World Tracing: Generative Pixel-Aligned Geometry Beyond the Visible

Hao Zhang[1,2*] **Mohamed El Banani**[1] **Jen-Hao Cheng**[1] **Paul Zhang**[1] **Yi Hua**[1]
Ben Mildenhall[1] **Christoph Lassner**[1] **Narendra Ahuja**[2] **Gengshan Yang**[1]
[1]World Labs [2]University of Illinois Urbana-Champaign

Project Page Code HuggingFace Demo

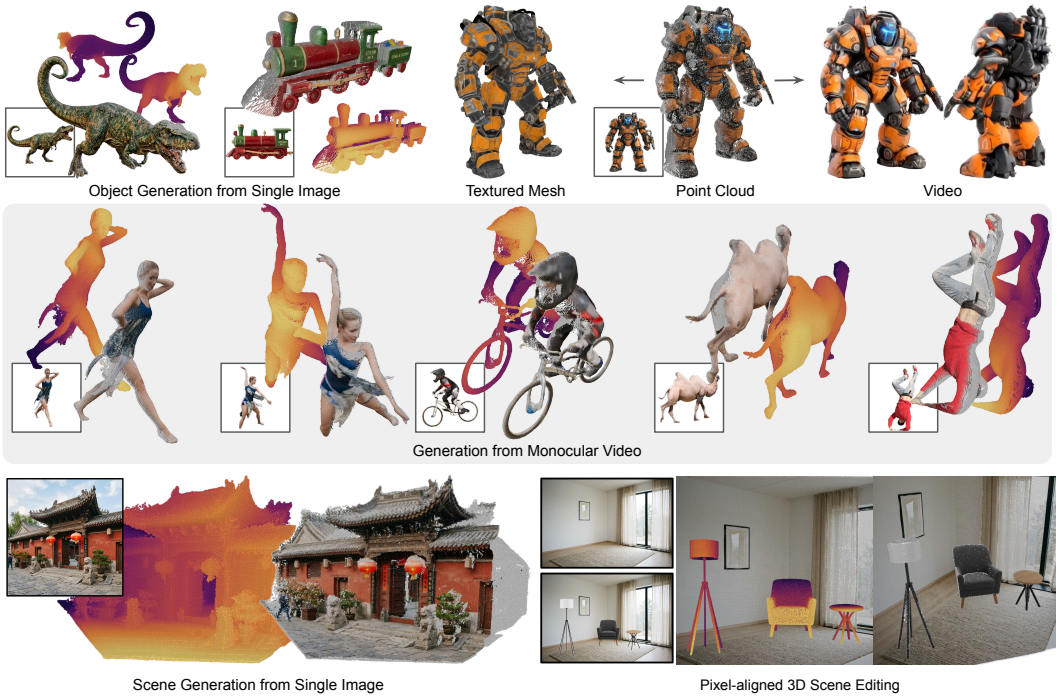

Figure 1: **World Tracing.** A pixel-aligned layered geometry representation that faithfully generates complete objects, scenes, and dynamic content from single images and monocular videos. Colored points are predicted visible surfaces; gray points are predicted surfaces hidden from inputs. We visualize depth in magma colormap. This pixel-aligned representation enables several downstream applications: training-free pose-aware mesh generation, view synthesis, scene generation and editing.

Abstract

Image-to-3D methods often trade off faithfulness and completeness: depth estimators are anchored to input pixels but stop at the visible surface, while image-to-3D models generate complete shapes that are often misaligned to the input. We introduce World Tracing, a generative pixel-aligned geometry representation that predicts 3D points aligned with the observed pixels while completing geometry beyond the visible surface. For each input pixel, World Tracing predicts

*Work done during an internship at World Labs.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

an ordered stack of camera-space 3D points, where the first layer of the stack represents the visible surface and subsequent layers represent front-to-back intersections with occluded surfaces. We instantiate this representation with a world-tracing diffusion transformer (WT-DiT), which treats multiple geometry layers as separate denoising tokens coupled through factorized and global attention. WT-DiT is trained with pixel-space flow matching and a mixed noise schedule that balances visible-surface reconstruction with occluded-geometry generation. On held-out object and scene benchmarks, World Tracing outperforms image-to-3D generators on complete geometry without sacrificing visible-surface accuracy, and it leads on in-domain dynamic benchmarks; on out-of-domain real-image benchmarks its visible layer remains competitive with dedicated depth predictors. It also preserves 2D-to-3D correspondence, enabling text-driven 3D scene editing, geometry-conditioned novel-view video synthesis, and training-free integration with textured-mesh generators.

1 Introduction

Single-image 3D estimation has traditionally been explored through two main directions. The first faithfully reconstructs the 3D structure of the observed pixels, but is restricted to the visible surface [28, 34, 74–76, 90, 91]. The second generates the full 3D object in a canonical frame, but does so at the cost of pixel alignment [6, 42, 46, 61, 68, 80]. As a result, neither paradigm provides downstream 3D pipelines what they need: faithful and complete geometry in the camera frame. The missing capability is *faithful generation*: 3D generation that accurately reconstructs the visible surface and plausibly generates the invisible ones.

This representation choice also matters for data scaling. A pixel-aligned geometry representation can directly consume image-grid supervision, such as depth maps, rather than relying only on curated 3D assets. Most foundation image-to-3D models [80] learn from datasets of artist-designed or scanned 3D assets that can be challenging to scale up and diversify. Moreover, their image backbones are often designed as a global latent encoder followed by a separate 3D generator. This design can discard fine-grained image evidence and weaken the pixel-level alignment needed for faithful reconstruction. In contrast, a pixel-aligned design preserves local visual evidence throughout the geometry prediction process and allows the model to learn from image- and depth-supervised data at greater scale.

We introduce World Tracing (WT), a generative model of pixel-aligned multilayer geometry for this purpose. Building on the layered-depth lineage [27, 31, 60], for each input pixel WT predicts an ordered stack of L 3D points in camera space along the corresponding ray: the first layer is the visible surface, deeper layers complete the occluded geometry behind it. Faithful visible-surface reconstruction and generative completion are therefore not separate outputs, but successive layers of one tensor on the input pixel grid. The model predicts an image-grid pointmap and does not require camera intrinsics as input; when a pinhole intrinsics matrix is needed downstream, we fit a self-consistent K in closed form from the predicted layer-0 geometry.

We instantiate this representation with WT-DiT, a flow-matching diffusion transformer that treats multiple geometry layers as separate denoising tokens attending to each other. Since every layer lives on the image grid, the model inherits strong 2D visual priors from pre-trained image encoders, such as DINO [49] or MoGe [74], rather than learning solely from rendered 3D assets. Unlike layered regressors that must also predict which layers are valid, through a per-layer mask or a ray-stopping index [31], we use a depth-filling strategy: missing deeper-layer intersections are forward-filled in the target, while pixels outside the layer-0 alpha are noise-filled in the network input and ignored by the endpoint loss. This yields a single XYZ-only diffusion objective that trains all layers jointly. The same representation and core objective cover objects and scenes; the dynamic model only adds temporal attention.

We demonstrate three downstream uses of this representation in Sec. 5: text-driven 3D scene editing, geometry-conditioned novel-view video synthesis, and training-free textured-mesh generation, all built without further per-task 3D training.

We evaluate WT across object, scene, and dynamic benchmarks with metrics that measure not only plausibility but also geometric consistency. On held-out benchmarks rendered from our train-

ing sources (in-domain for WT), it improves over image-to-3D generators on Chamfer distance to ground-truth geometry, and its visible layer matches or exceeds dedicated monocular-depth predictors. On out-of-domain real benchmarks, its visible layer stays comparable to these predictors and it outperforms LaRI on complete scene geometry. Our contributions are:

1. A generative formulation of the layered pixel-aligned representation [27, 31, 60]: flow matching directly on camera-space XYZ layers, with no VAE, tokenizer, or canonical latent, so that visible-surface estimation and occluded-geometry completion share one camera-space tensor across objects, scenes, and dynamic clips.
2. WT-DiT, a flow-matching diffusion transformer with efficient three-way factorized attention, trained with a maskless depth-filling objective and a layer-aware noise curriculum.
3. A comprehensive evaluation across objects, scenes, and videos showing that multilayer generation improves complete geometry at no cost to visible-surface accuracy.
4. Downstream demonstrations: 3D scene editing, geometry-guided video synthesis, and training-free textured-mesh generation, showing WT's effectiveness as a geometry prior for 3D pipelines.

2 Related Work

Monocular and multi-view 3D reconstruction. Pixel-aligned geometry has evolved from scale-ambiguous monocular depth [12, 51, 54, 90, 91] and depth-as-diffusion variants [17, 19, 28] to pointmap predictors that recover scale and intrinsics implicitly [74, 75] and to multi-view reconstructors [30, 34, 73, 76]. All share one structural limit: a single 3D point per pixel, so geometry behind the first visible surface is absent.

Layered geometry. The layered representation that WT predicts is not new. Layered depth images (LDIs) [60] store multiple depth samples along each ray of a single camera for image-based rendering; they are built from known geometry or multiple views rather than predicted from one image. Learning-based methods later inferred occluded structure from a single image, either as explicit layers [25, 62] or through implicit functions [57, 58]. Our closest predecessors are LaRI [31], which regresses layered ray intersections for objects and indoor scenes together with a ray-stopping index that marks the valid layers, and DualPM [27], which regresses posed and canonical point maps and extends them to amodal layers for articulated animals. Both are deterministic regressors. We adopt their camera-space layered output and do not claim the representation itself as a contribution. Our contributions concern how it is learned: (i) a generative flow-matching model over all layers jointly, which represents the multi-modal distribution of occluded surfaces rather than a regressed mean; (ii) a maskless depth-filling objective that removes the validity mask or stopping index; (iii) mix-training that additionally consumes single-layer RGBD captures; and (iv) one architecture across objects, scenes, and dynamic clips, with geometry priors inherited from a frozen MoGe ViT-L encoder built on DINOv2 [49]. In a data-matched control, LaRI retrained on our corpus still trails WT on full geometry, and a regression variant of WT loses most of the full-geometry gain (App. G.3).

Image-to-3D generation. Image-to-3D pipelines fall into feed-forward generators [6, 20, 41, 80, 85], multi-view-then-reconstruct approaches [42, 45, 46, 61], and SDS optimization [38, 52, 68], with diffusion backbones over point clouds [48], RGB-D images [64], or structured latents [22, 80], and Gaussian outputs [67]. They produce complete geometry, but typically in a canonical object frame, losing pixel alignment with the input. WT instead keeps the camera-aligned pixel grid as the native coordinate system. Recent hybrids [5, 86] inject VGGT point clouds or features into TRELLIS; we compare against both as baselines for our training-free TRELLIS hybrid.

Video-to-4D and feed-forward 4D. Optimization-based monocular 4D methods fit per-sequence NeRF / Gaussian / mesh / skeleton representations [23, 37, 87–89, 96–98], often relying on test-time optimization and auxiliary trackers or part templates. Feed-forward 4D predictors are faster, but they typically model dynamic geometry as single-surface point clouds, canonical meshes or Gaussians, structured spacetime latents, or trajectory fields [35, 44, 56, 79, 83, 95]. In contrast, WT-D keeps the same pixel-aligned multilayer target as our static models and adds temporal attention to maintain coherence across frames.

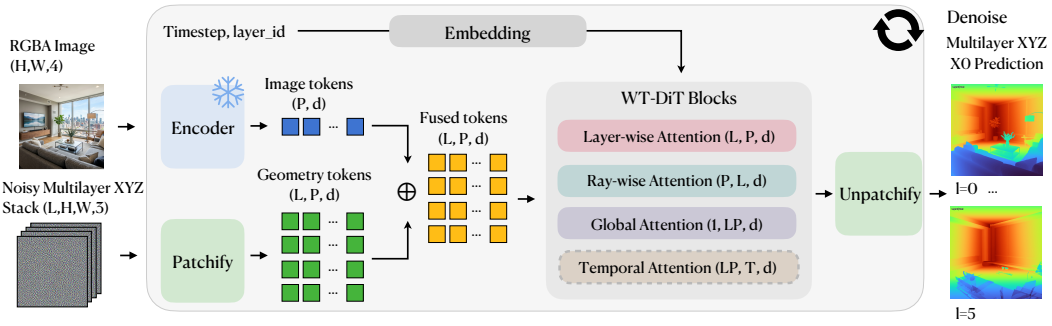

Figure 2: **WT-DiT architecture.** A frozen MoGe encoder provides pixel-aligned image features, while noisy multilayer XYZ is patchified into geometry tokens. Pixel-aligned fusion combines image and geometry tokens before DiT decoder blocks with layer-wise, ray-wise, and global self-attention, plus temporal attention for WT-D. A linear patch projection maps each decoder token to the XYZ of its 14×14 patch, followed by unpatchification to the full multilayer image grid.

3 Method

We introduce an image-to-3D pipeline that produces L layer pointmaps from a single image. A frozen MoGe ViT-L turns the input RGBA image into pixel-aligned image tokens, which are used to condition WT-DiT, a flow-matching diffusion transformer that operates directly on the same image grid. Starting from Gaussian noise on a multilayer XYZ tensor, WT-DiT integrates the flow ODE to produce a dense layered camera-space point stack: layer 0 recovers the visible surface, and deeper layers complete the occluded geometry along each pixel ray. Since every layer lives on the input pixel grid, the visual alignment with the input image and pixel-to-3D correspondences are preserved by construction.

3.1 World Tracing Representation

Given an RGBA input $I\in\mathbb{R}^{H\times W\times4}$ (an RGB image and a binary validity mask), WT predicts geometry as a pixel-aligned tensor of ordered 3D points

$$\mathbf{X} \in \mathbb{R}^{L\times H\times W\times3}, \qquad \mathbf{X}[\ell,\mathbf{u}] = \mathbf{x}_\ell(\mathbf{u}), \tag{1}$$

where $\mathbf{x}_\ell(\mathbf{u})$ is the camera-space 3D point at the ℓ-th front-to-back intersection of the ray through pixel $\mathbf{u}$. We do not require camera intrinsics as input: when needed, the parameters of a camera model can be recovered from the predicted layer-0 point cloud by fitting pixel-to-3D correspondences. The alpha channel of I marks which pixels are valid: for the object model it isolates the foreground, and for the scene model it masks out infinite-depth pixels such as sky.

This representation turns faithful 3D generation into a layer-indexed multi-image generation problem. Layer 0 is the visible surface and is tightly constrained by the observed pixels. Deeper layers correspond to hidden surfaces, which are less directly observed and therefore require increasingly more conditional generation. All predictions are made on the input pixel grid and expressed in the camera's 3D coordinate system, so the output preserves the pixel-to-3D correspondences and camera pose information that canonical-frame generators typically discard.

Dense targets without per-layer mask prediction. Depth peeling produces sparse pixels on later geometry layers: many rays have fewer than L true intersections. A direct multilayer formulation [21, 31] would require predicting both depth/XYZ coordinates and which layers are valid, through a binary mask per layer or a ray-stopping index. This is difficult because deeper layers are valid on only a small fraction of rays, creating a severe class imbalance and encouraging mask collapse. It also couples coordinate regression with visibility classification, which we find can produce conflicting training signals. Rather than ask the network to predict a separate visibility mask for each deeper layer, we make the supervision target dense by forward-filling empty intersections. If position $(\ell,\mathbf{u})$ has zero intersection, we fill it from the nearest earlier valid layer:

$$\mathbf{x}_\ell(\mathbf{u}) \leftarrow \mathbf{x}_{\ell'}(\mathbf{u}), \qquad \ell' = \max\{\, k<\ell : \mathbf{x}_k(\mathbf{u}) \text{ is valid} \,\}. \tag{2}$$

Let $\bar{\mathbf{X}}$ denote the resulting dense multilayer XYZ target. Every valid input ray therefore supervises all layers; the recovered shape is unchanged because filled entries are repeated points rather than new geometry. Modeling overlapping intersection points can express that some rays have already terminated by predicting deeper layers that collapse onto the front surface, avoiding the severe class imbalance of predicting sparse validity masks for deeper layers (Sec. 4.3); termination remains decodable from the predicted inter-layer gaps (App. G.4).

Scale normalization. Raw camera-space coordinates span very different ranges for objects and scenes, so we train in a normalized coordinate frame, $\tilde{\mathbf{X}} = \mathcal{N}(\bar{\mathbf{X}})$, using an invertible mapping chosen per regime. For objects, we use a dataset-level per-channel z-score normalization to standardize the target data to unit normal distribution; for scenes, whose depth can vary by orders of magnitude across samples, we keep them bounded by applying a per-sample median normalization, followed by a signed log transform. The same architecture predicts the normalized tensor in both regimes; only the coordinate transform changes (App. A).

Flow-matching objective. We train WT-DiT with flow matching [13, 40, 43, 69] directly on the normalized coordinate signal rather than on a learned VAE latent [32]. We set the clean endpoint to $\mathbf{x}_0 = \tilde{\mathbf{X}}$, sample $\mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $t \sim p_{\text{train}}(t)$, and form $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1$. Let $\mathbf{f}_I$ denote the projected MoGe feature grid. Architecturally, $\mathbf{f}_I$ is fused pixel-wise with the noisy geometry tokens before the decoder, rather than supplied through a separate conditioning branch. The network outputs an $\mathbf{x}_0$-parameterized prediction $\hat{\mathbf{x}}_0 = F_\theta(\mathbf{x}_t^{\text{net}}, t, \mathbf{f}_I)$ trained with the endpoint loss

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_1, t}\Big[\big\| A \odot \big(F_\theta(\mathbf{x}_t^{\text{net}}, t, \mathbf{f}_I) - \mathbf{x}_0 \big) \big\|_2^2 \Big], \tag{3}$$

where A is the alpha validity mask broadcast over layers and XYZ channels, so pixels outside the valid region are ignored by the loss. During training, the network input $\mathbf{x}_t^{\text{net}}$ replaces the noisy geometry at invalid pixels with max Gaussian noise so the network learns to ignore them (App. A). At inference, the endpoint prediction induces the flow-ODE velocity $\mathbf{v}_\theta(\mathbf{x}_t, t, \mathbf{f}_I) = (\mathbf{x}_t - F_\theta(\mathbf{x}_t^{\text{net}}, t, \mathbf{f}_I))/t$, which we integrate from max noise towards $t{=}0$ using 20 ODE steps. The full loss adds a soft adjacent-layer monotonicity penalty $\mathcal{L}_{\text{mono}}$ that pushes adjacent layers toward front-to-back order in the normalized coordinate space; because our normalization maps are monotone in depth, they preserve the layer ordering (App. A). There is no per-layer mask, silhouette, or visibility classification head.

3.2 WT-DiT Architecture

WT-DiT is a flow-matching diffusion transformer that maps an input image and a noisy multilayer point stack to an x_0 prediction on $\mathbb{R}^{L\times H\times W\times 3}$. Following recent feed-forward 3D transformers [73, 76], the design is largely standard: a frozen 2D foundation encoder produces pixel-level evidence, and a stack of pre-norm DiT blocks predicts the full layered pointmaps. We introduce two design choices specific to our multilayer representation: a three-way attention factorization (within-layer, along-ray, and global) and a FiLM layer embedding that breaks the layer permutation symmetry.

Encoder and tokenization. Image evidence comes from a frozen MoGe ViT-L [74]; only a feature projection from its last blocks is trained, so WT-DiT inherits MoGe's in-the-wild visual priors rather than learning them from rendered geometry. The noisy XYZ tensor is patchified on the same pixel grid (one geometry token per patch, per layer; $L{=}6$, discussed in App. H.1); at every $(\ell, \mathbf{u})$ the noisy geometry is concatenated with the repeated image feature and projected to the decoder width, so image evidence and geometry state stay in correspondence throughout the decoder without a separate cross-attention path. Hyperparameters and parameter counts are in App. B.

Three-way attention factorization. Full attention over all $L \times P$ tokens is unstructured and costly, so the decoder cycles through three attention shapes: **layer-wise** attention $(B{\cdot}L,\ P,\ D)$, where each layer attends within itself as a 2D image with 2D RoPE over (y, x); **ray-wise** attention $(B{\cdot}P,\ L,\ D)$, where tokens at the same pixel attend along the front-to-back layer axis, enforcing depth ordering and layer coherence at each ray; and **global** attention $(B,\ LP,\ D)$ that recovers object- or scene-level context at higher cost. Compared to the frame/global alternation used by VGGT [73] for multi-view inputs, the explicit ray axis couples the layers of each ray at almost no cost: a capacity-matched VGGT-style variant that replaces it with layer-wise attention reaches similar quality while spending about $216\times$ more attention compute in the replaced blocks (App. G.3).

Layer-aware conditioning. Each token must know which layer it represents and which diffusion time it is being denoised at. **Layer FiLM** maps a per-layer embedding $\mathbf{e}_\ell$ through an MLP to channel-wise $(\gamma_\ell, \beta_\ell)$ and applies feature-wise modulation $h \leftarrow \gamma_\ell \odot h + \beta_\ell$, which breaks the layer permutation symmetry and outperforms a capacity-matched additive layer embedding (App. G.3). The diffusion time t uses the standard AdaLN modulation [13] shared across all tokens of the stack.

Temporal attention for dynamic clips. For WT-D, we keep the static decoder unchanged and insert one temporal-attention block after each global-attention block, with 1D RoPE along the time axis and a small LayerScale init [70] so that the single-frame WT-O checkpoint reproduces the static behavior on $T{=}1$ inputs and gradually picks up temporal coupling during fine-tuning (App. B).

3.3 Training and Model Variants

Model variants. We train three variants of the same model family with $L{=}6$ layers: WT-O (static objects), WT-S (static scenes), and WT-D (dynamic objects). They share the representation, flow-matching loss, depth filling, frozen image encoder, and structured attention; they differ only in the scale normalization (z-score for object/dynamic, log-median for scenes) and the temporal blocks added to WT-D. To stay robust to imperfect masks, we jitter the validity-mask boundary during training. Pixels whose support changes under this jitter are supervised with weak pseudo-targets obtained from the nearest valid rendered geometry after augmentation, and their losses are down-weighted. Downstream applications use the self-consistent intrinsics estimated from the layer-0 pointmap (App. B), whose accuracy we evaluate in App. G.6.

Training noise curriculum. The visible layer and the occluded layers have different uncertainty profiles. Layer 0 is strongly constrained by the input image and behaves more like a reconstruction target, while deeper layers are only indirectly constrained and require conditional generation. A single diffusion-time distribution therefore under-serves one of these regimes. We use a layer-aware curriculum: early in training, the visible layer is sampled from a plateaued logit-normal distribution and deeper layers from the standard logit-normal [13]; once all layers have stabilized, the whole stack switches to a shared timestep drawn from an equal mixture of the two schedules. This respects the different uncertainty profiles of visible and occluded geometry while preserving a single objective. Analytic densities of the three schedules are visualized in App. B.

3.4 Data Pipeline

Depth-peeling supervision. Training the multilayer representation requires per-ray, ordered ray-surface intersections, which are not provided by datasets curated for visible-surface depth or mesh generation. We therefore construct multilayer supervision by rasterizing curated 3D assets with depth peeling [14, 29, 47]. For each rendered image, the first L front-to-back intersections along every camera ray are unprojected into camera-space XYZ targets aligned with the tensor predicted by WT. Rays with fewer than L intersections are populated using the forward-filling rule of Eq. 2.

Training corpora and augmentations. We build three corpora with the same representation: approximately 300K objects from public collections; scene frames from the public 3D-FRONT split plus an internal scene corpus, whose held-out split is used only for evaluation; and approximately 16.8K animated assets. All renderings use randomized lighting, viewpoints, and intrinsics, and are combined with online geometric, photometric, and validity-mask augmentations whose camera-space targets are transformed jointly with the image. We provide asset sources, rendering settings, augmentation details, and per-layer occupancy statistics in App. C.

3.5 Mix-Training Across Multilayer and Single-Layer Supervision

Prior single-image depth models are usually tied to one supervision regime. Monocular depth and pointmap predictors [39, 75, 77] expose only a single visible layer, while image-to-3D models such as TRELLIS and layered regressors such as LaRI [31] require full 3D geometry. This makes it difficult to train one model jointly from both full 3D asset geometry and single-layer RGBD captures. Our representation supports both regimes without changing the architecture. Let $b_{\text{single}} \in \{0, 1\}$ indicate whether a sample provides only visible-surface supervision or full multilayer supervision. We gate the loss-validity mask A of Eq. 3 along the layer axis:

$$A^{(\ell)} \;\leftarrow\; A^{(\ell)} \left(1 - b_{\text{single}} \mathbb{1}[\ell \geq 1]\right). \tag{4}$$

Table 1: **Object geometry** on 100 held-out assets rendered from our training sources (in-domain for WT-O). Left: visible-surface depth after the same per-sample scale–shift alignment for all methods. Right: full object geometry and TRELLIS hybridization; every entry is registered to the ground truth with the same per-sample Sim(3) alignment (Umeyama + ICP) at a matched budget of 50K points before Chamfer/F-score (App. G.1). Stochastic methods are reported as oracle best-of-8 random seeds; the gray row gives the non-oracle mean over the same seeds (std and the visible-surface mean, AbsRel 0.0089, in App. G.2). DA3 denotes Depth Anything 3; Pi3X is the upgraded π^3 model; 3DGS denotes 3D Gaussians; PC denotes point clouds directly produced by WT-O; WT-O* denotes the 3D assets produced by combining WT-O with TRELLIS.2's Stage 2 (detailed geometry generation) model. Best Second best Third best.

Visible surface depth

Method	MAE↓	RMSE↓	AbsRel↓	δ<1.25 ↑	δ<1.25^2 ↑
DA3 [39]	0.0703	0.0920	0.0384	0.9973	0.9998
LaRI [31]	0.0366	0.0506	0.0198	0.9992	**0.9999**
Pi3X [77]	0.0317	0.0440	0.0172	0.9994	**0.9999**
MoGe-2 [75]	0.0261	0.0368	0.0141	0.9995	**0.9999**
VGGT [73]	0.0257	0.0370	0.0138	0.9995	**0.9999**
WT-O	**0.0149**	**0.0243**	**0.0079**	**0.9996**	**0.9999**

Image-to-3D (full geometry)

Method	L1↓	L2↓	F@0.01↑	F@0.05↑	Output
TRELLIS.2 [81]	0.0566	0.00717	0.204	0.598	Mesh
SAM 3D [6]	0.0475	0.00501	0.203	0.675	3DGS
LaS-Comp [86]	0.0477	0.00739	0.225	0.677	Mesh
ReconViaGen [5]	0.0478	0.00528	0.228	0.677	Mesh
WT-O*	0.0326	0.00530	0.321	0.808	Mesh
WT-O	**0.0213**	**0.00194**	**0.549**	**0.898**	PC
WT-O (mean-of-8)	0.0243	0.00221	0.497	0.876	PC

Table 2: **Scene geometry.** Visible-surface depth and point-cloud geometry after the same per-sample scale–shift alignment for all methods. DA3 denotes Depth Anything 3; Pi3X is the upgraded π^3 model; WT-S* denotes a scene model trained only on 3D-FRONT. CD-L1, CD-L2, and F-score are computed after unprojection; All-L metrics are only defined for multilayer methods. WT-S rows are oracle best-of-8; the non-oracle mean-of-8 for WT-S (AbsRel 0.0132 and All-L CD-L1 0.0236 on 3D-FRONT, All-L CD-L1 0.0791 on the internal set) leaves its ordering against every baseline unchanged (App. G.2). Both test sets are held out but in-domain (same sources as training) for WT-S; out-of-domain real benchmarks are in App. F.3 and App. G.5. Best Second best Third best.

Method	MAE↓	RMSE↓	AbsRel↓	L0 CD-L1↓	L0 CD-L2↓	L0 F@0.05↑	All-L CD-L1↓	All-L CD-L2↓	All-L F@0.05↑
3D-FRONT (held-out, 50 samples)									
VGGT [73]	0.0393	0.0568	0.0441	0.0345	0.002889	0.7839	–	–	–
DA3 [39]	0.0369	0.0521	0.0403	0.0330	0.002893	0.7992	–	–	–
LaRI-scene [31]	0.0319	0.0483	0.0359	0.0268	0.001994	0.8576	0.0575	0.009990	0.6671
MoGe-2 [75]	0.0248	0.0373	0.0269	0.0213	0.000973	0.9131	–	–	–
Pi3X [77]	0.0234	0.0375	0.0260	0.0192	0.000871	0.9206	–	–	–
WT-S*	0.0106	0.0220	0.0118	0.0097	0.000230	0.9845	0.0224	0.001350	0.8890
WT-S	**0.0102**	**0.0215**	**0.0114**	**0.0093**	**0.000204**	**0.9867**	**0.0216**	**0.001255**	**0.8951**
Internal test set (held out, 200 samples)									
VGGT [73]	0.0546	0.0835	0.0779	0.0452	0.017798	0.7502	–	–	–
LaRI-scene [31]	0.0542	0.0837	0.0769	0.0429	0.022125	0.7591	0.0872	0.052319	0.5565
DA3 [39]	0.0462	0.0727	0.0629	0.0387	0.005977	0.7772	–	–	–
MoGe-2 [75]	0.0395	0.0637	0.0561	0.0315	0.003969	0.8375	–	–	–
Pi3X [77]	0.0359	0.0622	0.0514	0.0304	0.003784	0.8479	–	–	–
WT-S	**0.0328**	**0.0586**	**0.0470**	**0.0278**	**0.002850**	**0.8720**	**0.0750**	**0.045000**	**0.6500**

Together with the depth-filling target, this per-sample gate lets a single WT model train concurrently on multilayer 3D-asset renders and single-layer RGBD captures. The data mix is described in App. C, and the resulting gain on real-scene visible-surface benchmarks is reported in App. F.3.

4 Experiments

We verify two hypotheses: our pixel-aligned multilayer paradigm yields (i) more faithful 3D geometry across objects, scenes, and dynamic clips, and (ii) geometry that benefits downstream pipelines depending on pose, correspondence, or disoccluded structure. A long-form supplementary video provides additional qualitative results across all regimes.

4.1 Experimental Setup

Data. The multilayer 3D-asset training mixture has three regimes: ~300K objects (~17M rendered views) from public collections including Objaverse-XL [9], Objaverse [10], 3D-FUTURE [16], Toys4k [65], GSO [11], and Truebones [71]; scene frames from the public 3D-FRONT corpus [15]

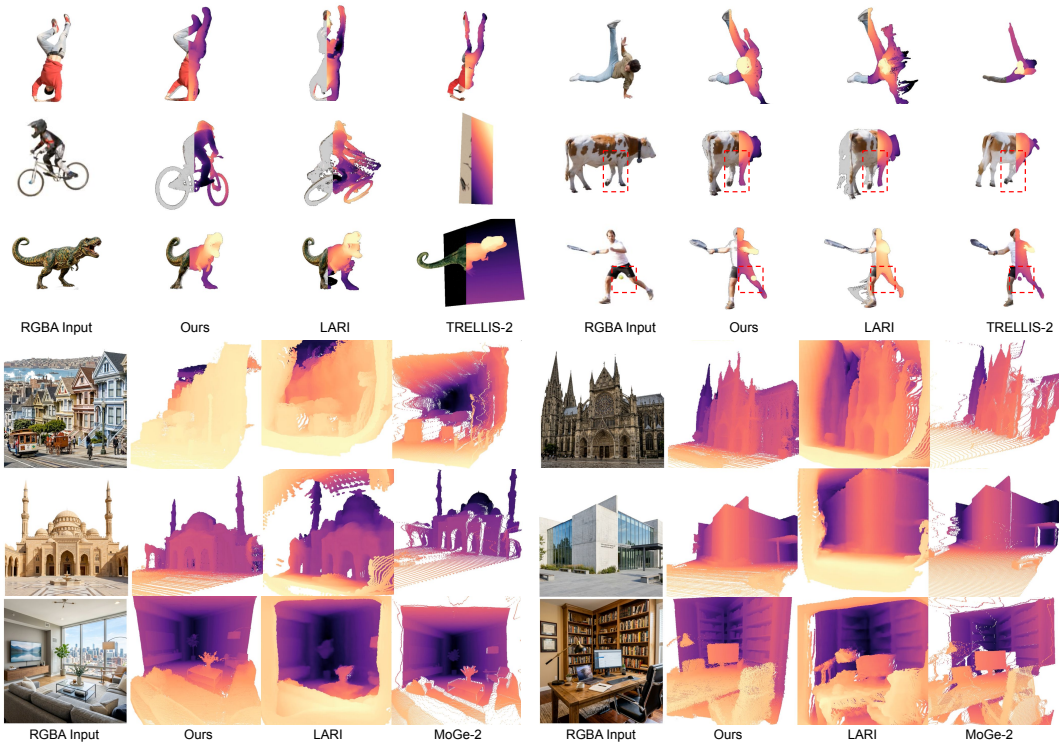

Figure 3: **Qualitative comparison on out-of-distribution inputs.** All inputs are deliberately drawn from outside our training distributions to probe generalization. Top: object examples comparing WT-O against LaRI-O and TRELLIS.2; inputs are real-world DAVIS [50] video frames and generated images, neither of which appear in our object training corpus. Bottom: scene examples comparing WT-S against LaRI-S and MoGe-2 on generated room images that are not drawn from any of our scene training sources. WT preserves pixel alignment while generating complete geometry, improving real-image generalization and avoiding several common failure modes of existing methods.

plus an internal scene corpus that supplies roughly half of the scene training frames; and ~16.8K animated assets sampled for WT-D. Via the mix-training of Sec. 3.5, WT-S can additionally consume a 12-dataset single-layer RGBD-style corpus (App. C). Every result in this section uses the 3D-asset-only checkpoints; only the real-scene depth comparison in App. F.3 adds the mix-trained WT-S as a separate row.

Training details. WT-DiT is a 1.7B-parameter (1.4B trainable) DiT trained at 504×504 resolution with $L{=}6$ layers, optimized with AdamW on 64 H100 GPUs (global batch size 512); WT-D is fine-tuned from the WT-O checkpoint. Inference uses 20 denoising steps and peaks at 9.4 GB of GPU memory (App. B, App. G.7).

Evaluation. We separate in-domain from out-of-domain evaluation. *In-domain* benchmarks are held-out splits rendered from the same sources as our training data: 100 held-out object assets, the held-out 3D-FRONT split, a 200-sample internal scene test set (a held-out split of the internal scene corpus from which no training frame is drawn), and the Obj.-Val and Truebone dynamic splits. No test asset or scene is seen in training, but the rendering pipeline and source distributions are shared, which favors WT over baselines trained on other data. *Out-of-domain* benchmarks share neither: ActionBench for dynamic clips, real-image depth on NYU Depth V2 and ETH3D (App. F.3), complete real-scene geometry on SCRREAM (App. G.5), and the qualitative comparisons on real and generated images (Fig. 3). Baselines cover dedicated depth predictors, layered predictors, image-to-3D generators, dynamic-geometry methods, and TRELLIS hybrids. **For stochastic methods, including WT, we draw $K{=}8$ random seeds per input and report the oracle best result**; non-oracle mean$\pm$std over the same seeds leaves every object and scene conclusion unchanged (App. G.2). Full split definitions and baseline lists are in App. C.

Table 3: **Dynamic geometry.** Global CD-L2 (↓). Obj.-Val and Truebone are in-domain held-out splits, ActionBench is out-of-domain; Mean is unweighted.

Benchmark	GVFD [95]	SS4D [35]	AM [56]	WT-D
ActionBench	0.0879	0.0882	**0.0243**	0.0291
Truebone	0.0166	0.0145	0.0120	**0.0063**
Obj.-Val	0.0248	0.0249	0.0142	**0.0034**
Mean	0.0431	0.0425	0.0168	**0.0129**

Table 4: **Timestep sampling ablation.** CD-L2 (↓) on the visible layer (L0), occluded layers (L1–L5), and all layers.

Schedule	L0	L1–L5	All
Plat. logit-normal	0.021	0.033	0.031
Logit-normal	0.023	0.028	0.027
Mixture	**0.020**	**0.025**	**0.024**

Metrics. We emphasize geometry faithfulness over aesthetic plausibility: standard depth errors on visible surfaces and Chamfer/F-score on complete geometry, with visible and occluded breakdowns where applicable. For full object geometry, every method, WT-O included, is registered to the ground truth by the same per-sample Sim(3) fit (Umeyama initialization [72] and 10 ICP iterations [2]), and Chamfer/F-score are computed on 50K points per side, so canonical-frame baselines are not penalized for their output convention (full protocol and sampling-variance checks in App. G.1). The TRELLIS hybrid is measured with the same geometry metrics; scene editing and view synthesis are qualitative demonstrations. Metric definitions are in App. C.

4.2 Faithful Geometry Generation

Objects. Table 1 evaluates the object model and Fig. 3 (top) shows qualitative comparisons on out-of-distribution inputs (DAVIS [50] video frames and generated images). Although WT-O predicts six layers, its layer 0 achieves the best visible-surface accuracy among all depth and pointmap baselines on MAE, RMSE, and AbsRel on this in-domain benchmark: generation of occluded geometry does not cost visible-surface faithfulness. WT-O also achieves the best complete-object geometry, improving over textured-mesh and 3D-Gaussian generators under identical Sim(3) registration (its non-oracle mean still beats every baseline's oracle best), and further improves TRELLIS.2 when used as its Stage-1 prior. The qualitative gains stem from two complementary factors over prior work: a pixel-aligned diffusion (vs. LaRI's regression head) better models the multi-modal uncertainty of occluded surfaces and avoids deep-layer mask collapse, and inheriting 2D image priors from the frozen MoGe encoder generalizes better to real images than canonical-frame asset-trained generators that can flip pose or hallucinate spurious planes (App. F.1).

Scenes. Table 2 reports scene geometry on the held-out 3D-FRONT benchmark and the 200-sample internal test set, with qualitative examples on out-of-distribution generated images in Fig. 3. After the same scale-and-shift-invariant (SSI) alignment used for all depth baselines, WT-S achieves the best visible-surface depth and L0 point-cloud geometry among the compared methods, and improves substantially over LaRI-scene on all-layer geometry, where single-layer baselines do not produce occluded layers. Both test sets are in-distribution in source for WT-S, and its margin narrows on the harder internal set. The 3D-FRONT-only WT-S* nearly matches WT-S on held-out 3D-FRONT, isolating the effect of the broader internal scene corpus. On the out-of-domain SCRREAM benchmark with complete real ground-truth meshes, WT-S also outperforms LaRI-scene on both L0 AbsRel (0.0331 vs. 0.0697) and All-L CD-L1 (0.058 vs. 0.073) (App. G.5). On NYU Depth V2 [63] and ETH3D [59], layer 0 is comparable to dedicated depth predictors and improves further with mix-training (App. F.3). Figure 6 in the appendix shows the full $L{=}6$ layer stack produced by WT-S.

Dynamic clips. Table 3 evaluates WT-D against GVFDiffusion (GVFD), SS4D, and ActionMesh (AM) on three complementary splits. Obj.-Val uses held-out Objaverse-XL animated objects and Truebone uses held-out rigged assets with explicit articulation (both in-domain), while ActionBench is an out-of-domain benchmark of action-driven motions. WT-D wins on the two in-domain splits and achieves the best mean CD-L2. AM is strongest on the out-of-domain ActionBench, where ground truth is tracked animated surfaces matching its native output format.

4.3 Discussion

Controlled ablations are in App. G.3; further design choices, generalization, and limitations are discussed in App. H.

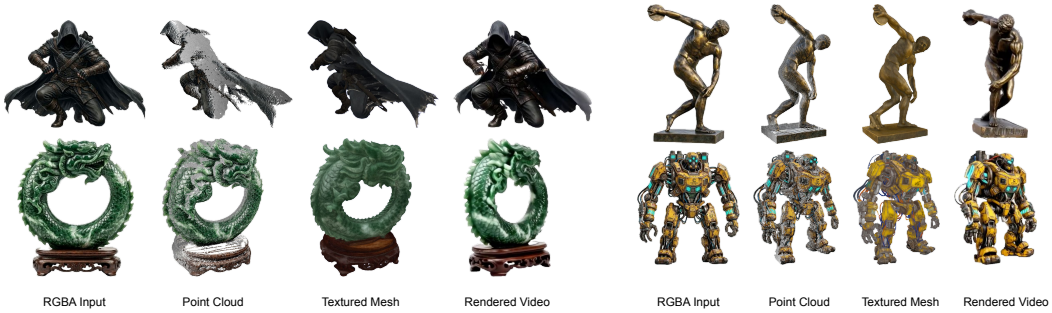

Figure 4: **Pixel-aligned geometry as a unified 3D interface.** WT generates complete multilayer geometry in the input camera frame while preserving pixel correspondences. This representation enables pose-aware structure for training-free textured-mesh pipelines such as TRELLIS-style decoders, and serves as geometry memory for novel-view video synthesis.

Depth filling rather than mask prediction. Predicting a validity mask per layer conflicts with XYZ regression (near-zero or negative gradient cosine similarity) and is severely imbalanced (8.14% valid area at L0 vs. 0.60% at L5), so mask heads collapse on deep layers (App. G.4). Forward filling turns this into dense XYZ regression: genuine and filled deep-layer pixels stay in the same error regime at every layer, and termination is recoverable from the predicted inter-layer gap with ROC-AUC 0.97.

Timestep sampling. Unlike single-layer geometry diffusion with a uniform schedule [84], the visible layer behaves like faithful prediction while occluded layers behave like conditional generation, so one diffusion-time distribution is suboptimal. Plateaued logit-normal sampling helps L0 but hurts deeper layers; the mixture is best overall (Table 4).

What drives the gains. In data-matched controls (App. G.3), LaRI retrained on our corpus improves but still trails WT (All-L CD-L1 0.0407 vs. 0.0276), and a single-step regression variant of WT matches the visible layer but is 33% worse on full geometry. A single-layer ($L{=}1$) control shows that adding occluded layers does not degrade, and slightly improves, layer-0 accuracy. No single factor explains the gap: data, objective, and architecture each contribute.

5 Downstream Pipeline Demonstrations

Because WT outputs complete geometry in the input camera frame with pixel-level correspondence, one prediction supports three training-free pipelines (full details in App. E; more examples in the supplementary video). **3D scene editing:** a 2D editor performs the edit in image space; WT-S predicts the original scene, WT-O the inserted or replaced content, and the two stacks compose in closed form on the shared pixel grid, supporting insertion, removal, and replacement without per-edit optimization (Fig. 1). **Geometry-guided video synthesis:** video models conditioned on visible-surface depth tracks [4, 24, 36, 93] must hallucinate disocclusions once the camera moves; rasterizing WT's multilayer cloud along the trajectory supplies guidance with the back side already present. **Pose-aligned TRELLIS hybrid:** we voxelize WT's camera-space stack as TRELLIS' Stage-1 structure and keep its decoders unchanged, which gives a stronger Stage-1 prior than TRELLIS.2 or VGGT-guided alternatives [5, 86] (Table 1) with meshes aligned to the input image (Fig. 4).

6 Conclusion

We presented World Tracing, a generative model of pixel-aligned multilayer geometry for faithful generation, where visible-surface fidelity and occluded-geometry completion become successive layers of one camera-space tensor. WT-DiT learns this representation with a frozen 2D foundation encoder, flow matching, and an XYZ-only depth-filling objective, and applies across objects, scenes, and dynamic clips. The resulting geometry also serves as a 2D-to-3D interface for scene editing, view synthesis, and pose-aligned textured mesh generation. Its main limitations are a fixed layer count, multilayer supervision that is still synthetic, and iterative sampling cost (App. H.6). We hope WT becomes a common substrate for 3D-aware perception, generation, editing, and video.

Acknowledgments and Disclosure of Funding

This work was done while Hao Zhang was an intern at World Labs. We gratefully acknowledge the support of the Office of Naval Research under grant N00014-24-1-2169, the USDA National Institute of Food and Agriculture under grant AFRI 2020-67021-32799/1024178, and the Amazon–Illinois Center on AI for Interactive Conversational Experiences. This research used the Delta advanced computing and data resource [18], which is supported by the National Science Foundation (award OAC 2005572) and the State of Illinois. Delta is a joint effort of the University of Illinois Urbana-Champaign and its National Center for Supercomputing Applications.

Competing interests. Mohamed El Banani, Jen-Hao Cheng, Paul Zhang, Yi Hua, Ben Mildenhall, Christoph Lassner, and Gengshan Yang are employees of World Labs, and Hao Zhang was an intern at World Labs during this work. Narendra Ahuja has no affiliation with World Labs. The authors declare no other competing interests.

References

[1] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, and Elad Shulman. ARKitScenes: A diverse real-world dataset for 3D indoor scene understanding using mobile RGB-D data. In *NeurIPS Datasets and Benchmarks Track*, 2021.

[2] Paul J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992. doi: 10.1109/34.121791.

[3] Black Forest Labs. FLUX.2: Analyzing and enhancing the latent space of FLUX – representation comparison. `https://bfl.ai/techblog/representation-comparison/`, 2025.

[4] Wei Cao, Hao Zhang, Fengrui Tian, Yulun Wu, Yingying Li, Shenlong Wang, Ning Yu, and Yaoyao Liu. FreeOrbit4D: Training-free arbitrary camera redirection for monocular videos via foreground-complete 4D reconstruction. In *SIGGRAPH Conference Papers*, 2026. doi: 10.1145/3799902.3811122.

[5] Jiahao Chang, Chongjie Ye, Yushuang Wu, Yuantao Chen, Yidan Zhang, Zhongjin Luo, Chenghong Li, Yihao Zhi, and Xiaoguang Han. ReconViaGen: Towards accurate multi-view 3D object reconstruction via generation. In *ICLR*, 2026.

[6] Xingyu Chen, Fu-Jen Chu, Pierre Gleize, Kevin J. Liang, Alexander Sax, Hao Tang, Weiyao Wang, Michelle Guo, Thibaut Hardin, Xiang Li, Aohan Lin, Jia-Wei Liu, Ziqi Ma, Anushka Sagar, Bowen Song, Xiaodong Wang, Jianing Yang, Bowen Zhang, Piotr Dollár, Georgia Gkioxari, Matt Feiszli, and Jitendra Malik. SAM 3D: 3Dfy anything in images. In *CVPR*, 2026.

[7] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *CVPR*, 2017.

[8] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *NeurIPS*, 2022.

[9] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-XL: A universe of 10M+ 3D objects. In *NeurIPS*, 2023.

[10] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3D objects. In *CVPR*, 2023.

[11] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B. McHugh, and Vincent Vanhoucke. Google Scanned Objects: A high-quality dataset of 3D scanned household items. In *ICRA*, 2022.

[12] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NeurIPS*, 2014.

[13] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*, 2024.

[14] Cass Everitt. Interactive order-independent transparency. White paper, NVIDIA Corporation, 2001.

[15] Huan Fu, Bowen Cai, Lin Gao, Lingxiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, and Hao Zhang. 3D-FRONT: 3D furnished rooms with layouts and semantics. In *ICCV*, 2021.

[16] Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Steve Maybank, and Dacheng Tao. 3D-FUTURE: 3D furniture shape with texture. *IJCV*, 2021.

[17] Xiao Fu, Wei Yin, Mu Hu, Kaixuan Wang, Yuexin Ma, Ping Tan, Shaojie Shen, Dahua Lin, and Xiaoxiao Long. GeoWizard: Unleashing the diffusion priors for 3D geometry estimation from a single image. In *ECCV*, 2024.

[18] William Gropp, Tim Boerner, Brett Bode, and Greg Bauer. Delta: Balancing GPU performance with advanced system interfaces. Technical report, University of Illinois Urbana-Champaign, 2023. URL `https://hdl.handle.net/2142/117179`.

[19] Jing He, Haodong Li, Wei Yin, Yixun Liang, Leheng Li, Kaiqiang Zhou, Hongbo Zhang, Bingbing Liu, and Ying-Cong Chen. Lotus: Diffusion-based visual foundation model for high-quality dense prediction. In *ICLR*, 2025.

[20] 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 *ICLR*, 2024.

[21] Tao Hu, Wenhang Ge, Yuyang Zhao, and Gim Hee Lee. X-Ray: A sequential 3D representation for generation. In *NeurIPS*, pages 136193–136219, 2024.

[22] Tao Hu, Fangzhou Hong, and Ziwei Liu. StructLDM: Structured latent diffusion for 3D human generation. In *ECCV*, 2024.

[23] Yanqin Jiang, Li Zhang, Jin Gao, Weiming Hu, and Yao Yao. Consistent4D: Consistent 360° dynamic object generation from monocular video. In *ICLR*, 2024.

[24] Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. VACE: All-in-one video creation and editing. In *ICCV*, 2025.

[25] Ziyu Jiang, Buyu Liu, Samuel Schulter, Zhangyang Wang, and Manmohan Chandraker. Peek-a-boo: Occlusion reasoning in indoor scenes with plane representations. In *CVPR*, 2020.

[26] HyunJun Jung, Weihang Li, Shun-Cheng Wu, William Bittner, Nikolas Brasch, Jifei Song, Eduardo Pérez-Pellitero, Zhensong Zhang, Arthur Moreau, Nassir Navab, and Benjamin Busam. SCRREAM: SCan, register, REnder and map: A framework for annotating accurate and dense 3D indoor scenes with a benchmark. In *NeurIPS Datasets and Benchmarks Track*, pages 44164–44176, 2024.

[27] Ben Kaye, Tomas Jakab, Shangzhe Wu, Christian Rupprecht, and Andrea Vedaldi. DualPM: Dual Posed-Canonical point maps for 3D shape and pose reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6425–6435, June 2025.

[28] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *CVPR*, 2024.

[29] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Trans. Graph.*, 2020.

[30] Vincent Leroy, Yohann Cabon, and Jerome Revaud. Grounding image matching in 3D with MASt3R. In *ECCV*, 2024.

[31] Rui Li, Biao Zhang, Zhenyu Li, Federico Tombari, and Peter Wonka. LaRI: Layered ray intersections for single-view 3D geometric reasoning. In *ICML*, 2026.

[32] Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *CVPR*, 2026.

[33] Zhengqi Li and Noah Snavely. MegaDepth: Learning single-view depth prediction from Internet photos. In *CVPR*, 2018.

[34] Zhengqi Li, Richard Tucker, Forrester Cole, Qianqian Wang, Linyi Jin, Vickie Ye, Angjoo Kanazawa, Aleksander Holynski, and Noah Snavely. MegaSaM: Accurate, fast and robust structure and motion from casual dynamic videos. In *CVPR*, 2025.

[35] Zhibing Li, Mengchen Zhang, Tong Wu, Jing Tan, Jiaqi Wang, and Dahua Lin. SS4D: Native 4D generative model via structured spacetime latents. *ACM Trans. Graph.*, 2025.

[36] Hanwen Liang, Junli Cao, Vidit Goel, Guocheng Qian, Sergei Korolev, Demetri Terzopoulos, Konstantinos N. Plataniotis, Sergey Tulyakov, and Jian Ren. Wonderland: Navigating 3D scenes from a single image. In *CVPR*, 2025.

[37] Ting-Hsuan Liao, Haowen Liu, Yiran Xu, Songwei Ge, Gengshan Yang, and Jia-Bin Huang. PAD3R: Pose-aware dynamic 3D reconstruction from casual videos. In *SIGGRAPH Asia Conference Papers*, 2025. doi: 10.1145/3757377.3763836.

[38] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3D: High-resolution text-to-3D content creation. In *CVPR*, 2023.

[39] Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Yang Zhao, Sida Peng, Hengkai Guo, Xiaowei Zhou, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. In *ICLR*, 2026.

[40] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *ICLR*, 2023.

[41] Minghua Liu, Ruoxi Shi, Linghao Chen, Zhuoyang Zhang, Chao Xu, Xinyue Wei, Hansheng Chen, Chong Zeng, Jiayuan Gu, and Hao Su. One-2-3-45++: Fast single image to 3D objects with consistent multi-view generation and 3D diffusion. In *CVPR*, 2024.

[42] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3D object. In *ICCV*, 2023.

[43] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.

[44] Xinhang Liu, Yuxi Xiao, Donny Y. Chen, Jiashi Feng, Yu-Wing Tai, Chi-Keung Tang, and Bingyi Kang. Trace anything: Representing any video in 4D via trajectory fields. In *ICLR*, 2026.

[45] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. SyncDreamer: Generating multiview-consistent images from a single-view image. In *ICLR*, 2024.

[46] Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, and Wenping Wang. Wonder3D: Single image to 3D using cross-domain diffusion. In *CVPR*, 2024.

[47] Abraham Mammen. Transparency and antialiasing algorithms implemented with the virtual pixel maps technique. *IEEE Computer Graphics and Applications*, 9(4):43–55, 1989.

[48] Luke Melas-Kyriazi, Christian Rupprecht, and Andrea Vedaldi. PC^2: Projection-conditioned point cloud diffusion for single-image 3D reconstruction. In *CVPR*, 2023.

[49] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *TMLR*, 2024.

[50] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016.

[51] Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. UniDepth: Universal monocular metric depth estimation. In *CVPR*, 2024.

[52] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D diffusion. In *ICLR*, 2023.

[53] Aashish Rai, Dilin Wang, Mihir Jain, Nikolaos Sarafianos, Kefan Chen, Srinath Sridhar, and Aayush Prakash. UVGS: Reimagining unstructured 3D Gaussian splatting using UV mapping. In *CVPR*, pages 5927–5937, 2025.

[54] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *TPAMI*, 2022.

[55] 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 *ICCV*, 2021.

[56] Remy Sabathier, David Novotny, Niloy J. Mitra, and Tom Monnier. ActionMesh: Animated 3D mesh generation with temporal 3D diffusion. In *CVPR*, pages 34312–34321, 2026.

[57] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. PIFu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *ICCV*, pages 2304–2314, 2019.

[58] Shunsuke Saito, Tomas Simon, Jason Saragih, and Hanbyul Joo. PIFuHD: Multi-level pixel-aligned implicit function for high-resolution 3D human digitization. In *CVPR*, 2020.

[59] Thomas Schöps, Johannes L. Schönberger, 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 *CVPR*, 2017.

[60] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 231–242. ACM, 1998. doi: 10.1145/280814.280882.

[61] Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multi-view diffusion base model. *arXiv preprint arXiv:2310.15110*, 2023.

[62] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 3D photography using context-aware layered depth inpainting. In *CVPR*, 2020.

[63] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, 2012.

[64] Gabriela Ben Melech Stan, Diana Wofk, Scottie Fox, Alex Redden, Will Saxton, Jean Yu, Estelle Aflalo, Shao-Yen Tseng, Fabio Nonato, Matthias Muller, and Vasudev Lal. LDM3D: Latent diffusion model for 3D. *arXiv preprint arXiv:2305.10853*, 2023.

[65] Stefan Stojanov, Anh Thai, and James M. Rehg. Using shape to categorize: Low-shot learning with an explicit shape bias. In *CVPR*, 2021.

[66] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo Open dataset. In *CVPR*, 2020.

[67] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. LGM: Large multi-view Gaussian model for high-resolution 3D content creation. In *ECCV*, 2024.

[68] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. DreamGaussian: Generative Gaussian splatting for efficient 3D content creation. In *ICLR*, 2024.

[69] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *TMLR*, 2024.

[70] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *ICCV*, 2021.

[71] Truebones Motions Animation Studios. Truebones motions animation studios. `https://truebones.com`, 2024.

[72] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991. doi: 10.1109/34.88573.

[73] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *CVPR*, 2025.

[74] Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. MoGe: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. In *CVPR*, 2025.

[75] Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. MoGe-2: Accurate monocular geometry with metric scale and sharp details. In *NeurIPS*, 2025.

[76] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *CVPR*, 2024.

[77] 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 *ICLR*, 2026.

[78] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *NeurIPS Datasets and Benchmarks Track*, 2021.

[79] Zijie Wu, Chaohui Yu, Fan Wang, and Xiang Bai. AnimateAnyMesh: A feed-forward 4D foundation model for text-driven universal mesh animation. In *ICCV*, pages 13557–13568, 2025.

[80] Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3D latents for scalable and versatile 3D generation. In *CVPR*, 2025.

[81] Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, and Jiaolong Yang. Native and compact structured latents for 3D generation. In *CVPR*, 2026.

[82] Tiange Xiang, Kai Li, Chengjiang Long, Christian Häne, Peihong Guo, Scott Delp, Ehsan Adeli, and Li Fei-Fei. Repurposing 2D diffusion models with Gaussian atlas for 3D generation. In *ICCV*, pages 16492–16502, 2025.

[83] Yuxi Xiao, Jianyuan Wang, Nan Xue, Nikita Karaev, Yuri Makarov, Bingyi Kang, Xing Zhu, Hujun Bao, Yujun Shen, and Xiaowei Zhou. SpatialTrackerV2: Advancing 3D point tracking with explicit camera motion. In *ICCV*, pages 6726–6737, 2025.

[84] Gangwei Xu, Haotong Lin, Hongcheng Luo, Haiyang Sun, Bing Wang, Guang Chen, Sida Peng, Hangjun Ye, and Xin Yang. Pixel-perfect visual geometry estimation. *IEEE TPAMI*, 2026. doi: 10.1109/TPAMI.2026.3727248.

[85] Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. InstantMesh: Efficient 3D mesh generation from a single image with sparse-view large reconstruction models. *arXiv preprint arXiv:2404.07191*, 2024.

[86] Weilong Yan, Haipeng Li, Hao Xu, Nianjin Ye, Yihao Ai, Shuaicheng Liu, and Jingyu Hu. LaS-Comp: Zero-shot 3D completion with latent-spatial consistency. In *CVPR*, 2026.

[87] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Huiwen Chang, Deva Ramanan, William T. Freeman, and Ce Liu. LASR: Learning articulated shape reconstruction from a monocular video. In *CVPR*, 2021.

[88] Gengshan Yang, Minh Vo, Natalia Neverova, Deva Ramanan, Andrea Vedaldi, and Hanbyul Joo. BANMo: Building animatable 3D neural models from many casual videos. In *CVPR*, 2022.

[89] Gengshan Yang, Shuo Yang, John Z. Zhang, Zachary Manchester, and Deva Ramanan. PPR: Physically plausible reconstruction from monocular videos. In *ICCV*, 2023.

[90] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *CVPR*, 2024.

[91] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything V2. In *NeurIPS*, 2024.

[92] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. BlendedMVS: A large-scale dataset for generalized multi-view stereo networks. In *CVPR*, 2020.

[93] Mark Yu, Wenbo Hu, Jinbo Xing, and Ying Shan. TrajectoryCrafter: Redirecting camera trajectory for monocular videos via diffusion models. In *ICCV*, 2025.

[94] Amir R. Zamir, Alexander Sax, William Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *CVPR*, 2018.

[95] Bowen Zhang, Sicheng Xu, Chuxin Wang, Jiaolong Yang, Feng Zhao, Dong Chen, and Baining Guo. Gaussian variation field diffusion for high-fidelity video-to-4D synthesis. In *ICCV*, pages 12502–12513, 2025.

[96] Hao Zhang, Fang Li, Samyak Rawlekar, and Narendra Ahuja. Learning implicit representation for reconstructing articulated objects. In *ICLR*, 2024.

[97] Hao Zhang, Fang Li, Samyak Rawlekar, and Narendra Ahuja. S3O: A dual-phase approach for reconstructing dynamic shape and skeleton of articulated objects from single monocular video. In *ICML*, pages 59191–59209, 2024.

[98] Hao Zhang, Di Chang, Fang Li, Mohammad Soleymani, and Narendra Ahuja. MagicPose4D: Crafting articulated models with appearance and motion control. *TMLR*, 2025.

Technical Appendices and Supplemental Material

This supplementary document provides details deferred from the main paper:

- App. A: method details (scale normalization, invalid-pixel noise fill, monotonicity penalty).
- App. B: training schedule, optimizer, model hyperparameters, decoder details, and the layer-aware diffusion-time schedule.
- App. C: rendering pipeline, source datasets, the RGBD-style mix-training corpus, evaluation splits, baselines, and metric definitions.
- App. D: per-layer validity statistics on the object corpus.
- App. E: full pipelines for the three downstream applications.
- App. F: qualitative analysis, the multilayer depth gallery, and real-scene depth benchmarks with mix-training.
- App. G: additional experiments: full-geometry registration protocol, non-oracle results, controlled ablations and a data-matched LaRI comparison, filled-layer analyses, the out-of-domain SCREAM benchmark, intrinsics recovery, and computational cost.
- App. H: extended discussion on layer count, prediction target, parameterization, architecture details, generalization, limitations, and future work.

A Method Details

This appendix expands Sec. 3.1 with details deferred from the main paper.

Scale normalization. Raw camera-space coordinates span very different ranges for objects and scenes, so we normalize XYZ with a reversible map chosen per regime. For object and dynamic-object data, whose scale is controlled by rendering, we use a global per-channel z-score

$$\tilde{\mathbf{x}} \;=\; (\mathbf{x} - \boldsymbol{\mu}) \,/\, \boldsymbol{\sigma}, \tag{5}$$

where $\boldsymbol{\mu}, \boldsymbol{\sigma} \in \mathbb{R}^3$ are channel-wise statistics computed once on the training corpus. For scenes, where depth can vary by orders of magnitude within a trajectory, we use a per-sample log-median map. Let m be the median valid depth over the multilayer stack:

$$\tilde{z} \;=\; \ln(z/m), \qquad \tilde{x} \;=\; \operatorname{sign}(x)\,\ln(1+|x|/m), \qquad \tilde{y} \;=\; \operatorname{sign}(y)\,\ln(1+|y|/m). \tag{6}$$

Both maps are reversible, so predictions can be returned to camera-space metric coordinates at inference. The same architecture predicts the normalized tensor in both regimes; only the input/output coordinate transform changes.

Invalid input pixels and noise fill. The input image contains invalid pixels outside the layer-0 silhouette support (e.g., the background of a segmented object, or sky pixels in a scene). Let $A(\mathbf{u}) \in \{0,1\}$ denote the layer-0 alpha validity, broadcast across all L layers and XYZ channels. These invalid pixels are excluded from the endpoint loss in Eq. 3. During both training and inference, the noisy geometry feature at invalid pixels is replaced by fresh Gaussian noise before patchification:

$$\mathbf{x}_t^{\text{net}}(\ell, \mathbf{u}) \;=\; A(\mathbf{u}) \cdot \mathbf{x}_t(\ell, \mathbf{u}) \;+\; (1 - A(\mathbf{u})) \cdot \boldsymbol{\epsilon}(\ell, \mathbf{u}), \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{7}$$

This invalid-pixel corruption tells the network to ignore geometry tokens at invalid pixels, while the dense forward fill (Eq. 2) defines what the network should predict along valid input rays. Together they create a dense XYZ prediction problem with the layer-0 alpha as the only visibility input and no auxiliary per-layer mask head: valid input rays always have a target for every layer, and invalid pixels carry only uninformative noise and do not contribute to the endpoint loss.

Monotonicity penalty. We add a soft adjacent-layer monotonicity penalty in the normalized coordinate space. Because the object/dynamic z-score and scene log-median depth transform are monotone in depth, this penalty preserves the same front-to-back ordering as camera-space depth and is active only when adjacent layers violate that order:

$$\mathcal{L}_{\text{mono}} \;=\; \frac{1}{L-1} \sum_{\ell=0}^{L-2} \mathbb{E}_{\mathbf{u}}\big[\, \operatorname{relu}\big(z_\ell(\mathbf{u}) - z_{\ell+1}(\mathbf{u})\big)^2 \big], \tag{8}$$

where $z_\ell(\mathbf{u})$ is the normalized depth coordinate of the predicted layer-ℓ point at pixel $\mathbf{u}$. The full training loss is $\mathcal{L} = \mathcal{L}_{\text{FM}} + \lambda_{\text{mono}}\mathcal{L}_{\text{mono}}$ with λ_{mono}=0.1. The penalty is one-sided (zero when ordering is correct) and acts as a gentle structural prior; we found it stabilizes early training without measurably affecting final geometry quality.

B Training Details

This appendix expands the *Training details* paragraph in Sec. 4.1 with the full hyperparameter list and recipe.

Backbone and resolution. All three models are trained at 504×504 resolution with patch size 14 and L=6 layers. The frozen MoGe ViT-L [74] encoder supplies image tokens: we aggregate features from its last four blocks and project them to the decoder dimension, and only this projection is updated; the rest of the encoder is never trained. The object backbone uses 48 pre-norm DiT blocks [13] at width D=1536 with 24 attention heads, interleaved as [layer-wise, ray-wise, global]×16. At every $(\ell, \mathbf{u})$ token the layer-ℓ noisy geometry is concatenated channel-wise with the (repeated) image feature at pixel $\mathbf{u}$ and projected to D, producing a (B, L, P, D) token grid where P is the number of 14×14 patches. The decoder cycles through layer-wise, ray-wise, and global attention shapes (Sec. 3.2) followed by a linear projection from each decoder token to the XYZ x_0 of its patch and unpatchification to the full multilayer image grid. The model contains about 1.7B parameters in total, of which about 1.4B are trainable after freezing MoGe.

Temporal attention block. For WT-D, a clip of T=16 frames is flattened along $(T \cdot L)$, and one temporal-attention block is inserted after each global-attention block. Each temporal block reshapes tokens to $(B{\cdot}L{\cdot}P,\ T,\ D)$, applies a single self-attention layer along T with 1D RoPE on the time axis, and reuses the host decoder's time AdaLN. The block is initialized with LayerScale [70] γ=10^{-5} on its residual, so warm-starting WT-D from a single-frame WT-O checkpoint reproduces the static behavior on T=1 inputs up to a 10^{-5}-scaled residual and only gradually picks up temporal coupling during fine-tuning.

Training schedule. We train the object and scene models with the same two-stage schedule: a 196×196 low-resolution stage for 100K iterations for faster representation learning, followed by 100K iterations at 504×504. The dynamic model is initialized from the static object checkpoint, augmented with temporal attention blocks, and fine-tuned on clips for another 50K iterations.

Optimizer. AdamW with learning rate 10^{-4}, minimum learning rate 10^{-5} under cosine decay, 2000 warmup iterations, weight decay 0.01, and betas $(0.9, 0.999)$. The main object run uses 64 H100 GPUs with per-GPU batch size 2 and gradient accumulation 4, giving an effective batch size of 512. Gradients are clipped to norm 1.0. EMA weights use decay 0.9995 after warmup. Rare loss spikes above $4\times$ the running EMA are skipped to avoid corrupting Adam statistics.

Inference. We solve the flow ODE with 20 denoising steps. For object and dynamic models the predicted normalized tensor is mapped back to camera-space XYZ via the inverse z-score; for scene models, via the inverse log-median map (App. A).

Robust silhouettes. Real input masks are imperfect: they may shift boundaries, miss thin parts, or leave small holes inside the visible support. During training we jitter the alpha boundary and treat newly exposed or missing-support pixels as pseudo-labeled regions. These pixels inherit the filled geometry target and are supervised at reduced weight, rather than being treated as fully reliable surfaces. This improves boundary and mask-error robustness without introducing a separate silhouette loss, and explains why the same checkpoint remains stable in the TRELLIS hybrid and scene-editing pipelines, which depend on externally produced masks.

Self-consistent intrinsics. For images with unknown K, we recover a self-consistent pinhole intrinsics matrix directly from the predicted layer-0 pointmap by least-squares fitting the projection equations, following MoGe [74]. Concretely, with predicted camera-space points $\mathbf{x}_0(\mathbf{u}) = (X, Y, Z)$ at image pixels $\mathbf{u} = (u, v)$, we solve for (f_x, f_y, c_x, c_y) by minimizing

$$\sum_{\mathbf{u}} \Big\| (u, v) - \big(f_x X/Z + c_x,\ f_y Y/Z + c_y\big) \Big\|^2$$

over valid layer-0 pixels. The principal point is not assumed to be centered. The recovered intrinsics are self-consistent with the predicted pointmap and input pixel grid, which is essential for camera-

aware downstream uses such as object insertion and novel-view video synthesis; App. G.6 evaluates their accuracy.

Layer-aware diffusion-time schedule. The training noise curriculum summarized in Sec. 3.3 uses two complementary timestep distributions over $t \in [0, 1]$. The standard *logit-normal* schedule [13] concentrates samples near t=0.5 and is well suited to deeper layers, where the prediction problem is closer to image-conditioned 3D generation. The *plateaued logit-normal* variant inspired by representation-comparison studies [3] follows the logit-normal density up to its mode and stays constant beyond it, allocating more probability to high-noise (large-t) steps; empirically this favors layer 0 (Table 4), where the prediction is closer to faithful visible-surface reconstruction. Training proceeds in two phases: (i) early in optimization, diffusion times are sampled *independently* per layer (visible layer from the plateaued logit-normal, deeper layers from the standard logit-normal), so each layer is exposed primarily to the noise regime that matches its uncertainty profile; (ii) once all layers have stabilized, the full stack switches to a *shared* timestep drawn from an equal-weight mixture of the two distributions, which keeps coupling across layers consistent at inference. Figure 5 plots the analytic densities of the two base schedules and their 50/50 mixture used in phase (ii).

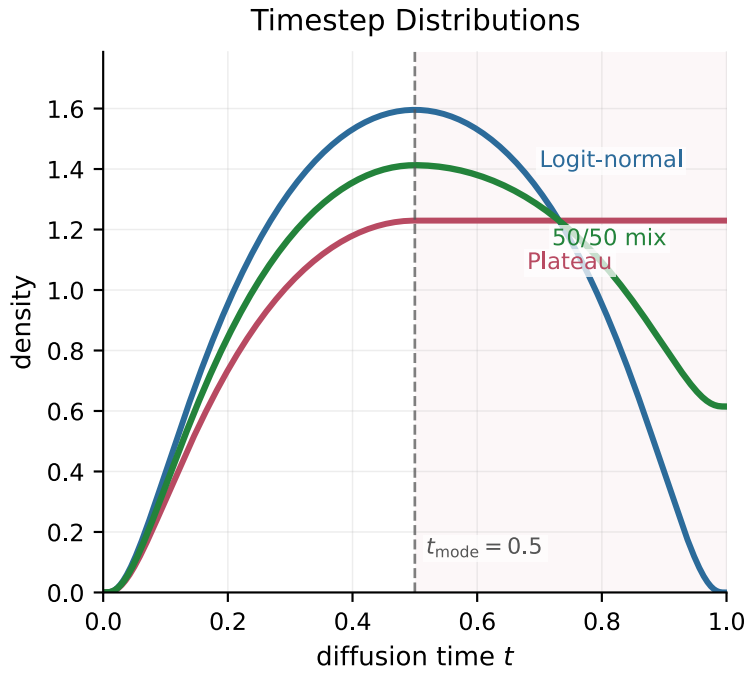

Figure 5: **Training timestep distributions.** Analytic densities of the standard logit-normal schedule, the plateaued logit-normal variant, and their equal-weight 50/50 mixture used in phase (ii) of the layer-aware curriculum.

C Data Pipeline Details

This appendix expands Sec. 3.4 with the full data-pipeline specification.

Rendered inputs and source datasets. We render RGBA images with randomized lighting, viewpoints, and intrinsics. *Object data* covers roughly 300K unique assets and 17M rendered views drawn from Objaverse-XL [9], Objaverse [10], 3D-FUTURE [16], Toys4k [65], GSO [11], and rigged characters from Truebones Motions Animation Studios [71] rendered as static views. *Scene data* includes 3D-FRONT [15] indoor rooms and an internal curated scene corpus of indoor and outdoor scenes with varied intrinsics; the two sources contribute scene training frames in roughly equal proportion. A held-out split of the internal corpus, from which no training frame is drawn, forms the 200-sample internal test set. *Dynamic data* covers roughly 16.8K animated assets sampled as short clips, used to train and evaluate WT-D, combining held-in subsets of Objaverse-XL animated assets with rigged characters from Truebones [71]; held-out splits of the same sources form the Obj.-Val and Truebone evaluation benchmarks.

RGBD-style mix-training corpus. The mix-training paradigm of Sec. 3.5 additionally exposes WT-S to a 12-dataset RGBD-style corpus that provides only the visible-surface layer L_0. Following Eq. 4, every frame in this corpus carries $b_{\text{single}}=1$, so the supervision contributes to L_0 only and the network's predictions for $L_1, \ldots, L_5$ remain shaped exclusively by the multilayer 3D-asset corpus. The constituent datasets group into two families that match what dedicated monocular-depth predictors consume:

- *Real RGBD photographs.* ScanNet v2 [7] ($\sim$2.5M structured-light RGB-D frames from $1{,}513$ indoor scans); MegaDepth [33] (Internet photo collections with MVS-derived depth on $\sim$150K landmark images); BlendedMVS [92] ($\sim$17K frames with photometrically blended MVS depth across diverse object- and scene-scale captures); ARKitScenes [1] ($\sim$5K rooms with Apple ARKit ToF depth and laser-scanner ground truth); Argoverse2 [78] (LiDAR-derived sparse depth on $1{,}000$ outdoor driving scenes); and Waymo Open [66] (LiDAR-derived sparse depth on $\sim$1,150 outdoor driving segments).

- *Synthetic single-view RGBD.* Hypersim [55] ($\sim$77K photorealistic indoor renders with dense depth); Taskonomy [94] ($\sim$4.6M indoor frames with dense depth from $\sim$500 buildings); and several smaller synthetic single-view datasets (including Aria Synthetic Environments, Parallel-Domain, and TartanAir v2) used as additional indoor and outdoor diversity.

For each mini-batch we draw from the multilayer 3D-asset corpus with probability $p_{\text{asset}}=0.6$ and from the RGBD-style corpus with $p_{\text{rgbd}}=0.4$. Within the RGBD branch, datasets are weighted by $\sqrt{N_{\text{rows}}}$ with a $1.5\times$ boost on the six real-photograph sets. Frames are loaded at the same 504×504 resolution as the rendered corpus and pass through the same online augmentation (random crop, resize, horizontal flip, photometric jitter, no per-frame rotation for sparse-depth driving frames). Depth values are converted to camera-space XYZ with the original intrinsics, normalized by the same per-sample log-median map used for scene data (App. A), and broadcast across $L=6$ layers; the $b_{\text{single}}=1$ flag then masks all but L_0 from the endpoint loss. No new model parameters, head, or regime embedding are added; only the data-side mask gate of Eq. 4 differs between the two regimes.

Evaluation splits and protocol. For object evaluation, we randomly sample 100 unique assets from the held-out object corpus and evaluate all rendered views for those assets. For scene evaluation, we use a held-out 3D-FRONT split and the 200-sample internal test set described above, which probes generalization beyond furnished rooms but is in-distribution in source. For dynamic-object evaluation, Obj.-Val denotes held-out Objaverse-XL animated-object assets, Truebone denotes a held-out split of rigged animated assets, and ActionBench, an external benchmark not drawn from our training sources (out-of-domain), evaluates action-driven articulated motion. Baselines cover dedicated depth predictors, layered predictors (LaRI), image-to-3D generators (TRELLIS.2, SAM 3D, LaS-Comp, ReconViaGen), dynamic-geometry methods (GVFDiffusion, SS4D, ActionMesh), and TRELLIS hybrids. For stochastic diffusion or generative methods, including TRELLIS-style baselines, our hybrids, and WT, we draw $K=8$ random seeds per input and report the best geometry result; App. G.2 reports the non-oracle mean$\pm$std for WT.

Metrics. We emphasize geometry faithfulness over aesthetic plausibility, intentionally separating these measurements from mesh or video scores that often reward visual plausibility rather than agreement with the input geometry. For visible surfaces we report standard depth errors (MAE, RMSE, AbsRel, $\delta<1.25^k$). For complete geometry we report Chamfer distance (L1, L2) and F-score to ground-truth meshes or point samples, with visible (L0) and occluded (All-L) breakdowns where possible; the registration and point-sampling protocol is given in App. G.1. For dynamic clips we average per-frame geometry errors.

Multilayer geometry by depth peeling. Ground-truth layers are generated with depth peeling [14, 29, 47]. We render the scene repeatedly from the same camera: the first pass records the nearest visible surface, and each following pass ignores surfaces already captured at smaller depth so that the next intersection along the ray becomes visible. This produces an ordered stack of depth maps, where layer ℓ stores the ℓ-th front-to-back surface hit for each pixel when such a hit exists. We retain the first L surfaces per ray and unproject every valid depth value with the render intrinsics to obtain camera-space XYZ supervision. Rays with fewer than L intersections are handled by the target filling rule in Sec. 3.1, giving the network a dense multilayer tensor while preserving the true set of observed intersections.

Augmentation. We apply online augmentations to narrow the gap between renderings and photos and expand our dataset. Geometric augmentations include random crop, resize, horizontal flip, in-plane rotation, and small affine perturbations such as image-plane translation, anisotropic scale/aspect-ratio jitter, and mild shear, with the camera-space targets transformed consistently with the image operation. Photometric augmentations include brightness, contrast, saturation, hue, gamma, exposure, and white-balance jitter, together with occasional blur, sharpening, compression artifacts, and sensor noise. For alpha masks, we randomly dilate or erode the silhouette, perturb the boundary, drop thin structures, and fill or remove small holes; the affected pixels are handled with the pseudo-labeling strategy described above. For dynamic clips, all geometric and mask-space augmentations are shared across frames to preserve temporal consistency, while photometric noise can vary mildly over time to mimic real video capture.

D Layer Validity Statistics

Table 5 reports the per-layer fraction of valid pixels in our object renderings, supporting the discussion in Sec. 4.3 (Depth filling vs. mask prediction) and App. H.1 (How many layers?). Coverage is measured over the full 504×504 rendered image, including background outside the object silhouette. Mean valid coverage drops from 8.14% at L0 to 0.60% at L5, a more than $13\times$ imbalance that explains why a per-layer mask head collapses on deeper layers. Six layers cover the overwhelming majority of valid rays while keeping the multilayer tensor compact.

Table 5: **Valid multilayer pixels in object renderings.** Per-layer pixel-coverage statistics aggregated over the object validation corpus, measured over the full rendered image including background. Deeper intersections are much sparser, which makes per-layer mask prediction severely imbalanced.

Layer	Mean (%)	Median (%)	P25 (%)	P75 (%)	P90 (%)	Valid views
L0	8.14	6.21	3.92	10.56	16.19	964,508
L1	7.90	6.01	3.80	10.25	15.82	963,163
L2	2.53	1.46	0.61	3.02	5.97	910,810
L3	1.74	0.86	0.29	1.93	4.20	835,007
L4	0.72	0.21	0.05	0.68	1.78	696,663
L5	0.60	0.15	0.04	0.50	1.44	516,342

E Application Pipelines

This appendix expands Sec. 5 with the full description of each downstream pipeline. All three pipelines reuse the same WT prediction (camera-space multilayer point stack with recoverable intrinsics) and add only lightweight, training-free composition logic; the predictor itself is the 1.7B-parameter model whose cost is reported in App. G.7.

E.1 Text-Driven 3D Scene Editing

Given a photograph and a natural-language edit, a 2D editor can produce a plausible edited image, but not a 3D-consistent scene. WT closes this gap. As shown by the object-insertion example in the final panel of Fig. 1, we first rely on the 2D editor's ability to add, remove, or replace image content, then lift the edited result into 3D while preserving image-grid correspondence. We predict the original scene with WT-S, predict the generated object or edited region with WT-O, and insert it into the 3D scene in closed form because both outputs already live in the photograph's camera frame and share pixel/point correspondences. The resulting agent supports object insertion, removal, replacement, and compositional edits from a single image, without per-edit optimization or a rendering loop. The long-form demo video provides many additional examples. This is precisely where pixel alignment matters: a canonical-frame object generator may produce a plausible asset, but it does not know where that asset lies in the input camera.

E.2 Geometry-Guided Novel-View Video Synthesis

Figure 4 illustrates this view-synthesis path. Image-to-video world models often use a depth map or rendered geometry track as conditioning [4, 24, 36, 93]. A single visible layer, however, is incomplete by construction: once the camera moves, newly exposed regions must be invented from scratch. WT provides a complete geometry memory before video generation begins. Because WT's multi-layer geometry is pixel-aligned with the input image by construction, it integrates with these video models: we rasterize the predicted point cloud along a target trajectory to generate dense, multi-view depth guidance. For object orbits, the back side is already present, allowing the frozen video model to edit and fill a geometry-consistent signal rather than hallucinating disoccluded structure without support.

E.3 Pose-Aligned TRELLIS Hybrid

Figure 4 also shows the pose-aligned mesh path. TRELLIS-style pipelines produce visually polished textured meshes, but their sparse structure is generated in a canonical frame and has no explicit input camera pose [80]. The mesh may look plausible while failing to reproject to the image that conditioned it. Since WT already predicts a camera-space, pixel-aligned point stack with recoverable intrinsics, we voxelize this stack and use it as the sparse structure for the later TRELLIS stages, without retraining TRELLIS. Depth-filled duplicate points are benign here: Stage 1 consumes a voxelized occupancy grid, where duplicates collapse into one cell. The hybrid keeps TRELLIS' texture and mesh decoder while replacing its weakest link: the pose-agnostic Stage-1 structure. We evaluate against pure TRELLIS.2 and VGGT-guided hybrids such as ReconViaGen [5] and LaS-Comp [86].

F Geometry Generation: Qualitative Analysis and Real-Scene Depth

This appendix expands the qualitative discussion deferred from Sec. 4.2 (*Objects* and *Scenes*) into the underlying mechanisms, and reports real-scene depth benchmarks.

F.1 Why WT-O outperforms regression-based and canonical-frame baselines

Two complementary mechanisms drive the object gains in Table 1 and Fig. 3 (top).

Diffusion vs. regression on occluded surfaces. Compared with LaRI, which uses a regression-style layered head, our diffusion formulation is better suited to fully invisible back-side geometry: regression tends to average plausible completions, producing smooth or detail-poor backs, whereas diffusion can model the multi-modal uncertainty of occluded surfaces. LaRI also jointly predicts the layered points and a ray-stopping index that marks the valid layers; because deeper-layer supervision is extremely imbalanced (App. D), this validity prediction often stops after the first few layers, effectively producing only shallow geometry. Our depth-filling objective avoids this failure mode by making all foreground rays dense without requiring a separate validity prediction.

Pixel-aligned representation vs. canonical-frame generation. Compared with TRELLIS-style generators trained primarily from 3D assets, WT-O generalizes better to real images because its pixel-aligned representation can inherit 2D image priors through the frozen MoGe encoder while still predicting complete geometry. Canonical-frame generators sometimes produce visually polished meshes whose geometry is unfaithful to the input: a plausible-looking mesh can flip front/back leg relationships, while pixel-aligned methods such as LaRI and WT-O preserve the input pose ordering. TRELLIS.2 also occasionally introduces spurious planar structures. These failures are consistent with Table 1: canonical generators can look attractive, but they are weaker at faithful generation when evaluated against the input geometry and ground truth.

Architecture vs. data scale. The data-matched controls in App. G.3 isolate this question quantitatively: LaRI retrained on our corpus for the same number of iterations improves over its release but still trails WT on both the visible layer and full geometry, so the gains are not solely a consequence of scaling the dataset.

F.2 Why WT-S preserves planar structure

LaRI-scene is visibly less complete and less stable than WT-S, again reflecting the difficulty of regressing sparse deeper layers and their validity. More surprisingly, even without MoGe-2-scale real-scene training and mostly on virtual scene data, WT-S can produce more faithful geometry on some generated indoor/outdoor images that are outside its training set. In the examples of Fig. 3 (bottom), MoGe-2 sometimes bends structures that should be planar or vertical, such as facades and walls, while WT-S preserves straighter, more coherent scene layout. Figure 6 shows the full $L{=}6$ layer stack: deeper layers progressively populate occluded geometry (e.g., floor and back-wall regions behind furniture) while the visible layer 0 remains faithful to the input.

F.3 Real-scene depth benchmarks

We evaluate WT-S on two widely used real-scene depth benchmarks released alongside dedicated monocular-depth methods: NYU Depth V2 [63] (indoor only, $1{,}449$ frames) and ETH3D [59] (split into 7 indoor and 6 outdoor scenes, 454 frames). Table 6 reports per-sample SSI-aligned visible-surface depth (AbsRel, RMSE, $\delta{<}1.25$) under the same alignment protocol used in Table 2, at evaluation resolution 504. Both benchmarks are out-of-domain for WT-S. We report two WT-S checkpoints to isolate the mix-training contribution of Sec. 3.5: the *3D-asset-only* checkpoint used in every other table, trained exclusively on the multilayer corpus of App. C, and the *mix-trained* checkpoint, warm-continued from it for 50K iterations with the additional RGBD-style corpus supervised on L_0 only (Eq. 4).

Setup disparity. The 3D-asset-only scene corpus consists almost entirely of two sources, in roughly equal proportion: the public 3D-FRONT split [15] and the internal curated scene corpus. Both are chosen because they admit ground-truth multilayer geometry through depth peeling, not because of their visible-surface scale, and outdoor scenes are under-represented by construction. The rendered-frame budget is several orders of magnitude smaller than the real RGB-D corpora that MoGe-2 [75], Pi3X [77], Depth Anything 3 [39], and VGGT [73] train on. Despite this disparity, the 3D-asset-only WT-S is already comparable to MoGe-2 on the indoor regime and remains competitive outdoors; the residual outdoor gap is concentrated on the `playground` scene, which is far out of distribution for every method evaluated.

Mix-training narrows the visible-surface gap. Adding the RGBD-style corpus reduces AbsRel under the default 20-step inference on all three splits (NYU $0.0398 \rightarrow 0.0382$, ETH3D-indoor $0.0398 \rightarrow 0.0345$, ETH3D-outdoor $0.0533 \rightarrow 0.0495$). With 50 denoising steps, the best-of-8 AbsRel is the best on ETH3D-indoor and second only to Pi3X on NYU and ETH3D-outdoor. Under the non-oracle mean over the same 8 seeds of this 50-step configuration, the mix-trained model is ahead of MoGe-2 on AbsRel on both ETH3D splits and on $\delta{<}1.25$ on NYU and ETH3D-outdoor, within 0.0004 of it on NYU AbsRel (0.0392 vs. 0.0388), and comparable to Pi3X overall. These numbers are taken after only 50K mix-training iterations while the validation loss is still decreasing, so they are a lower bound for this recipe. Mix-training only masks the deeper-layer loss for single-layer samples and never modifies the multilayer supervision, so the multilayer capability is preserved.

Position of this work. Our central goal is not to surpass dedicated single-layer depth predictors on visible surfaces, but to additionally produce occluded geometry that none of these baselines can predict, while remaining competitive on the visible surface itself. All other tables use the 3D-asset-only checkpoints; only Table 6 reports the mix-trained WT-S, as an internal ablation of the RGBD mix.

G Additional Experiments

This appendix collects additional experiments on evaluation protocol, sampling variance, design choices, out-of-domain generalization, intrinsics recovery, and cost. Unless stated otherwise, controlled experiments use the same evaluation pipeline as the main paper.

G.1 Full-geometry registration and point sampling

All entries of the full-geometry part of Table 1, WT-O included, are registered to the ground truth with the identical per-sample Sim(3) alignment (Umeyama initialization [72] followed by 10 ICP

Table 6: **Real-scene visible-surface depth.** Sample-mean L0 depth metrics on NYU Depth V2 [63] (1,449 indoor frames) and ETH3D [59] split into its 7 indoor and 6 outdoor scenes. All methods use the same SSI alignment at evaluation resolution 504. Pi3X is the upgraded π^3 model; DA3 denotes Depth Anything 3. WT-S rows are best-of-8 random seeds (selection by AbsRel) unless marked as mean-of-8 (non-oracle; RMSE not measured; excluded from the ranking colors). The *3D-asset-only* checkpoint is used in every other table; the *3D-asset + RGBD* checkpoint additionally consumes the RGBD-style corpus of App. C on L_0 only. Rows marked $*$ use 50 denoising steps instead of the default 20. Best Second best Third best.

	NYU (indoor, 1449 frames)			ETH3D indoor (222 frames)			ETH3D outdoor (232 frames)		
Method	AbsRel↓	RMSE↓	δ<1.25 ↑	AbsRel↓	RMSE↓	δ<1.25 ↑	AbsRel↓	RMSE↓	δ<1.25 ↑
DA3 [39]	0.0458	0.1453	0.9676	0.0575	0.1831	0.9666	0.0652	0.2381	0.9366
LaRI-scene [31]	0.0480	0.1424	0.9708	0.0523	0.1503	0.9742	0.0618	0.1964	0.9549
VGGT [73]	0.0415	0.1403	0.9731	0.0417	0.1380	0.9748	0.0562	0.1789	0.9557
MoGe-2 [75]	0.0388	0.1270	0.9758	0.0378	**0.1265**	**0.9878**	0.0501	0.1849	0.9674
Pi3X [77]	**0.0341**	**0.1190**	0.9827	0.0378	0.1269	0.9852	**0.0434**	**0.1674**	**0.9760**
WT-S (3D-asset only)	0.0398	0.1384	0.9735	0.0398	0.1470	0.9804	0.0533	0.2104	0.9537
WT-S (3D-asset + RGBD)	0.0382	0.1346	0.9750	0.0345	0.1306	0.9863	0.0495	0.1822	0.9633
WT-S (3D-asset + RGBD*)	0.0374	0.1312	**0.9871**	**0.0332**	0.1297	0.9849	0.0451	0.1731	0.9726
WT-S (3D-asset + RGBD*), mean-of-8	0.0392	–	0.9863	0.0350	–	0.9841	0.0469	–	0.9718

iterations [2]) before computing Chamfer distance and F-score. Because the Sim(3) fit removes global pose and scale, no canonical-frame method is penalized for its output convention, and WT-O's registration-free camera-frame output earns no credit in this table: the margin reflects completion quality, not alignment. All methods are compared against the same ground-truth set at a matched budget of 50K points. Mesh baselines (TRELLIS.2, LaS-Comp, ReconViaGen, WT-O*) are randomly downsampled from their dense vertex sets ($\gg$ 50K vertices), SAM 3D uses its Gaussian centers, and WT-O's native points and the ground-truth set are subsampled to the same 50K. Across 3 independent subsamplings, CD-L1 varies by at most ± 0.0003, and sampling the mesh baselines uniformly on the surface instead of from their vertices changes CD-L1 by less than 0.001, far below the margins we draw conclusions from.

G.2 Non-oracle results

The main tables report the oracle best of $K=8$ random seeds for every stochastic method. Table 7 reports the non-oracle mean$\pm$std over the same 8 seeds for WT. Every object and scene conclusion is unchanged: WT-O's non-oracle mean AbsRel (0.0089) is still better than the best deterministic baseline (VGGT, 0.0138), and its non-oracle mean CD-L1 (0.0243) is still better than the oracle best-of-8 of every stochastic image-to-3D baseline (e.g., SAM 3D, 0.0475). Since those baselines are reported with the oracle protocol that favors them, the ordering does not depend on seed selection for either side.

Table 7: **Oracle best-of-8 vs. non-oracle mean$\pm$std over the same 8 seeds** for WT. T1 and T2 denote Tables 1 and 2.

Benchmark, metric	Best-of-8	Mean$\pm$std
T1 visible surface, AbsRel↓	0.0079	0.0089$\pm$0.0002
T1 full geometry, L1↓	0.0213	0.0243$\pm$0.0012
T1 full geometry, L2↓	0.00194	0.00221$\pm$0.00012
T1 full geometry, F@0.01↑	0.549	0.497$\pm$0.015
T1 full geometry, F@0.05↑	0.898	0.876$\pm$0.006
T2 3D-FRONT, AbsRel↓	0.0114	0.0132$\pm$0.0003
T2 3D-FRONT, All-L CD-L1↓	0.0216	0.0236$\pm$0.0003
T2 internal, All-L CD-L1↓	0.0750	0.0791$\pm$0.0003

G.3 Controlled ablations and data-matched comparison with LaRI

Retraining every variant on the full 17M-view corpus is infeasible, so we train a ladder of controls on a shared 500K-image object subset with identical optimizer, schedule, number of iterations, res-

olution, and evaluation (100 held-out assets). Each variant changes exactly one factor relative to the full model B. Table 8 reports best-of-8 with the non-oracle mean-of-8 in parentheses.

Table 8: **Controlled single-factor ablations** on a shared 500K-image object subset (best-of-8; mean-of-8 in parentheses). LaRI rows are deterministic. D predicts x_0 from the noise input in a single step without the flow-matching ODE.

ID	Variant (single change vs. B)	L0 AbsRel↓	All-L CD-L1↓
B	full model (L=6)	0.0119 (0.0127)	0.0276 (0.0287)
A	single layer (L=1)	0.0123 (0.0131)	–
C	w/o MoGe conditioning	0.0141 (0.0148)	0.0321 (0.0328)
D	single-step regression instead of flow matching	0.0121 (0.0122)	0.0368 (0.0370)
F	additive layer embedding instead of layer FiLM	0.0123 (0.0130)	0.0283 (0.0291)
G	ray-wise attention replaced by layer-wise	0.0117 (0.0125)	0.0280 (0.0291)
E	LaRI [31], same data and iterations	0.0167	0.0407
–	LaRI [31], released checkpoint	0.0198	0.0491

Multilayer prediction (A vs. B). With everything else fixed, predicting six layers modestly improves layer 0 ($0.0123 \to 0.0119$ best-of-8; $0.0131 \to 0.0127$ mean). We do not claim a large representational gain on the visible surface: occluded-layer generation costs no visible-surface fidelity and, if anything, slightly helps. The visible-surface margin in Table 1 therefore reflects representation, architecture, conditioning, and data combined.

Image prior (C). Removing MoGe conditioning slows convergence and degrades both metrics. All other variants keep it, so it does not confound the remaining comparisons.

Generative objective (D). Single-step regression matches B on layer 0 but is 33% worse on full geometry (0.0368 vs. 0.0276), with mode-averaged, detail-poor backs: the generative objective matters where occluded geometry is multi-modal, not for the visible surface.

Layer conditioning (F). Layer FiLM outperforms a capacity-matched additive layer embedding on both metrics.

Ray-wise attention (G). Replacing ray-wise with layer-wise attention (identical parameter count; only the grouping changes) leaves the decoder alternating within-layer and global attention as VGGT-style backbones do, so cross-layer coupling passes only through the 16 global blocks. G reaches parity on layer 0 (0.0117 vs. 0.0119) and is slightly worse on full geometry, while costing about 216× more attention compute in the 16 replaced blocks: with P=1296 and L=6, layer-wise attention runs 6 sequences of length 1296 (≈61.9 GFLOPs per block), whereas ray-wise attention runs 1296 sequences of length 6 (≈0.29 GFLOPs). The ray axis thus delivers cross-layer coherence at almost no cost rather than being the only way to obtain it.

Data vs. method (E). LaRI retrained on our corpus for the same number of iterations improves over its release ($0.0198/0.0491 \to 0.0167/0.0407$), so data curation helps, but it still trails WT substantially. Comparing the LaRI rows isolates data curation, D vs. B isolates the objective, and E vs. D isolates architecture and depth filling: no single factor explains the gap.

G.4 Filled vs. genuine deep layers

Why not mask prediction. A direct multilayer formulation couples XYZ regression with classifying whether each layer exists. In early joint depth+mask runs, the EMA cosine similarity between the two gradients stays near zero or negative (e.g., -0.19 at 500 iterations), strongest in the early decoder blocks. Mask supervision is also severely imbalanced (mean valid area 8.14% at L0 vs. 0.60% at L5; App. D), which makes the trivial "invalid" prediction attractive and leads mask heads to suppress deeper surfaces. Forward filling removes the classification task and turns supervision into dense XYZ regression.

Filled entries dominate deep-layer supervision (about 92% of in-silhouette L5 entries). We test whether this impairs genuine deep-layer prediction, whether termination can be read out without a mask head, and whether filled points affect the All-L metrics.

Genuine deep-layer accuracy. Table 9 splits per-layer AbsRel on the 100 held-out object assets (best-of-8, as in Table 1) by pixel type. Genuine deep-surface error is flat across layers and comparable to layer 0 (0.0079), and the two populations stay in the same error regime at every layer. If copied targets dominated learning at the expense of real geometry, genuine-pixel error would degrade sharply with layer index; it does not.

Table 9: **Per-layer AbsRel of WT-O split by pixel type** (genuine intersections vs. forward-filled entries; best-of-8).

AbsRel↓	L1	L2	L3	L4	L5
Genuine	0.0089	0.0090	0.0099	0.0094	0.0095
Filled	0.0101	0.0097	0.0098	0.0098	0.0098

Termination is decodable from the output. Although WT has no explicit termination channel, thresholding the predicted inter-layer gap Δz recovers it reliably: the median gap is about $70\times$ larger for genuine new surfaces than for collapsed layers (0.028 vs. 0.0004 in normalized depth), giving ROC-AUC 0.99 at L1 (F1 0.99 with one global threshold) and 0.97 pooled over L1–L5. When a back surface truly coincides with the front surface, a collapsed layer and the true geometry contribute the same repeated points, so the residual ambiguity does not change the predicted shape.

Filled points in All-L Chamfer. Predicted outputs are continuous, so filled entries are never exactly equal to the surface point they repeat, and we apply no separation when computing All-L metrics. Table 10 (top) stress-tests this with voxel deduplication, which is defined purely in 3D output space and also merges coincidences across rays (object benchmark, 50 samples, paired differences, best-of-8). Even removing 69% of the points changes CD-L1 by at most 0.0004: repeated points on a correct surface leave the ground-truth-to-prediction term invariant and only re-weight already-correct points in the prediction-to-ground-truth term. When a redundancy-free cloud is wanted, the ray structure provides a cheaper filter: dropping a layer point whose gap to the previous kept layer is below τ (Table 10, bottom) removes a quarter to a third of the points with unchanged or slightly better metrics, confirming that they are redundant rather than spurious geometry.

Table 10: **Effect of removing repeated points on full-geometry metrics** (WT-O). Top: voxel deduplication, with Δ = after minus before (positive is worse for CD-L1, negative is worse for F). Cell sizes are relative to the object extent. Bottom: ray-gap filter with threshold τ on the full benchmark of Table 1.

Duplicate removal	Points removed	ΔCD-L1	ΔF@0.05
Exact duplicates only	0.1%	+0.00001	0.000
Voxel dedup, cell = 0.2% of extent	23.8%	−0.0002	−0.001
Voxel dedup, cell = 0.5% of extent	69.3%	+0.0004	−0.004
Ray-gap threshold τ	**Points removed**	**CD-L1↓**	**F@0.05↑**
none (Table 1)	–	0.0213	0.898
0.002	27.4%	0.0208	0.899
0.005	31.8%	0.0208	0.899

G.5 Out-of-domain real benchmark with complete ground truth: SCRREAM

Dense occluded ground truth cannot be captured at scale, which is why our headline multilayer benchmarks are synthetic. SCRREAM [26] is a rare exception: real indoor captures with complete ground-truth scene meshes, so occluded geometry can be scored on real data, with no training overlap for either method. Table 11 compares the 3D-asset-only WT-S with LaRI-scene, the only other method that outputs occluded layers, under the same SSI protocol. WT-S is reported as the non-oracle mean over 8 seeds. For objects we know of no real-image benchmark with ground-truth geometry for segmented foreground objects, let alone occluded geometry, so we report qualitative results on DAVIS instead (Fig. 3 and the supplementary video).

Table 11: **Out-of-domain real scenes with complete ground truth (SCRREAM).** WT-S is the 3D-asset-only checkpoint used in Table 2 (non-oracle mean-of-8).

Method	L0 AbsRel↓	All-L CD-L1↓
LaRI-scene [31]	0.0697	0.073
WT-S	**0.0331**	**0.058**

G.6 Intrinsics recovery

Several downstream uses rely on the closed-form intrinsics fit of App. B, which jointly recovers (f_x, f_y, c_x, c_y) from the predicted layer-0 pointmap without assuming a centered principal point.

Protocol. We use 100 held-out 3D-FRONT renders at 840×840 (a multiple of the 14-pixel patch size) with ground-truth render cameras. Native 3D-FRONT cameras are centered, which makes c_x, c_y unidentifiable, so we randomly crop each image to 70–90% of its side length off-center, resize it back, and transform the ground-truth intrinsics analytically. As a reference we use MoGe-2 [75], but rather than its reported intrinsics, which fix the principal point at the image center, we fit its raw pointmap with the identical full-K solver over the same pixels, so both methods are asked the same question about their own geometry. All WT values are non-oracle means over 8 seeds per image.

Focal length and field of view. With the solver fixed, layer-0 geometry from WT-S supports a roughly $3\times$ more accurate focal estimate (Table 12, left), consistent with its more accurate visible surface (Table 2). On objects, whose native cameras are centered so that only focal length and FoV are identifiable, the same solver gives WT-O $12.8\%/18.0\%$ median/mean relative focal error and $5.3°/6.3°$ FoV error, against $17.1\%/22.8\%$ and $7.1°/8.2°$ for MoGe-2.

Principal point. We present this as evidence that WT's geometry retains off-center calibration information, not as a head-to-head comparison: MoGe-2 does not model a free principal point, so off-center crops lie outside its parameterization. WT-S recovers roughly two thirds of the off-center shift and is closer to the true principal point on 91 of 100 images, whereas the free fit on MoGe-2's pointmap lands on the centered reference (Table 12, right).

Table 12: **Intrinsics recovered from layer-0 pointmaps** on off-center crops of 100 held-out 3D-FRONT renders (median / mean). Left: relative focal error and FoV error. Right: principal-point (PP) error in pixels and as a fraction of the image diagonal; the last row fixes the PP at the image center (MoGe-2's official intrinsics).

Method (same full-K solver)	Focal err. (%)↓	FoV err. (°)↓	PP err. (px)↓	PP err. (% diag.)↓
WT-S	**3.64 / 4.83**	**1.97 / 2.59**	**25.55 / 31.47**	**2.15 / 2.65**
MoGe-2 raw pointmap	11.19 / 17.37	5.83 / 8.17	72.85 / 81.49	6.13 / 6.86
Reference: PP fixed at center	–	–	72.74 / 81.44	6.12 / 6.86

G.7 Computational cost

The 48 decoder blocks interleave [layer-wise, ray-wise, global]$\times16$, so only 16 blocks attend over the full $L \cdot P = 6 \times 1296 = 7{,}776$-token sequence at 504×504. All attention calls dispatch to FlashAttention [8] after reshaping, so no attention score matrix is materialized and memory grows linearly with sequence length. The 16 global blocks account for about 5.9 of the roughly 28 TFLOPs of one forward pass ($\approx21\%$). Table 13 reports measured cost on one H100 GPU in BF16 for the 1.7B-parameter model at 504×504 with L=6. Inference fits on a 16 GB GPU.

Table 13: **Measured cost of WT-DiT** (one H100, BF16, 504×504, L=6, 1.7B parameters).

Setting	Peak memory	Time
Inference, batch 1, 20 ODE steps	9.4 GB	4.48 s / image
Training micro-step, batch 2, activation checkpointing	44.6 GB	1.48 s / step

H Additional Discussion

This appendix expands the design choices summarized in Sec. 4.3.

H.1 How many layers?

Six layers are a practical sweet spot rather than an arbitrary constant. As reported in App. D, valid support falls rapidly after the first two surfaces; later layers capture thin structures and repeated occlusions, but their marginal area is small. Increasing L would improve rare highly perforated shapes such as foliage, cages, or grates, but it also increases memory and makes the already sparse tail harder to supervise. This motivates future variable-depth or adaptive-ray representations, while keeping $L{=}6$ as the default.

H.2 XYZ pointmaps versus depth plus intrinsics

We also tested predicting depth with camera intrinsics instead of camera-space XYZ. Although depth+intrinsics is compact and interpretable, it couples every ray through a global calibration prediction: small errors in focal length or principal point coherently warp the whole shape. Direct XYZ prediction absorbs this calibration into the pointmap itself, which leads to more plausible global shape, especially on real-world images whose crop, mask, and camera metadata are uncertain. This matches the motivation of pointmap-based reconstructors: the output is already a 2D-to-3D map, and intrinsics can be recovered from the predicted L0 geometry when needed.

H.3 Camera-ray stacks versus canonical surface parameterizations

Recent generative surface methods instead flatten 3D Gaussians onto a canonical 2D map [53, 82], which keeps the representation stationary when the object rotates. Ray stacks are view-dependent: under a change of viewpoint, a surface point can move to a different layer index. This is a genuine trade-off, and we chose camera rays deliberately for three reasons. First, pixel alignment: every point WT predicts is tied to an input pixel, which enables pixel-level 3D editing and intrinsics recovery, whereas canonical maps provide no such correspondence. Second, generality: an object-centric map is defined for a single complete object in a canonical pose, and has no natural definition for unbounded scenes viewed from within or for multiple disconnected objects, while one ray-stack formulation covers objects, scenes, and dynamic content. Third, data: layer 0 is a pointmap, so the representation directly ingests real depth data and inherits priors from geometry foundation models, whereas a single RGBD frame cannot be converted into a canonical map of a complete object. Because WT is conditional, geometry is always predicted in the conditioning camera's frame, and layer reassignments occur only at tangency events, analogous to depth discontinuities at occlusion boundaries in single-layer pointmaps. Canonical spherical maps face a related issue, since splats along one radial direction are assigned to a fixed number of slots and rotations relative to the canonical frame reshuffle the map.

H.4 Architecture details

Small-scale ablations support the final transformer choices. LayerScale initialization improves early optimization substantially in a controlled small-scale ablation: adding LayerScale with initialization 10^{-4} reduced total loss from 0.009258 to 0.007018 and XYZ loss from 0.007242 to 0.003537 at 10k iterations. RoPE alone was neutral in that small setup, but remains useful in the full model where spatial and ray-wise attention must extrapolate across resolutions and camera crops. These results are not the central contribution, but they justify keeping RoPE and LayerScale in the stable training recipe. Ablations of the attention factorization and layer conditioning are in App. G.3.

H.5 Generalization beyond the training regime

Two empirical behaviors are especially encouraging. First, the object model, even without scene training, produces reasonable geometry on scene images. This suggests that pixel-aligned multilayer prediction transfers across object and scene regimes more naturally than canonical image-to-3D generation, whose output frame and single-object assumptions break on cluttered scenes. Second, the object model is surprisingly robust to multiple objects in one image, despite not being explicitly

trained with multi-object augmentation. Because WT predicts one stack per input ray rather than one canonical asset per crop, multiple disconnected objects can coexist in the same output tensor. These observations point to a clear next step: unify object and scene training into a single stronger model, rather than maintaining separate specialists.

H.6 Limitations and future work

WT is still bounded by a fixed layer count, rendered supervision, and iterative flow sampling. Highly perforated geometry may require more than six surfaces; synthetic-to-real artifacts remain on texture-less or reflective regions; and real-time applications will require distillation or fewer sampling steps. Our headline multilayer benchmarks are in-domain for WT, because dense occluded ground truth is rarely available for real captures; SCRREAM (App. G.5) is the main out-of-domain exception. The current dynamic model also operates on short clips, leaving long-range memory and persistent identities across long videos to future work. A natural extension is to make WT-D predict not only per-frame multilayer geometry, but also explicit pixel/point correspondences and 3D trajectories across time. Another direction is to extend WT-S from single-frame scene lifting to multi-frame input, where multiple observations can be fused into a larger camera-aligned world rather than a single-view local scene. More broadly, the strongest future direction is to train one unified object-scene-dynamic model with stronger real-mask augmentation and adaptive layer allocation.

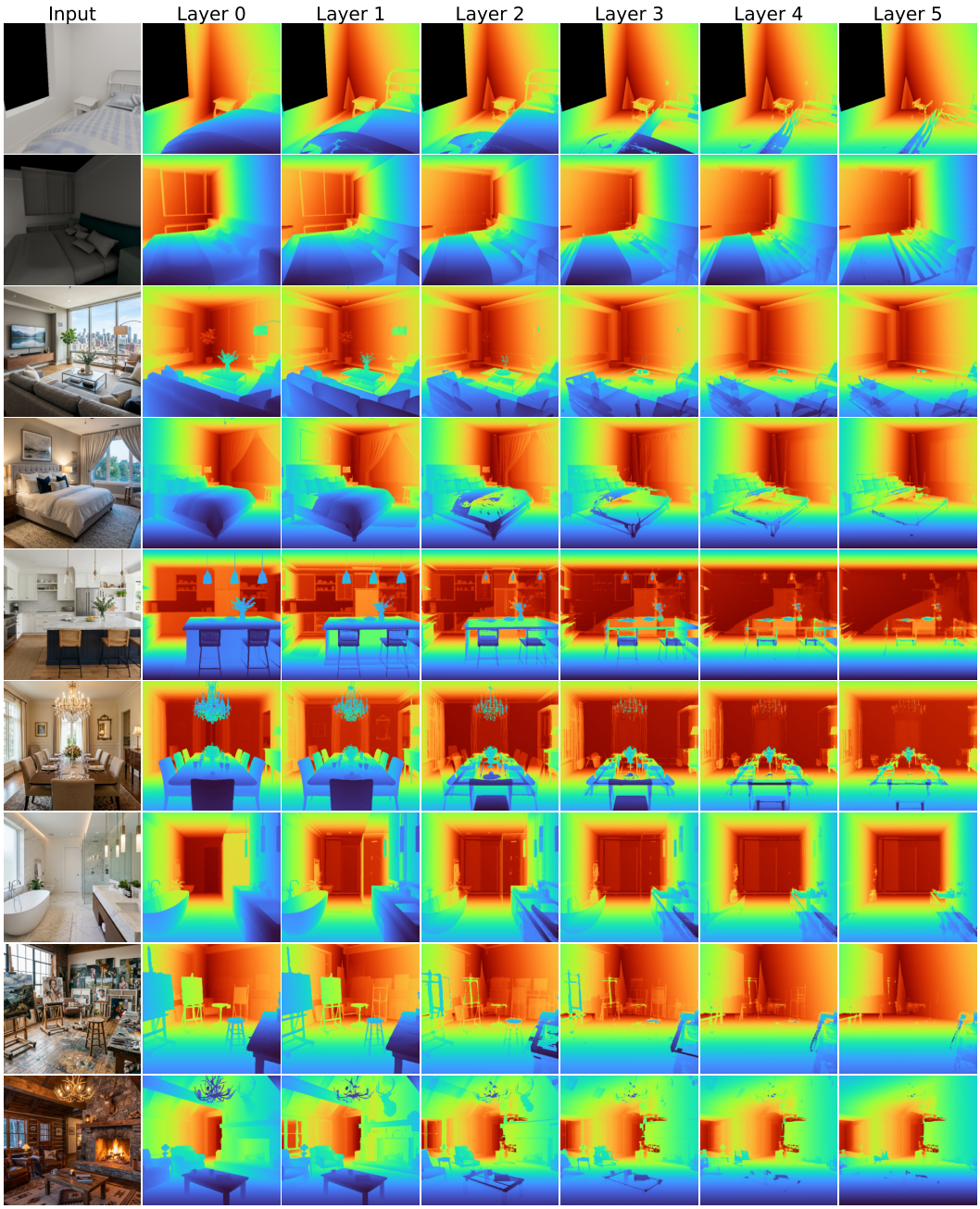

Figure 6: **Multilayer depth stack produced by WT-S.** Each row shows one input (left) followed by the six predicted depth layers in turbo colormap. Top two rows are held-out 3D-FRONT frames; the remaining seven are out-of-distribution generated indoor rooms. As ℓ increases, occluded geometry behind near surfaces is filled in (e.g., floor and walls behind furniture, room interiors behind doorways and chandeliers) while layer 0 stays pixel-aligned with the input. Predictions use the WT-S checkpoint with 20 denoising steps.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Sec. 4 state the two claims (more faithful 3D geometry across regimes, and benefits to downstream pipelines). Each is supported by the corresponding evidence: Tables 1–2 for geometry quality, Sec. 5 and Fig. 4 for downstream pipelines. We further qualify the scope (e.g., visible-surface depth on *real* scenes is reported as comparable, not SOTA, in App. F.3).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are explicitly discussed in App. H.6 (*Limitations and future work*), covering the fixed layer count L=6, reliance on rendered synthetic supervision, iterative flow-matching sampling cost, short-clip dynamic regime, and the under-representation of outdoor scenes in training data (App. F.3). Future directions (adaptive layer allocation, distillation, longer temporal context, unified object-scene-dynamic model) are also outlined.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: This paper introduces a representation, network architecture, and training objective; it does not contain formal theorems or proofs. The flow-matching objective and target-filling rule (Eqs. 3 and 2) are stated explicitly with all definitions.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The architecture is fully described in Sec. 3.2 (with Fig. 2); the loss, scale normalization, and depth-filling rules are written out in Sec. 3.1. Training schedule, optimizer, and decoder details are in App. B; data sources, rendering pipeline, evaluation splits, baselines, and metric definitions are in App. C. All hyperparameters needed to reproduce WT-DiT, WT-O, WT-S, and WT-D are specified.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We release inference code (`https://github.com/haoz19/world-tracing`), object, scene, and dynamic checkpoints on Hugging Face, and an interactive demo, under CC BY-NC-ND 4.0. The released checkpoints are not identical to the evaluated ones (the object and dynamic checkpoints were later updated, and the released scene model is a higher-resolution variant), and training code is not released. The training data combines openly available public collections (Objaverse-XL, Objaverse, 3D-FUTURE, Toys4k, GSO, 3D-FRONT) with an internal scene corpus that is not public (App. C). Public evaluation data (NYU Depth V2, ETH3D, SCRREAM, 3D-FRONT) are independently accessible.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Sec. 4.1 (Experimental Setup) summarizes the data, training setup, evaluation splits, baselines, and metrics; full details (model size of 1.7B parameters, 1.4B trainable, training resolution 504×504, L=6 layers, AdamW optimizer, 64 H100 GPUs, effective batch size 512, 20 flow-ODE inference steps, layer-aware diffusion-time schedule, baseline lists, alignment protocol, and metric definitions) are in App. B and App. C.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: For stochastic methods the main tables report the oracle best of $K{=}8$ sampling seeds, applied identically to WT and to stochastic baselines (Sec. 4.1). App. G.2 reports the non-oracle mean$\pm$std (1-sigma standard deviation of the benchmark-level metric across the 8 sampling seeds) for the key object and scene metrics, Table 1 includes the non-oracle mean row, and the ablations in App. G.3 report best-of-8 and mean-of-8. App. G.1 reports the variation of Chamfer distance across independent point subsamplings. We do not repeat training with multiple seeds because each 1.7B-parameter run requires 64 H100 GPUs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Sec. 4.1 reports that WT-DiT is trained on 64 H100 GPUs at effective batch size 512, and the full training schedule is in App. B. Inference uses 20 flow-ODE steps on a single GPU; App. G.7 reports FLOPs, peak memory (9.4 GB at inference, 44.6 GB per GPU in training), and per-sample and per-iteration wall-clock time. We acknowledge that the full research project, including ablation runs, design exploration, and the layer-aware schedule study, required additional GPU-hours beyond the reported final runs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics. The work uses publicly available 3D asset and RGB-D collections under their original licenses (some, such as Objaverse-XL, were assembled from web sources by their creators) and an internal corpus of synthetic or permissioned 3D scenes; no human subjects are involved and we collect no personal data.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: WT is foundational research on pixel-aligned multilayer 3D geometry. Positive impacts include enabling more faithful 3D scene editing, view synthesis, and asset creation pipelines for content creation, simulation, robotics, and accessibility (e.g., novel-view rendering for low-vision or AR users). Possible negative impacts inherit from those of generic image-to-3D and image-to-video generation: misuse for deceptive media when combined with downstream rendering or video models. Mitigations include releasing models with usage guidelines and not providing identity-preserving capabilities; the underlying geometry output itself is not directly photorealistic.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: WT predicts geometry (camera-space point clouds), not photorealistic imagery, identities, or text. The misuse risk surface is therefore comparable to other monocular geometry models such as MoGe and depth predictors. The released checkpoints come with documentation and a non-commercial, no-derivatives license (CC BY-NC-ND 4.0); no novel safeguards beyond those of the underlying frozen 2D backbone (MoGe) are required.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All public datasets and pretrained models used in this work are properly cited at first use in Sec. 3.1, Sec. 4.1, and App. C: object/scene/dynamic data sources (Objaverse-XL, Objaverse, 3D-FUTURE, Toys4k, GSO, Truebones, 3D-FRONT), the RGB-D mix-training corpora (ScanNet v2, MegaDepth, BlendedMVS, ARKitScenes, Argoverse 2, Waymo Open, Hypersim, Taskonomy, Aria Synthetic Environments, ParallelDomain, TartanAir v2), the frozen image encoder (MoGe ViT-L), the rendering toolchain (`nvdiffrast`), evaluation benchmarks and inputs (NYU Depth V2, ETH3D, SCRREAM, DAVIS, Obj.-Val, Truebone, ActionBench), and all baseline methods compared against (LaRI, MoGe-2, Pi3X, Depth Anything 3, VGGT, TRELLIS/TRELLIS.2, SAM 3D, ReconViaGen, LaS-Comp, GVFDiffusion, SS4D, ActionMesh). Each is used under its respective license; no scraped or licensed-restricted data is repackaged or redistributed by this work.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release inference code and trained object, scene, and dynamic checkpoints, documented in the repository README (installation, checkpoint configurations, usage, and license CC BY-NC-ND 4.0) at `https://github.com/haoz19/world-tracing`. The rendered multilayer training data derived from public 3D collections is not redistributed.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or research with human subjects. All training and evaluation data are synthetic 3D renderings or established public datasets and benchmarks.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This work does not involve human subjects, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method (WT representation, WT-DiT architecture, depth-filling objective, layer-aware diffusion-time schedule) does not use LLMs as an important or non-standard component. The text-driven scene-editing application (Sec. 5) uses an off-the-shelf 2D image editor whose internal model may be a vision-language model, but it is treated as a black-box module and is not contributed by this work.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.