

Efficient Construction of Implicit Surface Models From a Single Image for Motion Generation

Wei-Teng Chu¹Tianyi Zhang²Matthew Johnson-Roberson^{3,4}Weiming Zhi^{*,3,4,5}

Abstract—Implicit representations have been widely applied in robotics for obstacle avoidance and path planning. In this paper, we explore the problem of constructing an implicit distance representation from a single image. Past methods for implicit surface reconstruction, such as NeuS and its variants generally require a large set of multi-view images as input, and require long training times. In this work, we propose Fast Image-to-Neural Surface (FINS), a lightweight framework that can reconstruct high-fidelity surfaces and SDF fields based on a single or a small set of images. FINS integrates a multi-resolution hash grid encoder with lightweight geometry and color heads, making the training via an approximate second-order optimizer highly efficient and capable of converging within a few seconds. Additionally, we achieve the construction of a neural surface requiring only a single RGB image, by leveraging pre-trained foundation models to estimate the geometry inherent in the image. Our experiments demonstrate that under the same conditions, our method outperforms state-of-the-art baselines in both convergence speed and accuracy on surface reconstruction and SDF field estimation. Moreover, we demonstrate the applicability of FINS for robot surface following tasks and show its scalability to a variety of benchmark datasets.

I. INTRODUCTION

For autonomous robots to navigate and interact safely with the real world, they must form reliable geometric representations of their surroundings. Distance-based representations are a powerful representation widely used in motion planning and obstacle avoidance [1]–[8]. Accurate and efficient SDF estimation is therefore a key enabler of downstream decision-making and control.

Recent neural implicit surface methods, such as NeuS [9] and its successors [10]–[14], have demonstrated impressive capability in reconstructing fine-grained object surfaces. However, these approaches suffer from two key drawbacks: (i) they rely on dense multi-view supervision, which is impractical in robotics settings where only sparse observations are available; and (ii) they require long training times, from minutes to hours, making them unsuitable for real-time use in navigation or manipulation. A complementary line of work [15]–[18] has sought to improve generalization to sparse views, reducing the dependency on extensive image collections. However, these approaches can often still require a sizeable number of images, and can be relatively inefficient to train from scratch.

In this paper, we introduce *Fast Image-to-Neural Surface (FINS)*, a lightweight framework that overcomes these limita-



Fig. 1: We present Fast Image-to-Neural Surface (FINS), an efficient framework (~ 10 s on consumer-grade hardware) that can reconstruct high-fidelity surfaces and SDF fields based on sparse or even a single image. Top row: Input RGB image of a statue (left), and the corresponding implicit representation enabling robot motion to trace on the surface. Next two rows from left to right: A single image input for SDF field reconstruction; The result mesh; The result colored mesh; The top view of the colored mesh; The trained SDF iso-contours corresponding to the top view.

tions. FINS reconstructs high-fidelity surfaces and SDF fields from as few as *a single image*, or a small set of images, within seconds. Our framework integrates three components: (1) off-the-shelf 3D foundation models, such as DUST3R [19] and VGGT [20], to lift single-view inputs into point clouds for SDF supervision; (2) a multi-resolution hash grid encoder [21] to enable efficient feature encoding; and (3) lightweight geometry and color heads trained with an approximate second-order optimizer, yielding rapid convergence.

By leveraging strong priors from pre-trained 3D models, FINS scales naturally from single objects to multi-view, scene-level settings. This flexibility supports deployment on mobile platforms, where continuous observations can be assimilated into an evolving SDF representation. As a result, FINS enables real-time reconstruction and refinement of neural surfaces for downstream robotics tasks such as obstacle avoidance, path planning, and surface following. We empirically evaluate the quality and efficiency of building implicit distance reconstructions of a diverse range of objects, and demonstrate the applicability of these representations for robot surface following [22]. Concretely, this paper presents the following technical contributions:

- 1) We propose FINS, an end-to-end method that achieves high-precision SDF training from a single image in only

¹ Department of Electrical Engineering, Stanford University, USA.

² Aurora, USA.

³ College of Connected Computing, Vanderbilt University, USA.

⁴ Robotics Institute, Carnegie Mellon University, USA.

⁵ School of Computer Science, The University of Sydney, Australia.

*Correspondence to wzhi@andrew.cmu.edu.

- a few seconds.
- 2) We leverage pre-trained 3D foundation models to generate point clouds for SDF supervision, enabling efficient and complete reconstruction with limited visual input.
 - 3) We adopt multi-resolution hash encoding and lightweight geometry/color heads with a mixed optimization strategy to eliminate heavy optimization and enable real-time convergence.

II. RELATED WORK

Neural Implicit Surface Reconstruction: Representations are critical in robotics [23]–[26]. Neural implicit representations have become a dominant approach for 3D surface reconstruction. NeRF [27] pioneered neural radiance fields by learning volumetric scene representations with differentiable rendering. Subsequent works incorporated signed distance functions (SDFs) into the rendering pipeline, including VolSDF [11], NeuS [9], and NeuS2 [10], which enabled more consistent and detailed surface reconstruction. Extensions such as IDR [12], UNISurf [13], and GSurf [14] further improved generalization and fidelity. Despite their success, these methods typically require dense multi-view supervision and lengthy optimization (tens of minutes to hours).

Sparse-View and Generalizable Reconstruction: To relax the dependence on dense viewpoints, several methods target sparse-view reconstruction. GenS [15] enforced multi-scale feature-metric consistency to improve reconstruction under limited inputs. SparseNeuS [16] combined multi-level geometry reasoning with color blending and consistency-aware fine-tuning to enhance robustness to sparse views. SparseCraft [17] reconstructed detailed surfaces from very few inputs in under ten minutes, while SurfaceSplat [18] hybridized SDF-based and Gaussian splatting approaches for high-fidelity meshes. These methods significantly improve surface reconstruction from limited inputs. Other methods such as VGER [28] looked at integrating video generators to push image sparsity. However, their focus remains on recovering meshes or surfaces, rather than constructing complete SDF fields, thereby limiting their utility for robotics tasks such as continuous collision checking and motion planning.

3D Foundation Models: Large-scale 3D foundation models have recently shown remarkable progress in geometry estimation from sparse observations. DUSt3R [19] introduced a transformer-based architecture for dense correspondence and depth estimation, with subsequent improvements such as MONSt3R [29]. An efficient model, VGGT [20] learned generic 3D geometric priors from large-scale training. These models can generate high-quality point clouds from only a few images, providing strong geometric priors at low computational cost. 3D foundation models have been used within calibration [30], photorealistic reconstruction [31], pose estimation [32]. However, these models do not directly yield SDF fields. Instead, they serve as powerful geometric initializations that can be integrated with implicit representations, making it feasible to construct accurate SDFs from limited views. This integration motivates our approach, where we leverage

3D foundation priors to enable real-time, single-image SDF reconstruction.

III. PRELIMINARIES

A. Multi-Resolution Hash Grid Encoding

A fundamental problem in implicit neural field learning is how to embed spatial coordinates $\mathbf{x} \in \mathbb{R}^3$ into a high-dimensional representation that preserves geometric detail across scales. Standard sinusoidal positional encodings map coordinates to Fourier features, which expand the input into a fixed set of bases. While effective, this requires $\mathcal{O}(K)$ parameters per input dimension and leads to slow convergence in practice.

Instant-NGP [21] addresses this with a *multi-resolution hash grid encoding*. Let $\{r_\ell\}_{\ell=1}^L$ denote a set of resolutions, where the ℓ -th grid partitions space into r_ℓ^3 cells. For a point \mathbf{x} , its grid coordinates at level ℓ are

$$\mathbf{u}_\ell = r_\ell \cdot \mathbf{x}, \quad \mathbf{i}_\ell = \lfloor \mathbf{u}_\ell \rfloor, \quad \delta_\ell = \mathbf{u}_\ell - \mathbf{i}_\ell, \quad (1)$$

where $\mathbf{i}_\ell \in \mathbb{Z}^3$ is the voxel index and $\delta_\ell \in [0, 1)^3$ is the local interpolation weight. Each grid vertex $\mathbf{v} \in \{\mathbf{i}_\ell + \mathbf{b} \mid \mathbf{b} \in \{0, 1\}^3\}$ is mapped to an embedding $\mathbf{e}_\ell(\mathbf{v}) \in \mathbb{R}^F$ via a hash function

$$\mathbf{e}_\ell(\mathbf{v}) = \Theta_\ell[h_\ell(\mathbf{v})], \quad h_\ell : \mathbb{Z}^3 \rightarrow \{1, \dots, T\}, \quad (2)$$

where $\Theta_\ell \in \mathbb{R}^{T \times F}$ is a trainable hash table shared across the grid and $T \ll r_\ell^3$. The feature vector $\phi_\ell(\mathbf{x}) \in \mathbb{R}^F$ is computed by trilinear interpolation. Finally, the embeddings across L levels are concatenated into a multi-scale representation

$$E(\mathbf{x}) = [\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \dots, \phi_L(\mathbf{x})] \in \mathbb{R}^{L \cdot F}. \quad (3)$$

This scheme simultaneously encodes low-frequency structure (from coarse grids) and high-frequency detail (from fine grids) with constant memory $\mathcal{O}(L \cdot T \cdot F)$, independent of the native grid resolution. Empirically, this yields orders-of-magnitude faster convergence than Fourier features while maintaining compact parameterization.

B. Approximate Second-Order Optimization

Optimization dynamics significantly affect the quality of learned signed distance fields (SDFs). First-order optimizers such as SGD, AdamW, or Lion [33] update parameters θ using only gradient information:

$$\theta_{t+1} = \theta_t - \eta \hat{g}_t, \quad \hat{g}_t \approx \nabla_\theta \mathcal{L}(\theta_t), \quad (4)$$

where \mathcal{L} is the training loss and η the learning rate. While cheap, such updates fail to account for the curvature of \mathcal{L} , leading to slow progress along high-curvature directions.

Second-order methods instead rescale the update using the inverse Hessian matrix H^{-1} :

$$\theta_{t+1} = \theta_t - \eta H^{-1} \nabla_\theta \mathcal{L}(\theta_t), \quad (5)$$

which preconditions the gradient according to the local geometry of the parameter space. However, computing and inverting H exactly is intractable for large networks.

Kronecker-Factored Approximate Curvature (K-FAC) [34] factorizes each Fisher block layer-wise. For a fully connected layer with weights $W \in \mathbb{R}^{m \times n}$, inputs $a \in \mathbb{R}^n$, and pre-



Fig. 2: we can leverage 3D foundation models to lift the skull image (shown in fig. 1) to a 3D point cloud (left), then leverage confidence estimates to further filter and clean the point cloud (right).

activation gradients $g \in \mathbb{R}^m$, the Hessian block is approximated as

$$\begin{aligned} H_W &\approx A \otimes G, \quad A = \mathbb{E}[aa^\top] \in \mathbb{R}^{n \times n}, \\ G &= \mathbb{E}[gg^\top] \in \mathbb{R}^{m \times m}. \end{aligned} \quad (6)$$

The Kronecker structure allows efficient inversion:

$$H_W^{-1} \approx A^{-1} \otimes G^{-1}. \quad (7)$$

The natural gradient update for W becomes

$$\text{vec}(\Delta W) = -\eta (A^{-1} \otimes G^{-1}) \text{vec}(\nabla_W \mathcal{L}), \quad (8)$$

where $\text{vec}(\cdot)$ denotes vectorization. This reduces the computational complexity from $\mathcal{O}(mn^3)$ to $\mathcal{O}(m^3 + n^3)$ per layer, making curvature-aware optimization feasible at scale. In practice, K-FAC stabilizes training and accelerates convergence, serving as an effective middle ground between purely first-order and exact second-order methods in implicit field optimization.

IV. THE FAST IMAGE-TO-NEURAL SURFACE FRAMEWORK

A. Problem Formulation

We address the task of learning a signed distance field (SDF) and corresponding appearance directly from sparse image observations. The input to our method is a single image or a small set of images, which are converted into 3D point clouds using off-the-shelf 3D foundation models. These point clouds serve as supervision for training the SDF.

Formally, each training input is a 3D coordinate

$$\mathbf{x} = [x, y, z]^\top \in \mathbb{R}^3, \quad (9)$$

and the network maps this point to a 4D output vector containing the signed distance and RGB color values:

$$f : \mathbb{R}^3 \rightarrow \mathbb{R}^4, \quad f(\mathbf{x}) = [d(\mathbf{x}), r(\mathbf{x}), g(\mathbf{x}), b(\mathbf{x})]^\top. \quad (10)$$

B. Preprocessing with 3D Foundation Models

Given an single RGB image or a small set of input images of an object or scene, we employ 3D foundation models such as DUST3R [19] or VGGT [20] to generate a colored point cloud. Each pixel in the input image is mapped to a 3D point with associated color and confidence, together with estimated camera intrinsics, extrinsics, and pose. The resulting point cloud therefore includes both object and background regions.

To improve the quality of supervision, we filter out low-confidence points using the per-pixel confidence values predicted by the foundation model. This removes unreliable regions of the cloud while retaining dense and geometrically consistent supervision for SDF training.

C. Model Design

Our SDF network is an implicit neural representation consisting of a shared multi-resolution hash grid encoder and two prediction heads: a geometry branch (GeoNet) and a color branch (ColorNet). The encoder provides a compact, multi-scale embedding of the input coordinate, while the heads separately predict geometry and appearance.

1) *Multi-Resolution Hash Grid Encoding*: To efficiently capture both coarse and fine geometric details, we employ the multi-resolution hash grid encoding proposed by Instant-NGP [21], which has since become standard in neural implicit surface modeling [10], [16]. Our implementation uses $L = 10$ resolution levels, each producing a feature vector of dimension $F = 4$. The hash table size is $T = 2^{16}$ entries, with base resolution $R_{\text{base}} = 14$ and a per-level scaling factor of $s = 1.5$. This design allows the encoder to represent both low-frequency structure and high-frequency detail with a compact parameter budget.

2) *Geometry & Color Heads*: The concatenated features from the hash encoder are processed by two lightweight branches:

- **Geometry branch (GeoNet)**: a two-layer MLP with Softplus activations that outputs the predicted signed distance $d(\mathbf{x})$.
- **Color branch (ColorNet)**: a single linear layer that outputs the predicted RGB color vector $[r(\mathbf{x}), g(\mathbf{x}), b(\mathbf{x})]^\top$.

Separating geometry and appearance has been noted in [9], [10] to improve training stability.

D. Optimization Strategy

A key novelty in FINS is an efficient *staged hybrid optimization scheme*. We observe that the parameter sizes of the geometry and color networks can be sufficiently small to be trained via approximate second-order optimization. Here, we devise:

- **Warm-up stage (first 60% of epochs)**: all parameters are trained end-to-end with a standard first order optimizer. Here, we use the momentum-based first-order method, the Lion optimizer [33].
- **Rapid Convergence (final 40% of epochs)**: the shared encoder continues to be updated using Lion, while the geometry and color heads are optimized using K-FAC [34], a Kronecker-factored approximation to second-order optimization. This enables curvature-aware updates for the prediction heads while keeping encoder updates efficient.

This staged strategy balances rapid early learning with stable late-stage convergence, yielding both faster training and higher reconstruction accuracy.

E. Training Objectives

To jointly recover geometry and appearance, we adopt a composite multi-objective loss. The SDF must satisfy both

local geometric fidelity (accurate surfaces) and global signed-distance consistency, while also matching photometric observations. A single objective is insufficient, so we combine complementary terms that enforce surface reconstruction, global regularization, and appearance supervision.

The full objective is expressed compactly as:

$$\mathcal{L} = \sum_{t \in \mathcal{T}_{\text{surf}}} w_t \mathcal{L}_t + \sum_{t \in \mathcal{T}_{\text{reg}}} w_t \mathcal{L}_t + \sum_{t \in \mathcal{T}_{\text{rgb}}} w_t \mathcal{L}_t, \quad (11)$$

where w_t are scalar weights. The index sets are

$$\begin{aligned} \mathcal{T}_{\text{surf}} &= \{\text{SDF, zero, eik-surf, normal}\}, \\ \mathcal{T}_{\text{reg}} &= \{\text{eik-glob, sparse, off-surf}\}, \\ \mathcal{T}_{\text{rgb}} &= \{\text{rgb}\}. \end{aligned}$$

Each component loss term \mathcal{L}_t plays a distinct role.

SDF Loss: Supervises the predicted signed distance against ground-truth values at noisy points.

$$\mathcal{L}_{\text{SDF}} = \frac{1}{\sum_i w_i} \sum_{i \in \mathcal{N}} w_i \left(d_\theta(x_i^{\text{noise}}) - d_i \right)^2. \quad (12)$$

This ensures that the predicted SDF values faithfully reproduce the metric distances provided by supervision.

Zero Loss: Encourages surface points to lie close to the zero-level set of the SDF.

$$\mathcal{L}_{\text{zero}} = \frac{1}{\sum_i w_i} \sum_{i \in \mathcal{N}} w_i |d_\theta(x_i)|. \quad (13)$$

This helps anchor the reconstructed surface and prevents drift away from the observed boundary.

Eikonal Loss: Enforces the Eikonal property $\|\nabla_x d(x)\|_2 = 1$ both near the surface and across the domain.

$$\mathcal{L}_{\text{eik-surf}} = \frac{1}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} \left(\|\nabla_x d_\theta(x_i)\|_2 - 1 \right)^2, \quad (14)$$

$$\mathcal{L}_{\text{eik-glob}} = \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} \left(\|\nabla_x d_\theta(\tilde{x}_i)\|_2 - 1 \right)^2, \quad (15)$$

By enforcing unit-gradient norms, the network is guided toward a valid signed distance representation rather than an arbitrary scalar field.

Normal Consistency Loss: Aligns predicted normals with ground-truth surface normals for stable geometry.

$$\mathcal{L}_{\text{normal}} = \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \left(1 - \langle \hat{n}_i, n_i^{\text{gt}} \rangle \right)^2, \quad (16)$$

where $\hat{n}_i = \frac{\nabla_x d_\theta(x_i)}{\|\nabla_x d_\theta(x_i)\|}$. This sharpens surface quality and improves reconstruction of fine details.

Sparse Regularization: Prevents the SDF from drifting by penalizing deviations from zero at randomly sampled points.

$$\mathcal{L}_{\text{sparse}} = \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} \exp(-\tau |d_\theta(\tilde{x}_i)|). \quad (17)$$

This discourages trivial solutions and improves stability when supervision is sparse.

Off-Surface Loss: Enforces correct distances for explicitly sampled off-surface points.

$$\mathcal{L}_{\text{off-surf}} = \frac{1}{|\mathcal{N}_{\text{off}}|} \sum_{i \in \mathcal{N}_{\text{off}}} \left(d_\theta(x_i) - d_i \right)^2. \quad (18)$$

This regularization ensures that the field encodes proper signed distance values even away from observed regions. Here, \mathcal{N}_{off} is a set of sampled points off the SDF surface.

RGB Reconstruction Loss: Supervises predicted colors against observed RGB values to ensure photometric consistency.

$$\mathcal{L}_{\text{rgb}} = \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} \|\mathbf{c}_\theta(x_i) - \mathbf{c}_i^{\text{gt}}\|_2^2. \quad (19)$$

This couples geometry with appearance, ensuring that the reconstructed shape also reproduces surface-level visual cues.

This structured formulation ensures geometric fidelity at the surface, enforces global signed distance properties, and aligns appearance with observations, enabling stable convergence and high-quality reconstructions. We empirically validate the importance of each component loss term in section V-D.

F. Surface Reconstruction

Once trained, the SDF network can be used to recover explicit 3D geometry from implicit predictions. Each input is a 3D coordinate vector $\mathbf{x} \in \mathbb{R}^3$, corresponding to the spatial position of a sampled point in the reconstruction volume. To extract the geometry, we evaluate the network on a dense uniform grid of 3D samples within the reconstruction bounds, thereby constructing a volumetric SDF field representation of the scene. The iso-surface corresponding to $d(\mathbf{x}) = 0$ is then extracted using the marching cubes [35] algorithm, which produces a watertight triangle mesh. This mesh can also be colored by evaluating the model’s predicted RGB values at the corresponding grid points, yielding a textured reconstruction.

For cleaner iso-surface extraction, we apply Gaussian smoothing to the raw SDF values in order to suppress high-frequency noise and improve mesh quality. Let \mathbf{x}_i denote a grid point with predicted signed distance value s_i . The smoothed value \tilde{s}_i is defined as:

$$\tilde{s}_i = \frac{\sum_{j \in \mathcal{N}_G(\mathbf{x}_i)} \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) \cdot s_j}{\sum_{j \in \mathcal{N}_G(\mathbf{x}_i)} \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)}, \quad (20)$$

where $\mathcal{N}_G(\mathbf{x}_i)$ is a local neighborhood around \mathbf{x}_i , and σ is the smoothing bandwidth.

This filtering process reduces artifacts from network prediction errors and enforces local smoothness in the reconstructed surface. By denoising the SDF prior to marching cubes, we obtain cleaner iso-surfaces with fewer spurious components and sharper, more visually coherent geometry.

G. Robot Surface Tracing

Implicit surfaces enable the construction of robot policies for *surface tracing*. This is a class of commonly found problems, where a robot must follow the geometry of an object or environment at a certain distance. Examples include robotic inspection (e.g., crack detection or defect scanning), automated surface treatment (painting, polishing, or cleaning), and quality assurance tasks. In such settings, the robot’s end-effector is required first to approach a target distance from the surface, and then to move tangentially while maintaining contact or a fixed standoff distance. Here, we take a reactive motion generation approach [36] and model surface tracing with a piecewise (mode-switched) velocity field over the learned

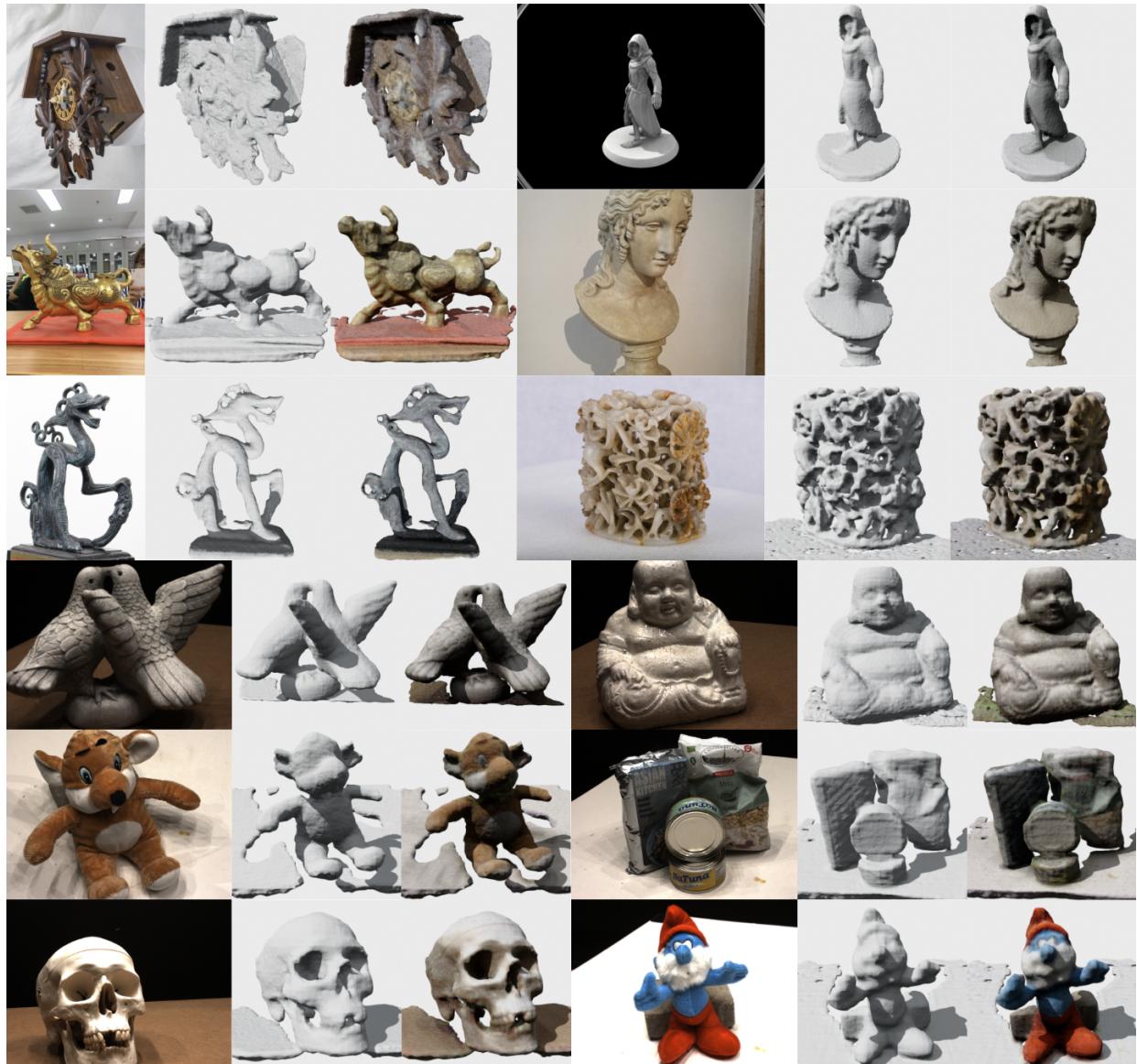


Fig. 3: Qualitative reconstruction results on marching-cube visualized implicit surfaces generated from a *single* input image from BlendedMVS [37] and DTU [38] datasets. We illustrate the input image, the resulting geometry without color, and that with color.

SDF $d(x)$, where $x \in \mathbb{R}^3$ is the end-effector, k is controller gain, d^* is the desired iso-value, $\hat{n}(x) = \nabla d(x)/\|\nabla d(x)\|$ is the surface normal, and $P_T(x) = I - \hat{n}(x)\hat{n}(x)^\top$ projects onto the tangent plane. Using a small tolerance $\varepsilon > 0$:

$$\dot{x} = \begin{cases} -k(d(x) - d^*)\nabla d(x), & \text{if } |d(x) - d^*| > \varepsilon \\ -kP_T(x)(x - x^*), & \text{if } |d(x) - d^*| \leq \varepsilon. \end{cases} \quad (21)$$

The first mode exponentially drives the end-effector toward the desired iso-surface; once within the band $|d - d^*| \leq \varepsilon$, the controller switches to tangential motion that approaches the target x^* while remaining on the same contour (since P_T removes the normal component).

V. EXPERIMENTS

A. Dataset and Metrics

We evaluate the proposed FINS framework on examples from the DTU [38] dataset and the BlendedMVS [37] dataset. The DTU dataset is one of the most widely used benchmarks

in multi-view reconstruction, providing high-quality desktop scenes with accurate ground-truth annotations. In contrast, the BlendedMVS dataset is designed for large-scale multi-view stereo. Using these two datasets, we first compare our method against several baselines, focusing on the quality of the reconstructed meshes when trained for the same number of iterations. We then perform an ablation study on the loss terms in our model to examine the contribution of each design choice. Additionally, we examine the applicability of learned representations for robot surface-tracing tasks.

All experiments are conducted on a single RTX 4060 Laptop GPU. For evaluation, we uniformly sample 200,000 vertices from both the reconstructed surface and the ground-truth surface. **Chamfer Distance (CD)** and **Normal Angle Error (NAE)** are employed as evaluation metrics to measure how closely the learned surface matches the ground truth. Specifically, given predicted surface points P and ground-truth

Method	# of Input Images	Training Time (s)	DTU [38]						BlendedMVS [37]			
			CD ↓			NAE (°) ↓			CD ↓		NAE (°) ↓	
			Smurf	Toy Tiger	Statue	Smurf	Toy Tiger	Statue	Sculpture	Bull	Sculpture	Bull
NeuS [9]	49	247	n/a	11.83	8.07	n/a	8.58	10.83	n/a	n/a	n/a	n/a
NeuS2 [10]	5	18	13.67	3.54	4.28	9.12	8.94	10.40	0.00890	0.0991	8.14	14.39
SparseNeuS [16]	2	127	16.10	5.57	8.18	9.90	9.01	10.39	0.0145	0.137	8.54	16.29
SparseCraft [17]	3	85	680.18	661.46	625.96	66.89	74.77	61.45	1.68	3.02	90.27	46.95
Ours	1	10	8.99	7.23	7.66	9.37	8.47	9.83	0.0198	0.0373	10.20	7.56

TABLE I: Comparison of **Chamfer Distance** ↓ and **Normal Angle Error** ↓ across DTU [38] and BlendedMVS [37] datasets.

Note: n/a denotes that the training did not converge. The experiments for SparseCraft are conducted on an NVIDIA A100 GPU due to the limited computational capability of the RTX 4060 Laptop GPU. The results for SparseNeuS are reported after 500 iterations of per-scene finetuning.

Variant	DTU [38]						BlendedMVS [37]			
	CD ↓			NAE (°) ↓			CD ↓		NAE (°) ↓	
	Smurf	Toy Tiger	Statue	Smurf	Toy Tiger	Statue	Sculpture	Bull	Sculpture	Bull
Full model	8.99	7.23	7.66	9.37	8.47	9.83	0.0198	0.0373	10.20	7.56
w/o \mathcal{L}_{SDF}	10.37	7.92	7.81	9.96	9.16	10.02	0.0138	0.0439	9.85	9.27
w/o $\mathcal{L}_{\text{zero}}$	12.94	14.26	8.58	9.64	8.99	9.87	0.0238	0.0761	10.29	9.13
w/o $\mathcal{L}_{\text{Leik}}$	5.77	5.94	4.67	9.02	8.72	10.22	0.0134	0.0400	10.90	8.75
w/o $\mathcal{L}_{\text{normal}}$	10.01	7.54	8.20	9.39	9.45	10.41	0.0192	0.0441	10.70	8.51
w/o $\mathcal{L}_{\text{sparse}}$	9.02	7.14	6.62	9.38	8.86	10.04	0.0209	0.0378	10.24	7.62
w/o $\mathcal{L}_{\text{off-surf}}$	8.22	6.56	5.83	9.32	9.42	10.14	0.0167	0.0368	8.82	7.80

TABLE II: Ablation study of different design choices in our method on DTU [38] and BlendedMVS [37] datasets, reported on multiple scans with **Chamfer Distance** ↓ and **Normal Angle Error** ↓.

points G , the CD is defined as:

$$\text{CD}(P, G) = \frac{1}{|P|} \sum_{p \in P} \min_{g \in G} \|p - g\|_2^2 + \frac{1}{|G|} \sum_{g \in G} \min_{p \in P} \|g - p\|_2^2, \quad (22)$$

and the NAE is defined as:

$$\text{NAE}(P, G) = \frac{1}{|P|} \sum_{p \in P} \min_{g \in G} \arccos \left(\frac{\langle n_p, n_g \rangle}{\|n_p\| \|n_g\|} \right), \quad (23)$$

where n_p and n_g denote the normal vectors at points p and g , respectively. The reported results are obtained by averaging over five independent runs. The specific data we use for quantitative evaluations are *Statue*, *Toy Tiger*, *Smurf* (labelled in DTU as Scans 114, 105, and 82 respectively); *Sculpture* and *Bull* are selected from the BlendedMVS dataset.

B. Evaluation Details

For examples from the DTU dataset, we adopt the reference meshes provided by the benchmark, specifically the Poisson surface reconstructions of the MVS point clouds generated by Tola et al. [39]. For the BlendedMVS dataset, we directly use the ground-truth meshes released by the authors.

Mesh Alignment. To ensure fair and consistent evaluation, all reconstructed meshes are aligned to the coordinate frame of the ground truth prior to metric computation. Without alignment, even small pose discrepancies can dominate Chamfer Distance and Normal Angle Error, obscuring the effect of reconstruction quality. Prior to alignment, large planar structures (e.g., desktops) that are present in both predicted and ground-truth meshes are manually removed using MeshLab, as they otherwise bias the correspondence search. Alignment is then performed in two stages:

Rough alignment: several pairs of corresponding landmarks are manually annotated between the reconstructed mesh and the ground truth. These correspondences provide a set of 3D

point matches, which are used to compute an initial similarity transform with the Umeyama method [40]. This step resolves large-scale differences in orientation, translation, and scale.

Fine alignment: to refine the initial transform, both meshes are uniformly sampled to generate dense point clouds. These point clouds are then aligned using point-to-point Iterative Closest Point (ICP) [41], which minimizes residual misalignment and brings the meshes into near-perfect correspondence.

Since the rough alignment depends on manually selected correspondences, perfect alignment cannot be guaranteed, and slight residual errors may affect the reported scores. Nevertheless, for each object case the same alignment transform is applied consistently across all methods, ensuring that the comparison remains fair and unbiased.

C. Object Surface Reconstruction

We next compare our approach against several baselines on object-level surface reconstruction. For consistency, we train all methods for 500 epochs. As summarized in Table I, FINS achieves competitive or superior reconstruction quality across many of the evaluated objects in the DTU and BlendedMVS datasets, while requiring dramatically fewer resources in terms of both input images and training time.

NeuS [9] relies on dense supervision with 49 input images and more than 240 seconds of optimization per scene. Even with this heavy supervision, its performance is inconsistent and in some cases fails to converge, making it impractical for robotic scenarios where rapid scene understanding is needed. NeuS2 [10] improves efficiency substantially, requiring only 5 input views and 18 seconds of training. It achieves the lowest Chamfer Distance on DTU’s *Toy Tiger* (3.54) and *Statue* (4.28) objects, demonstrating strong accuracy. However, the requirement of multiple input views remains restrictive when

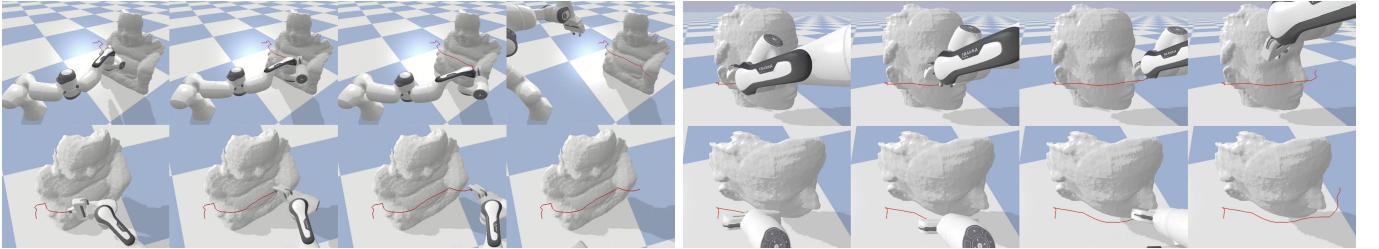


Fig. 4: The implicit distance representation produces accurate iso-surfaces, which enable robot surface-tracing motion generation. The robot’s motion can be generated by considering the normal and gradient vectors of iso-surfaces of the learned model, tracing the surface of reconstructions of the *Statue* and *Head* images. The red line denotes the Franka’s end effector path.

Variant	Time (s)	CD ↓		NAE ($^{\circ}$) ↓	
		Toy Tiger	Statue	Toy	Tiger
HE + Mixed 2nd Order	10	7.23	7.66	8.47	9.83
PE + 1st Order	219	7.89	7.57	9.14	10.55

TABLE III: Comparison between **Hash Encoding w/ 2nd-order optimizer (Ours)** and **Positional Encoding w/ 1st-order optimizer** on the DTU dataset, evaluated using **Chamfer Distance** ↓ and **Normal Angle Error** ↓.

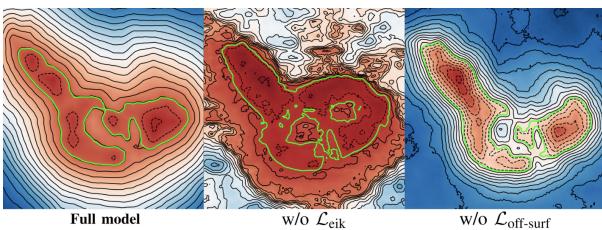


Fig. 5: Removing the Eikonal or Off-surface loss term can lead to a better surface reconstruction quality, which can lead to poor contours off the surface of interest. This observation is available in robotic deployment. SparseNeuS [16] further reduces the number of required inputs to 2 images, but finetuning times remain over 120 seconds per scene, and while its Normal Angle Error is competitive (e.g., 8.54° on *Sculpture*), its Chamfer Distance is significantly higher (e.g., 16.10 on *Smurf*). SparseCraft [17], despite leveraging an NVIDIA A100 GPU, produces divergent results: Chamfer Distances in the hundreds (e.g., 680.18 on *Smurf*) and Normal Angle Errors exceeding 45°, coupled with training times beyond 80 seconds.

In contrast, FINS reconstructs detailed geometry and a consistent signed distance field from only a *single RGB image*, converging in approximately 10 seconds on a consumer-grade RTX 4060 Laptop GPU. On DTU, our method achieves Chamfer Distances of 8.99 (*Smurf*), 7.23 (*Toy Tiger*), and 7.66 (*Statue*), along with Normal Angle Errors consistently around 7°–10°. On BlendedMVS, FINS attains strong results across both indoor and outdoor categories, with Normal Angle Errors as low as 7.56 (*Bull*), competitive with or better than all baselines. Although FINS does not always outperform NeuS2 on every metric, it consistently balances accuracy with extreme efficiency, reducing both input views (from 5–49 views to just 1) and training time (from 18–600+ seconds to only 10).

D. Ablation Study

We analyze the contribution of each loss and our optimization/encoding choices using single-image reconstruction

on DTU and BlendedMVS. All variants are trained for 500 iterations with identical data, sampling, and schedules.

Effect of loss terms: Table II reports Chamfer Distance (CD) and Normal Angle Error (NAE) for the full model and for variants with individual losses removed. The *full* configuration delivers consistently strong geometry across objects and datasets, indicating that the objectives are complementary. Interestingly, removing the Eikonal regularizer can sometimes reduce CD on DTU (e.g., *Smurf/Toy Tiger*), but overfits. This is shown in Fig. 5 where this overfitting comes at the expense of a valid signed-distance structure: gradients lose unit norm away from the surface, producing distorted level sets and degraded SDF quality. Similar trends appear when dropping the zero-level constraint or normal consistency: surfaces may remain visually plausible, yet normals become noisy and level sets drift, which is detrimental for downstream planning. Overall, enforcing (i) zero-crossing consistency, (ii) unit-norm gradients, and (iii) normal alignment stabilizes the field and preserves geometry in low-supervision regimes.

Encoding and optimizer: We further compare our multi-resolution hash encoding with mixed first/second-order training against a conventional positional encoding with a first-order optimizer (Table III). The hash+K-FAC variant attains comparable or better accuracy while reducing wall-clock time from minutes to ~10 s on a laptop GPU. The combination yields fast early progress via Lion on the shared encoder and curvature-aware refinement of the small geometry/color heads via K-FAC, improving convergence stability without incurring the cost of full second-order updates on the entire network.

Takeaways: The various loss terms, including the Eikonal, zero-level, and normal terms are jointly necessary to obtain high-quality SDFs, even when raw mesh metrics (CD) which measure surface quality momentarily improve without them. Hash encoding plus lightweight, head-only second-order updates offers a favorable accuracy-time trade-off, enabling practical single-image reconstruction efficiently.

E. Robot Surface Tracing

We validate the practical utility of the learned SDFs by applying them to a surface-tracing task in the PyBullet simulator [42] with a Franka Emika Panda arm. This task requires the end-effector to approach a desired standoff distance from the reconstructed surface and then follow the surface tangentially, as commonly needed in inspection, painting, or polishing.

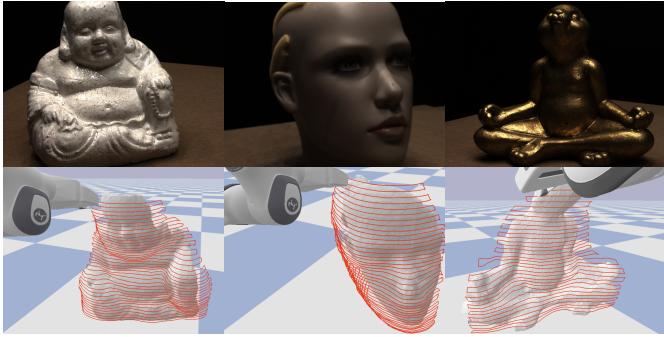


Fig. 6: FINS takes in single images (top), and produces implicit representations, which allow surface following. Lawn mower patterns tracing off the surface are shown in red (bottom).

The controller implemented is outlined in Section IV-G, using a piecewise velocity field defined over the learned SDF $d(x)$. In the *approach phase*, the end-effector is driven toward the target iso-value d^* by following the SDF gradient, ensuring exponential convergence to the desired contour. Once within a tolerance band, the controller switches to the *surface-following phase*, in which tangential motion is generated by projecting the velocity onto the local tangent plane. This guarantees that motion remains constrained to the iso-surface while progressing toward the goal. In simulation, we set $d^* = 0.05$ and commanded the robot to trace along the reconstructed object, by tracing hugging the surface at the specified offset, by following the iso-surface of our learned model. Additional examples of running lawn-mower patterns by considering the iso-surfaces of the learned implicit representations are shown in fig. 6. These experiments confirm that the fields produced by FINS are not only geometrically accurate at the surface but also suitable for real-time control tasks requiring gradient and iso-surface information.

VI. CONCLUSIONS

We propose Fast Image-to-Neural Surface (FINS), a framework that reconstructs a high-fidelity SDF field within a few seconds, given either a *single* image input. Our framework combines a multi-resolution hash grid encoder with light-weight geometry head and color head to accelerate convergence. We then leverage a hybrid optimization approach, where an approximate second-order Kronecker-factorized optimizer is used to speed up convergence and stability further. We empirically validate our method against a suite of methods against a suite of strong baselines on DTU and BlendedMVS, showing that FINS reaches competitive, and often superior, reconstruction quality while reducing both supervision (down to a single image) and wall-clock optimization time to ~ 10 s on consumer hardware.

REFERENCES

- [1] J. Ortiz, A. Clegg, J. Dong, E. Sucar, D. Novotny, M. Zollhoefer, and M. Mukadam, “isdf: Real-time neural signed distance fields for robot perception,” *arXiv preprint arXiv:2204.02296*, 2022.
- [2] W. Zhi, I. Akinola, K. van Wyk, N. Ratliff, and F. Ramos, “Global and reactive motion generation with geometric fabric command sequences,” in *IEEE International Conference on Robotics and Automation, ICRA*, 2023.
- [3] V. Vasilopoulos, S. Garg, P. Piacenza, J. Huh, and V. Isler, “Ramp: Hierarchical reactive motion planning for manipulation tasks using implicit signed distance functions,” in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 10551–10558.
- [4] Y. Li, X. Chi, A. Razmjoo, and S. Calinon, “Configuration space distance fields for manipulation planning,” *arXiv preprint arXiv:2406.01137*, 2024.
- [5] D. Driess, J.-S. Ha, M. Toussaint, and R. Tedrake, “Learning models as functionals of signed-distance fields for manipulation planning,” in *Conference on robot learning*. PMLR, 2022, pp. 245–255.
- [6] C. Quintero-Pena, W. Thomason, Z. Kingston, A. Kyriolidis, and L. E. Kavraki, “Stochastic implicit neural signed distance functions for safe motion planning under sensing uncertainty,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 2360–2367.
- [7] M. N. Finean, W. Merkt, and I. Havoutis, “Predicted composite signed-distance fields for real-time motion planning in dynamic environments,” in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 31, 2021, pp. 616–624.
- [8] S. T. Bukhari, D. Lawson, and A. H. Qureshi, “Differentiable composite neural signed distance fields for robot navigation in dynamic indoor environments,” *arXiv preprint arXiv:2502.02664*, 2025.
- [9] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, “Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction,” *arXiv preprint arXiv:2106.10689*, 2021.
- [10] Y. Wang, Q. Han, M. Habermann, K. Daniilidis, C. Theobalt, and L. Liu, “Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.
- [11] L. Yariv, J. Gu, Y. Kasten, and Y. Lipman, “Volume rendering of neural implicit surfaces,” in *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.
- [12] L. Yariv, Y. Kasten, D. Moran, M. Galun, M. Atzmon, B. Ronen, and Y. Lipman, “Multiview neural surface reconstruction by disentangling geometry and appearance,” *Advances in Neural Information Processing Systems*, vol. 33, 2020.
- [13] M. Oechsle, S. Peng, and A. Geiger, “Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction,” in *International Conference on Computer Vision (ICCV)*, 2021.
- [14] B. Xu, J. Hu, J. Li, and Y. He, “Gsurf: 3d reconstruction via signed distance fields with direct gaussian supervision,” *arXiv preprint*, 2024.
- [15] R. Peng, X. Gu, L. Tang, S. Shen, F. Yu, and R. Wang, “Gens: Generalizable neural surface reconstruction from multi-view images,” in *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*, 2023.
- [16] X. Long, C. Lin, P. Wang, T. Komura, and W. Wang, “Sparseneus: Fast generalizable neural surface reconstruction from sparse views,” in *European Conference on Computer Vision*. Springer, 2022, pp. 210–227.
- [17] M. Younes, A. Ouasfi, and A. Boukhayma, “Sparsecraft: Few-shot neural reconstruction through stereopsis guided geometric linearization,” *arXiv preprint*, 2024.
- [18] Z. Gao, J.-W. Bian, G. Lin, H. Chen, and C. Shen, “SurfaceSplat: Connecting Surface Reconstruction and Gaussian Splatting,” *arXiv e-prints*, 2025.
- [19] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud, “Dust3r: Geometric 3d vision made easy,” in *CVPR*, 2024.
- [20] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny, “Vggt: Visual geometry grounded transformer,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.
- [21] T. Müller, A. Evans, C. Schied, and A. Keller, “Instant neural graphics primitives with a multiresolution hash encoding,” *ACM Trans. Graph.*, 2022.
- [22] W. Zhi, H. Tang, T. Zhang, and M. Johnson-Roberson, “Teaching periodic stable robot motion generation via sketch,” *IEEE Robotics and Automation Letters*, 2025.

- [23] R. Senanayake and F. Ramos, “Bayesian hilbert maps for dynamic continuous occupancy mapping,” in *Conference on Robot Learning (CoRL)*, 2017.
- [24] W. Zhi, L. Ott, R. Senanayake, and F. Ramos, “Continuous occupancy map fusion with fast bayesian hilbert maps,” in *International Conference on Robotics and Automation (ICRA)*, 2019.
- [25] S. Vasudevan, S. Sagar, and R. Senanayake, “Strategic vantage selection for learning viewpoint-agnostic manipulation policies,” *arXiv preprint arXiv:2506.12261*, 2025.
- [26] H. Wright, W. Zhi, M. Johnson-Roberson, and T. Hermans, “V-prism: Probabilistic mapping of unknown tabletop scenes,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.
- [27] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “Nerf: Representing scenes as neural radiance fields for view synthesis,” in *ECCV*, 2020.
- [28] W. Zhi, Z. Ma, T. Zhang, and M. Johnson-Roberson, “From single images to motion policies via video-generation environment representations,” *arXiv preprint arXiv:2505.19306*, 2025.
- [29] J. Zhang, C. Herrmann, J. Hur, V. Jampani, T. Darrell, F. Cole, D. Sun, and M.-H. Yang, “Monst3r: A simple approach for estimating geometry in the presence of motion,” *arXiv preprint arXiv:2410.03825*, 2024.
- [30] W. Zhi, H. Tang, T. Zhang, and M. Johnson-Roberson, “Unifying representation and calibration with 3d foundation models,” *IEEE Robotics and Automation Letters*, 2024.
- [31] T. Zhang, W. Zhi, B. Meyers, N. Durrant, K. Huang, J. Mangelson, C. Barbalata, and M. Johnson-Roberson, “Recgs: Removing water caustic with recurrent gaussian splatting,” *IEEE Robotics and Automation Letters*, 2025.
- [32] W. Zhi, H. Tang, T. Zhang, and M. Johnson-Roberson, “Simultaneous geometry and pose estimation of held objects via 3d foundation models,” *IEEE Robotics and Automation Letters*, 2024.
- [33] X. Chen, C. Liang, D. Huang, E. Real, K. Wang, Y. Liu, H. Pham, X. Dong, T. Luong, C.-J. Hsieh, Y. Lu, and Q. V. Le, “Symbolic discovery of optimization algorithms,” 2023.
- [34] J. Martens and R. Grosse, “Optimizing neural networks with kronecker-factored approximate curvature,” in *Proceedings of the 32nd International Conference on Machine Learning*, 2015.
- [35] W. E. Lorensen and H. E. Cline, *Marching cubes: a high resolution 3D surface construction algorithm*. Association for Computing Machinery, 1998.
- [36] W. Zhi, T. Lai, L. Ott, and F. Ramos, “Diffeomorphic transforms for generalised imitation learning,” in *Learning for Dynamics and Control Conference, L4DC*, 2022.
- [37] Y. Yao, Z. Luo, S. Li, J. Zhang, Y. Ren, L. Zhou, T. Fang, and L. Quan, “Blendedmvs: A large-scale dataset for generalized multi-view stereo networks,” *Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [38] R. Jensen, A. Dahl, G. Vogiatzis, E. Tola, and H. Aanaes, “Large scale multi-view stereopsis evaluation,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.
- [39] E. Tola, C. Strecha, and P. Fua, “Efficient large-scale multi-view stereo for ultra high-resolution image sets,” *Machine Vision and Applications*, vol. 23, pp. 903–920, 2012.
- [40] S. Umeyama, “Least-squares estimation of transformation parameters between two point patterns,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1991.
- [41] P. Besl and N. D. McKay, “A method for registration of 3-d shapes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1992.
- [42] Coumans, Erwin and Bai, Yunfei, “Pybullet, a free and open source physics simulator for robotics, games and machine learning,” <http://pybullet.org>, 2016.