Perceptual 3D Simulation With Physical World Modeling

Supplementary Material

Overview of the Supplementary Material

This supplementary material provides additional technical details and qualitative results that complement the main paper.

- **Details of the Physical World Model** Ψ A.1 describes the model architecture, quantizer, and training setup.
- **Details of the Geometrizer** Γ A.2 explains how geometric conditioning signals such as flow and partial depth are constructed.
- **Details of the Persistent Memory Module** μ A.3 presents the mechanisms for incremental scene reasoning and occupancy update.
- **Additional Examples of Extracting a Complete Scene Description** A.4 illustrates the model's behavior in extracting, completing, and stabilizing 3D structure across time.

A.1. Details of the Physical World Model Ψ

This section provides implementation details of the physical world model Ψ. We first describe how local discrete tokens are constructed using Hierarchical Local Quantization (HLQ), followed by how multimodal tokens (RGB, depth, and optical flow) are serialized via Local Random Access Sequences (LRAS). We then explain how a transformer is trained to model their joint distribution. Next, we summarize the sequence-construction rules that mix modalities and temporal contexts, and present a probabilistic interpretation that clarifies how LRAS enables tractable conditional inference. We conclude with the inference settings used during quantitative evaluations.

Hierarchical Local Quantization. The world model operates on locally quantized tokens produced by a Hierarchical Local Quantizer (HLQ), a convolutional autoencoder whose receptive field is strictly contained within each patch. This design guarantees that no information ever propagates across patch boundaries during encoding. Each frame is divided into non-overlapping 16×16 patches, and each patch is represented by a short sequence of discrete codes. A coarse token that reconstructs a low-resolution preview, followed by several refinement tokens that add high-frequency detail. Because all information is confined strictly within a patch, the resulting codes behave predictably under interventions such as masking or overwriting.

This strict patch-level locality plays a central functional role in controllable inference. Because no information leaks across patch boundaries during quantization, local interventions, masking a patch, overwriting it with a counterfactual value, or re-ordering its decoding, produce localized and predictable effects. This is essential for the model's ability to perform sparse conditioning, patch regeneration, counterfactual edits, and user-guided generation using arbitrary pointer sequences.

Quantizer training. We train separate HLQ quantizers for RGB, depth, and optical flow using a patch-local encoder with a multi-codebook FSQ quantization [5] layer and a wavelet input transformation. The decoder is convolutional and nonlocal. All models are optimized on GPUs with a batch size of 512 and a learning rate of 10^{-4} until convergence. RGB quantization uses ImageNet and OpenImages with an ℓ_1 reconstruction loss and a DINOv2 perceptual loss, producing four codes per patch. A coarse head is supervised with an ℓ_1 loss of weight 10^{-2} on a 32×32 target. Depth maps used for training are generated by VideoDepthAnything [1]. The depth quantizer follows the same architecture and losses as RGB but produces two codes per patch. Optical flow fields are computed using DPFlow [6], with UFM [13] added for large-displacement and static-NVS cases. The flow quantizer is supervised with ℓ_2 reconstruction losses and an ℓ_2 coarse loss of weight 10^{-2}, and likewise produces two codes per patch. The resulting discrete RGB, depth, and flow tokens form the unified vocabulary consumed by Ψ for multimodal autoregressive modeling.

Autoregressive Transformer. The physical world model Ψ is a 7B-parameter decoder-only transformer trained to model the joint distribution over RGB, depth, and optical-flow tokens. Each discrete token is mapped to a learned embedding, augmented with a modality embedding and a spatiotemporal coordinate (x, y, t, c) indicating its patch location, frame index, and channel type. These coordinates are projected into a continuous embedding and added to the token representation before entering the transformer. The backbone uses standard residual-attention blocks with RMS normalization.

Rather than using raster-order flattening, Ψ employs a Local Random Access Sequence (LRAS) that randomly permutes patch locations during training and inference. LRAS exposes tokens to diverse spatial contexts, improves robustness under partial observations, and enables spatially local conditioning, which is an essential capability because geometric signals such as flow or partial depth are sparse and may occur at arbitrary locations.

Sequence Model Training. We train Ψ with next-token prediction cross-entropy loss, supervising only the content tokens and not the associated pointer tokens. The training corpus contains approximately 3 million video clips, corresponding to roughly 1.4 trillion multimodal tokens. The dataset spans indoor, outdoor, egocentric, and object-centric settings, combining large-scale sources (e.g., ScanNet++ [11], CO3D [8], RealEstate10K [14], MVImgNet [12], DL3DV [4], and EgoExo4D [2]) with curated internet videos selected for scene diversity and long-range dynamics. Training is conducted entirely on a TPU v5–256 pod through the Google TPU Research Cloud (TRC). We use a global batch size of 512 for a total of 1.5M optimization steps. Each training step requires approximately 3.8 seconds. The learning rate follows a Warmup–Stable–Decay schedule, increasing to a peak value of 3×10^{-4}, remaining constant during the stable phase, and decaying to zero over the final 100K steps. The default sequence length is 4096 tokens, and we increase it to 8192 for the final 20K training steps to improve the model's ability to generate longer rollouts. All training uses mixed precision, FSDP sharding, and gradient checkpointing.

Sequence Design. To support a unified autoregressive model capable of reasoning under heterogeneous perceptual inputs, we design a sequence construction pipeline that mixes RGB, depth, and optical flow tokens across a wide range of temporal configurations. Each training sample is produced by selecting a mode that specifies (i) the set of modalities to include, (ii) the number of frames (two to four), and (iii) the temporal spacing. Given the chosen mode, the dataset dynamically loads the quantized patch-level token blocks of RGB, depth, and optical flow.

A constrained resource allocator then assigns token budgets to each modality–frame pair, using hard capacity limits to ensure that the combined sequence fits within the global token limit (4096 or 8192 tokens). This allocation determines how many patches to keep for each signal before shuffling. Frame indices are sampled according to the specified gap regime, and each patch is annotated with a rotary-position index that encodes spatial location, temporal position, and channel group [9]. Depth tokens may appear before or after RGB tokens, and flow tokens are placed between the RGB frames they connect.

During target construction, the first frame and the pointer tokens are never supervised. The resulting sequence is a heterogeneous but structurally consistent stream of pointer–value tokens that exposes the model to diverse inference conditions, ranging from fully observed RGB sequences to diverse modality scenarios. This diversity is essential for teaching the model to perform general-purpose probabilistic inference over multimodal 3D scene variables.

Probabilistic graphical model interpretation. In our formulation, each pointer token $p \in \mathcal{P}$ identifies a scene variable x_p whose value lies in $\mathcal{V}$. An observed token–value pair (p, v) therefore corresponds to observing the variable $x_p = v$. The full set $\{x_p\}_{p\in\mathcal{P}}$ forms an implicit probabilistic graphical model (PGM) over all multimodal scene elements (RGB, depth, and flow). A Local Random-Access Sequence (LRAS) provides a traversal of this PGM. As the serialized sequence progresses, previously observed or generated pairs (p_i, v_i) become conditioning variables for predicting the next variable x_p. The autoregressive factorization

$$\Psi(X \circ p) \;=\; \Pr\big[\, v_k \;\mid\; p_0, v_0, \ldots, p_{k-1}, v_{k-1}, p \,\big]$$

acts as a tractable approximation to the joint distribution over $\{x_p\}_{p\in\mathcal{P}}$, avoiding explicit global inference over exponentially many conditioning subsets. Importantly, this formulation allows Ψ to compute conditional distributions of the form

$$p(x_p \mid X) \qquad \text{or more generally} \qquad p(x_i \mid x_S),$$

for arbitrary subsets $S \subset \mathcal{P}$ simply by providing the observed pointer–value pairs $(p, X(p))$ for $p \in S$ as observed evidence and autoregressively decoding all remaining variables. This capability enables a broad family of conditional inference pathways—including, but not limited to, the flow–depth prompting regimes described in the main paper for novel-view synthesis, rigid and deformable object manipulation, object interactions, and motion prediction. In general, any subset of multimodal observations (RGB, depth, or optical flow) can be supplied as evidence, allowing a single sequence model to flexibly adapt its inference behavior to diverse 3D reasoning tasks under a unified probabilistic architecture.

Inference settings. All evaluations use sequential decoding with a temperature of 1.0, top-$p = 0.9$, and top-$k = 1000$. Unless otherwise specified, single-image depth is predicted using the metric-depth model Depth Anything V2 [10]. The metric depth is then converted to disparity and normalized so that the conditioning disparity has a maximum scale of 0.6.

For novel view synthesis (NVS) evaluation, we use Depth Anything 3 because we empirically found that Depth Anything V2 tended to produce more distorted object shapes on DTU in our setup. We additionally estimate a camera pose translation scale factor per scene using UFM and single-view depth from Depth Anything 3 [3]. Specifically, we compute optical flow by warping the predicted depth using the ground-truth camera pose under candidate scales, compare it against UFM flow on covisible regions of the ground-truth video, and optimize the scalar scene scale

that minimizes this flow discrepancy. This estimated scale places the ground-truth camera motion in the same metric space as the single-view predicted depth.

A.2. Details of Geometrizer Γ

In this section, we describe the details of the geometrizer Γ. Given a depth history $\{D_{0:t-1}\}$, camera intrinsics K, a target camera pose P_t, and object-level SE(3) transformations $\{\mathcal{T}_t^{(o)}\}$, the geometrizer constructs geometric conditioning signals

$$(F_{t-1\to t},\, D_t^{\text{sparse}}).$$

Here, $F_{t-1\to t}$ is the induced optical flow and D_t^{sparse} is the rendered sparse depth. These signals convert the specified camera motion and per-object transformations into sparse depth and optical-flow fields that constrain the world model during autoregressive decoding.

The object transformations $\{\mathcal{T}_t^{(o)}\}$ are defined with respect to user-provided segmentations. Each object's segmentation mask identifies the pixels belonging to that object, allowing its motion to be applied independently during geometry rendering.

Surface and undersurface meshes per object. We begin by reconstructing the reference-frame geometry

$$\mathcal{G}_0 = \text{BackProject}(D_0,\, T_0),$$

where T_0 contains the reference camera pose and intrinsics. For each pixel $u = (u_x, u_y)$, let $X_0(u) \in \mathcal{G}_0$ denote the corresponding 3D point.

We partition this geometry according to the segmentation mask:

$$\mathcal{G}_0^{(o)} = \{\, X_0(u) \mid \text{seg}_0(u) = o \,\},$$

for each object label o, including background.

Surface meshes. For each object o, we construct a surface mesh $\mathcal{S}_0^{(o)}$ by triangulating adjacent pixels whose segmentation label equals o. We compute triangle normals $n(u)$ and apply the surface-orientation criterion

$$\theta(u) = \cos^{-1}(n(u) \cdot (-v(u))) < \theta_{\text{th}}, \qquad \theta_{\text{th}} = 80^\circ,$$

where $v(u)$ is the viewing direction. Only triangles satisfying this condition are retained.

Undersurface meshes. We also associate each object o with an undersurface mesh $\mathcal{U}_0^{(o)}$ that parameterizes the locally unobserved volume behind the visible scene. To initialize these meshes, we first build a single occupancy volume from the full-frame depth map D_0, assigning observed space the value $+1$ and unobserved space the value -1. We carve this volume using an offset depth map $D_0(u)+\varepsilon$ (with $\varepsilon = 3 \times 10^{-2}$), ensuring that the resulting undersurface remains strictly behind the observed surface. Extracting the 0-level isosurface via marching cubes then yields a global undersurface mesh $\tilde{\mathcal{U}}_0$ that lies just behind the visible surfaces $\mathcal{S}_0^{(o)}$ along each viewing ray. Because the structure of the unobserved volume is not yet known, every object (including the background) is initialized with a copy of this geometry,

$$\mathcal{U}_0^{(o)} = \tilde{\mathcal{U}}_0 \quad \forall o.$$

As the scene evolves and objects move, each $\mathcal{U}_t^{(o)}$ is updated independently based on the object motion and new depth observations, so the undersurface meshes gradually diverge.

Each object (including background) thus starts with paired geometry

$$(\mathcal{S}_0^{(o)}, \mathcal{U}_0^{(o)}),$$

where the undersurface meshes share the same initial shape but are subsequently deformed and carved in an object-specific manner.

Composite geometry and object motion. At time t, we apply the specified object motions:

$$\mathcal{S}_t^{(o)} = \mathcal{T}_t^{(o)}\, \mathcal{S}_0^{(o)}, \qquad \mathcal{U}_t^{(o)} = \mathcal{T}_t^{(o)}\, \mathcal{U}_0^{(o)}.$$

The full scene geometry is the union

$$\mathcal{M}_t = \bigcup_o \left(\mathcal{S}_t^{(o)} \cup \mathcal{U}_t^{(o)} \right),$$

ensuring that each object's surface and undersurface geometry move consistently with its SE(3) transformation.

Rendering sparse target depth. Rendering $\mathcal{M}_t$ from the target pose P_t produces a depth value $z(u)$ for rays that hit a surface mesh. We define

$$D_t^{\text{sparse}}(u) = \begin{cases} z(u), & \text{if the first hit is a surface mesh,} \\ \text{invalid}, & \text{otherwise.} \end{cases}$$

In practice, surface meshes are rendered in white and undersurface meshes in black. Any pixel that renders black is marked invalid. A patch is invalid if any of its pixels is invalid. We use `pyrender`, though any renderer is suitable.

Computing optical flow. We reconstruct the previous-frame geometry

$$\mathcal{G}_{t-1} = \text{BackProject}(D_{t-1},\, T_{t-1}),$$

and let $X_{t-1}(u) \in \mathcal{G}_{t-1}$ be the 3D point associated with pixel u.

Its object label is $\text{seg}_{t-1}(u)$, so the corresponding SE(3) motion is $\mathcal{T}_t^{(\text{seg}_{t-1}(u))}$. We apply this motion and reproject:

$$X_t(u) = \mathcal{T}_t^{(\text{seg}_{t-1}(u))}\, X_{t-1}(u),$$

$$u_t = \mathrm{Project}(P_t\, X_t(u),\, K)\,.$$

The optical flow is

$$F_{t-1\to t}(u) = u_t - u.$$

Pixels with unknown motion (e.g., unlabeled segments or masked regions) are excluded from the conditioning.

Motion scenarios. We summarize how Γ constructs conditioning signals for different motion cases.

Static scenes (novel view synthesis). Only the camera pose changes while all objects remain fixed. The surface and undersurface meshes remain at their reference-frame positions, and the composite geometry is rendered directly from P_t. Valid partial depth$_1$ corresponds to rays whose first intersection is a surface in the reference geometry, excluding occluded or undersurface hits. Flow is obtained by transforming reference-frame 3D points $X_0(u)$ into the target camera and measuring their 2D displacement.

Dynamic scenes with fully known motion. When both camera and object-level SE(3) motions $\{\mathcal{T}_t^{(o)}\}$ are provided, each mesh pair $(\mathcal{S}_0^{(o)}, \mathcal{U}_0^{(o)})$ is displaced independently:

$$\mathcal{S}_t^{(o)} = \mathcal{T}_t^{(o)}\mathcal{S}_0^{(o)}, \qquad \mathcal{U}_t^{(o)} = \mathcal{T}_t^{(o)}\mathcal{U}_0^{(o)}.$$

Rendering the resulting composite geometry $\mathcal{M}_t$ from P_t defines partial depth$_1$, and flow is computed by applying the corresponding object motion to each previous-frame point $X_{t-1}(u)$ before reprojecting into the target view.

Dynamic scenes with partially known motion. If only a subset of object motions is specified, partial depth$_1$ is constructed only from the objects with known SE(3) transformations. These objects have their surface meshes transformed and rendered, and all other surfaces are treated as unknown. Rays whose first hit belongs to an object with unknown motion are marked invalid and do not contribute to partial depth. Flow is defined in the same way. Pixels whose object labels correspond to unknown motion are masked, and the remaining regions follow the procedure used in the fully known-motion setting.

A.3. Details of Persistent Memory Module μ

This section provides an implementation of the abstract memory-update rules described in the main paper, specifying how the memory is represented using surface meshes and occupancy-based undersurface meshes. The persistent memory μ_t maintains, for each object o, a surface mesh $\mathcal{S}_t^{(o)}$ and an undersurface mesh $\mathcal{U}_t^{(o)}$ derived from an occupancy volume that tracks unobserved space. All geometry is stored in a globally aligned reference frame and updated using object-level SE(3) transformations.

Surface–mesh updating per object. From the depth map D_t and pose P_t, we obtain a point cloud

$$\mathcal{S}_t = \mathrm{BackProject}(D_t,\, T_t).$$

Each point is assigned to an object via the segmentation of frame t, obtained using SAM2 [7] with user-provided point prompts.

All previously stored surface meshes are transformed into the current frame:

$$\hat{\mathcal{S}}_{t-1}^{(o)} = \mathcal{T}_t^{(o)}\left(\mathcal{S}_{t-1}^{(o)}\right),$$

and fused with the newly observed surface points $\mathcal{S}_t^{(o)}$ to produce the updated surface mesh $\mathcal{S}_t^{(o)}$. This ensures that surfaces move coherently according to the object's segmentation and SE(3) motion. Because the system knows which regions are newly observed versus previously visible under the target viewpoint, it can optionally choose to overwrite, refine, or preserve the pre-existing surface regions during fusion.

Undersurface–mesh updating per object. Each object maintains an occupancy volume $\Phi_{t-1}^{(o)}(x) \in \{-1, +1\}$, where $+1$ denotes observed (free) space and -1 denotes unobserved space. Its zero-level isosurface defines the undersurface mesh $\mathcal{U}_{t-1}^{(o)}$.

Before carving, the occupancy is transported into the current frame:

$$\hat{\Phi}_{t-1}^{(o)}(x) = \Phi_{t-1}^{(o)}\left((\mathcal{T}_t^{(o)})^{-1}(x)\right).$$

Given D_t, we carve the occupancy volume ray-by-ray using an offset depth $D_t(u) + \varepsilon$ (with $\varepsilon = 3 \times 10^{-2}$), which guarantees that the undersurface mesh stays just behind the visible surface. For any voxel x lying in front of this offset depth along ray u,

$$\Phi_t^{(o)}(x) = +1, \qquad \text{(carved: now observed space)},$$

and otherwise

$$\Phi_t^{(o)}(x) = \hat{\Phi}_{t-1}^{(o)}(x).$$

The updated undersurface mesh is then extracted as the 0-level isosurface from $\Phi_t^{(o)}$ using marching cubes. This yields a geometry-coherent boundary of the still-unobserved volume behind the object at time t. This occupancy-based update enforces consistency with both past and current observations. Carving with the current depth removes regions inconsistent with the present view, and the monotonicity of Φ_t prevents reintroducing regions that were already ruled out by earlier viewpoints.

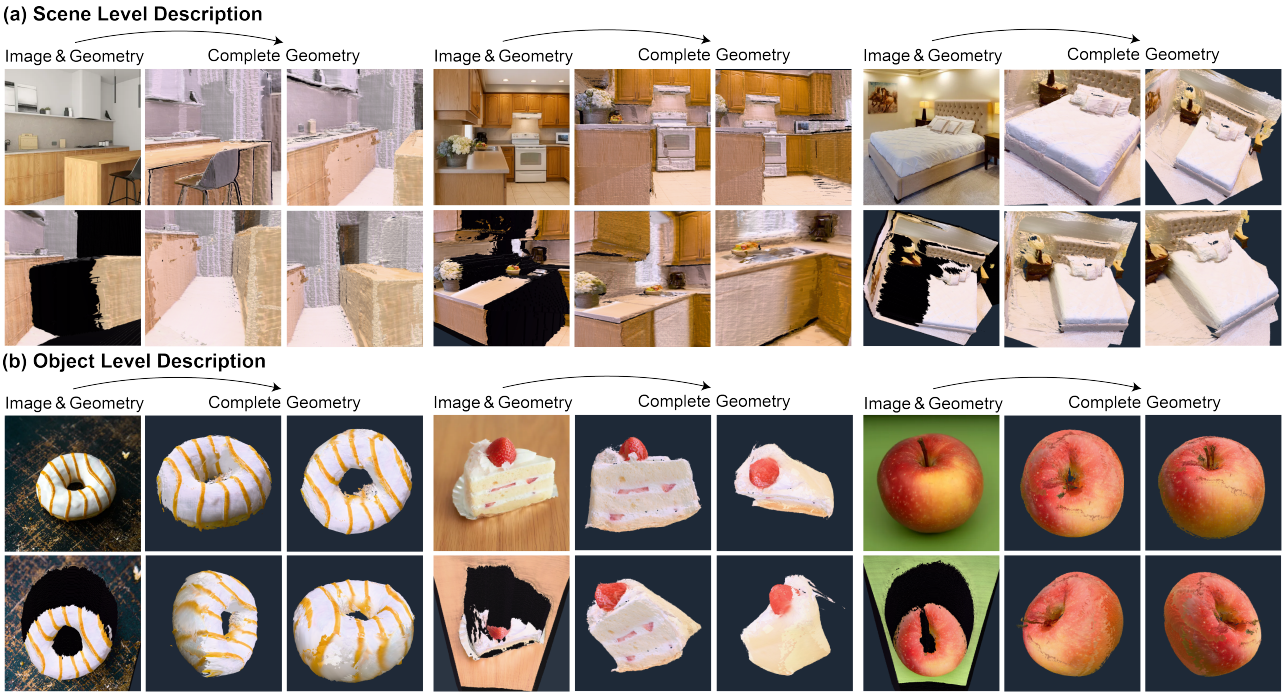

Figure A1. Additional Examples of Extracting a Complete Scene Description. (a) *Scene-level description:* additional examples of coherent 3D scene layout inferred from single images. (b) *Object-level description:* further examples of consistent object-surface reconstruction.

A.4. Additional Examples of Extracting a Complete Scene Description

We present additional qualitative results demonstrating geometry extraction from single images in Figure A1. For the first frame, depth D_0 is estimated using Depth Anything V2. The next frame depth is sampled as $D_1 \sim \Psi(I_0, D_0, F_{0\to1}, D_1^{\text{sparse}})$, followed by image synthesis $I_1 \sim \Psi(I_0, D_0, F_{0\to1}, D_1)$. Subsequent steps follow the analogous pattern $D_2 \sim \Psi(I_0, I_1, D_1, F_{1\to2}, D_2^{\text{sparse}})$ and $I_2 \sim \Psi(I_0, I_1, D_1, F_{1\to2}, D_2)$. The first frame I_0 remains fixed, while I_1 is continually replaced by the frame closest to the current target.

Throughout inference, the memory module μ is updated at every step so that geometric information accumulates across frames. This procedure recovers both scene-level geometry, useful for navigation, and object-level geometry, useful for object-centric tasks. For object-level visualization, we show only the surface meshes associated with each object according to the update rules of the memory module μ.

References

[1] Sili Chen, Hengkai Guo, Shengnan Zhu, Feihu Zhang, Zilong Huang, Jiashi Feng, and Bingyi Kang. Video depth anything: Consistent depth estimation for super-long videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22831–22840, 2025.

[2] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, et al. Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19383–19400, 2024.

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

[4] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22160–22169, 2024.

[5] Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: Vq-vae made simple. *arXiv preprint arXiv:2309.15505*, 2023.

[6] Henrique Morimitsu, Xiaobin Zhu, Roberto M Cesar, Xiangyang Ji, and Xu-Cheng Yin. Dpflow: Adaptive optical flow estimation with a dual-pyramid framework. In *Proceed-*

ings of the Computer Vision and Pattern Recognition Conference, pages 17810–17820, 2025.

[7] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

[8] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10901–10911, 2021.

[9] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[10] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in Neural Information Processing Systems*, 37:21875–21911, 2024.

[11] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–22, 2023.

[12] Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Chenming Zhu, Zhangyang Xiong, Tianyou Liang, et al. Mvimgnet: A large-scale dataset of multi-view images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9150–9161, 2023.

[13] Yuchen Zhang, Nikhil Keetha, Chenwei Lyu, Bhuvan Jhamb, Yutian Chen, Yuheng Qiu, Jay Karhade, Shreyas Jha, Yaoyu Hu, Deva Ramanan, et al. Ufm: A simple path towards unified dense correspondence with flow. *arXiv preprint arXiv:2506.09278*, 2025.

[14] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018.