETH3D [74] (rows 1–2), Hypersim [70] (rows 3–4) and InteriorVerse [117] (rows 5–6), shown as in Fig. 4 (PointX: RoMa-Ω tracks). The marked surface point is the same in both views; the insets show the value each method predicts for it, and Δ their mean absolute difference.

C.4. Accuracy of Per-View Maps

Figure 12 repeats the experiment of Fig. 5 on Hypersim. The views of a set are split into consecutive groups of N views that are predicted independently, so that every view is predicted from N images and the same views are scored for every N. As on InteriorVerse, every property predicted by PointX becomes more accurate with more images, MVInverse gains considerably less, and the gain remains without pooling.

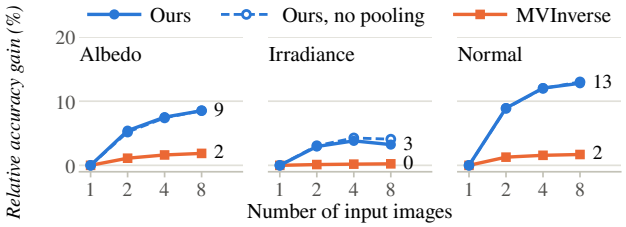

Figure 12. **Accuracy with more input images on Hypersim [70].** Reduction of the error relative to a single image, as in Fig. 5 (RMSE; mean angle for normals). PointX uses RoMa-Ω tracks.

C.5. Ablation Studies

Without the geometry channels, the model must infer the orientation of each camera in the mean-frame from the images alone (Sec. 4.4). Figure 13 shows a set where this fails: view 8 shares little content with the other views, and its normals are predicted in a wrong orientation. With the geometry channels, the normals of all views are oriented correctly. In most sets the effect is smaller, a slight tilt of the normals of individual views.

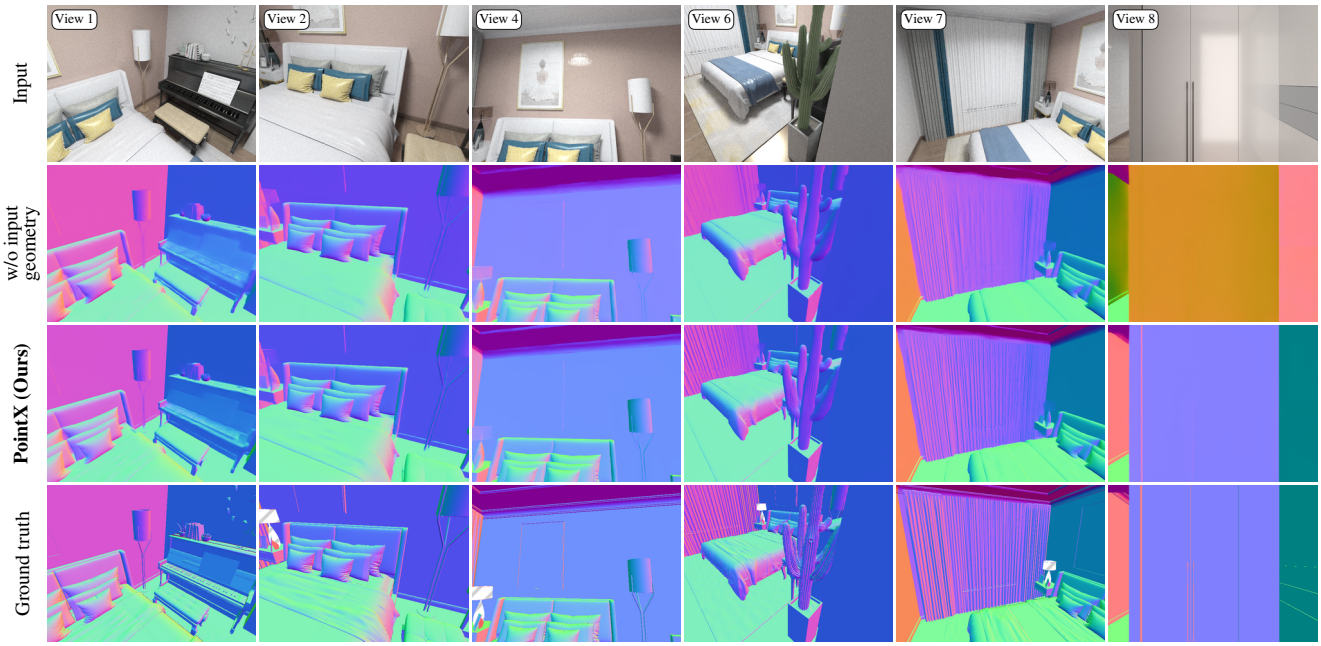

Figure 13. **Normals without the geometry channels.** Six of the eight views of an InteriorVerse [117] set and their normal maps, predicted from all eight images (RoMa-Ω tracks). Without the geometry channels, the normals of view 8, which shares little content with the other views, are predicted in a wrong orientation. Regions without ground truth are hatched.

D. Relighting and Editing

The point cloud of Fig. 1 holds one PointX query per point of the laser scan.

Relighting. Let L be the product of the predicted albedo and irradiance of a point, *i.e.*, its diffuse radiance under the original illumination. We dim it and add the radiance L_{new} of the point under new point lights, computed with a Cook–Torrance model [10] from its predicted albedo, roughness, metallicity and normal: $L' = kL + L_{\text{new}}$, with $k < 1$. Each photograph is then multiplied, in linear RGB, by L'/L at the pixels to which the points project.

Albedo editing. We scale the albedo of the points of an object and multiply each photograph by the same factor, in linear RGB, at the pixels to which these points project.

E. Limitations and Future Work

Geometry and tracks. PointX locates a surface point through the geometry or the matches it is given, so their errors carry over to its predictions. With the geometry estimated by VGGT-Ω [90], its per-view maps are less consistent than with the tracks of RoMa-Ω [66] or with ground-truth geometry (Tab. 2), and an error in an estimated camera tilts the normals of its view. RoMa-Ω in turn matches every pair of views, so its cost grows quadratically with their number. Since the tracks are an input, PointX benefits from any progress in reconstruction and matching without retraining.

Pooling. The weights of Eq. (3) have no learned parameters and depend only on the visibility of a point. They make the prediction of a point independent of the view, but not smooth along a surface: neighboring points that are visible in different sets of views are decoded from different sets of tokens, which can leave a small step between their predictions, *e.g.*, where the frame of another image ends. A confidence that also decreases towards the border of an image is a simple extension that we leave for future work.

Albedo and illumination. On Hypersim, PointX does not reach the most accurate baseline on albedo and irradiance (Tabs. 1 and 3). Part of this gap is the overall scale of the albedo, *i.e.*, the difference between PSNR and PSNR_s, and it remains when more views are added. Views of a scene under the same illumination add evidence about its surfaces, but not about how their brightness divides into albedo and irradiance. PointX also assumes a static scene whose illumination does not change between the images. Images captured under varying illumination could provide the missing evidence and are an interesting direction.

Training data. All multi-view supervision comes from rendered scenes: the real photographs are single images with pseudo-labels, and they carry no roughness or metallicity (Tab. 5). On real scenes, we evaluate the accuracy of the albedo and the consistency of all five properties, but not the accuracy of the other properties. Real multi-view captures with measured materials would benefit both training and evaluation.

Number of views. PointX encodes all images of a scene in one forward pass and is trained with up to 24 views. Its consistency holds by construction within such a set of images. Extending the pooling across sets, for long videos or large scenes, is left for future work.