# Next Best Sense: Guiding Vision and Touch with FisherRF for 3D Gaussian Splatting

Matthew Strong*[1], Boshu Lei*[3], Aiden Swann[2], Wen Jiang[3], Kostas Daniilidis[3], Monroe Kennedy III[2]

***Abstract*— We propose a framework for active next best view and touch selection for robotic manipulators using 3D Gaussian Splatting (3DGS). 3DGS is emerging as a useful explicit 3D scene representation for robotics, as it has the ability to represent scenes in a both photorealistic and geometrically accurate manner. However, in real-world, online robotic scenes where the number of views is limited given efficiency requirements, random view selection for 3DGS becomes impractical as views are often overlapping and redundant. We address this issue by proposing an end-to-end online training and active view selection pipeline, which enhances the performance of 3DGS in few-view robotics settings. We first elevate the performance of few-shot 3DGS with a novel *semantic depth alignment* method using Segment Anything Model 2 (SAM2) that we supplement with Pearson depth and surface normal loss to improve color and depth reconstruction of real-world scenes. We then extend FisherRF, a next-best-view selection method for 3DGS, to select views and touch poses based on depth uncertainty. We perform online view selection on a real robot system during live 3DGS training. We motivate our improvements to few-shot GS scenes, and extend depth-based FisherRF to them, where we demonstrate both qualitative and quantitative improvements on challenging robot scenes. For more information, please see our project page at armlabstanford.github.io/next-best-sense.**

## I. INTRODUCTION

As robots become more capable of performing manipulation in daily tasks, it is important for them to efficiently model the scene and relevant objects [1], [2]. One prominent explicit 3D representation is 3D Gaussian Splatting (3DGS) [3], which reconstructs a 3D scene from RGB images and camera poses. Traditionally, the creation of 3DGS scenes is done with a). many views and b). complete human supervision. 3DGS is well-known for requiring tens of views and near-perfect camera poses; as it is prone to overfitting on sparse input views. This approach, however, is inconsistent with many of the potential robotic applications of 3DGS. In a robotic setting, every additional view is costly, requiring movement and potentially limited onboard compute. It is often inefficient to collect hundreds of views required for training a 3DGS model in robotic environments. For future applications of 3DGS on robotics systems, robust few-view training methods are needed. Even the assistance of depth priors for regularization in prior work [4]–[7] often relies on high quality visual data for constructing a precise 3D scene.

This research was supported by NSF Graduate Research Fellowship No. DGE-2146755 and NSF Grant No. 2142773, 2220867.

[·][1]Department of Computer Science, [·][2]Department of Mechanical Engineering, Stanford University, Stanford CA, USA. Emails: {mastro1, swann, monroek}@stanford.edu. [·][3]School of Engineering and Applied Science, University of Pennsylvania, Philadelphia PA, USA. Emails: {leiboshu, wenjiang, kostas}@seas.upenn.edu.

*Both authors contributed equally to this work.

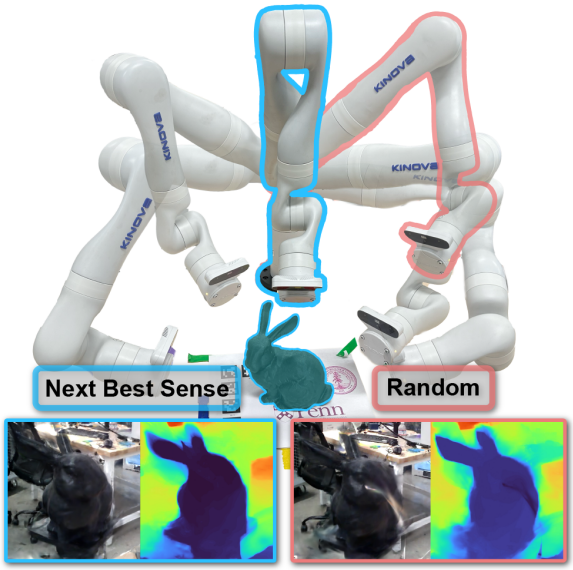


Fig. 1: Our method outperforms random view selection for few view 3DGS scenes. We show a series of robot views, with the next view proposed by our method compared to random. Our method maximizes the Fisher information gain for each view over both vision and touch inputs.

To address this, we propose the use of *semantic depth alignment* to enhance the impact of each additional view. We draw from the power of state-of-the-art monocular depth estimation models, and with an off-the-shelf depth camera and color image, we perform a semantic depth alignment by aligning objects and the background of an image with Segment Anything Model 2 (SAM2) [8]. This is used as a form of initialization in our few-shot scene. During training, depth supervision is enforced with a relaxed relative loss, which mitigates the scale ambiguities between the true depth and estimated depth from the depth estimation models. Gentle normal supervision and camera optimization guide the Gaussian Splat to reconstruct a visually accurate scene.

In addition to maximizing the impact of each of the limited training views, we propose a method to optimally select the *next best view* based on depth uncertainty and optionally color uncertainty. When there are a limited number of views available, uncertainty emerges from depth discontinuities, occluded regions, and sources of aleatoric error. It is critical to understand where *next* best views and touches should be made in a scene to reduce this uncertainty and construct precise representations for effective robot operation.

In order to determine the optimal next view in a radiance field, we develop a uncertainty metric and maximize

uncertainty gain. Past work in uncertainty quantification in radiance fields includes per-pixel uncertainty, ray entropy, ensemble methods, and interpreting the radiance field as a probability distribution [9]–[12]. These works require significant modifications to the original radiance field architecture, become computationally infeasible in the case of ensemble-based uncertainty, and make assumptions about ray density distributions that are not generalizable. FisherRF [13], a recent state-of-the-art work in uncertainty quantification, uses the **Fisher Information** to quantify observed information without any ground truth data. Only requiring one backward pass, FisherRF is broadly generalizable to explicit radiance fields like Gaussian Splatting, making it a suitable method for robotics applications.

FisherRF computes the expected information gain on candidate views of 3DGS to then select the next best view. Unlike FisherRF, recent research on NBV in offline and online applications [10]–[12] leverages prior methods for uncertainty quantification and is only applied to implicit radiance fields methods like Neural Radiance Fields (NeRFs).

We extend FisherRF to select views *and* touches based on **depth uncertainty**. We posit that the geometry and visual quality of a scene can be guided through depth, as we observe that regions of erroneous color in Gaussian Splats are often associated with ambiguous depth. This framework can be easily extended to touch, which has been shown to improve 3DGS quality [14].

The integration of touch into 3DGS is still limited, however – only Touch-GS [14] and Snap-It, Tap-It, Splat-It [15] address touch-informed 3DGS, as compared to several works fusing vision and touch for manipulation [16]–[21]. While Touch-GS demonstrates the improvements on 3DGS scenes with touch, it suffers from two shortcomings: 1. it requires a human to move the robot to desired vision and touch poses and 2. requires the order of 100-500 touches to construct a visually and geometrically sufficient scene. Snap-It, Tap-It, Splat-It similarly requires manual views. Our method automonously selects views and touches guided by uncertainty.

Altogether, we construct a framework that enables for the autonomous sense selection with a robotic manipulator, which we call *Next Best Sense*.

**Key Contributions.** We propose *Next Best Sense*, the first approach that enables uncertainty-guided view *and* touch selection for training a 3DGS for robotic manipulators. Our main contributions in Next Best Sense are as follows: **1)** We propose an improved method in few-shot 3DGS scenes with a novel depth alignment method using Segment Anything Model 2. **2)** We present a novel extension of FisherRF by leveraging depth uncertainty to assist in view and touch selection. **3)** We present a closed-loop framework coupling both perception and action for training few-view scenes for robotic manipulators with FisherRF. To the best of our knowledge, this is the first work that enables next best view planning for Gaussian Splatting for robot manipulators in real-time. **4)** We extend our method to uncertainty-guided touch, demonstrating qualitative improvements with only 10 touches.

The remainder of the paper is outlined as follows. Section II covers mathematical preliminaries, and Section III introduces our method. Section IV showcases our results, and finally, Section V discusses these results and future avenues of research.

## II. PRELIMINARIES

### A. 3D Gaussian Splatting

3D Gaussian Splatting is an explicit 3D representation that represents a scene with 3D Gaussians. Each Gaussian is parameterized by a mean position $\mu \in \mathbb{R}^3$ and covariance $\Sigma \in \mathbb{R}^{3\times3}$. The covariance is computed as the product of a diagonal scale matrix $S \in \mathbb{R}^{3\times3}$ and rotation matrix $R \in \mathbb{R}^{3\times3}$ as $\Sigma = RSS^TR^T$. Each Gaussian also contains an opacity value $\alpha \in \mathbb{R}$ and spherical harmonic parameters to compute color. The color and depth of each pixel in a rendered image can be computed as blending ordered points along a camera ray, which with camera origin $\mathbf{o}$ and orientation $\mathbf{d}$ is defined as: $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$. The color $C(\mathbf{r})$ and depth $D(\mathbf{r})$ is computed by blending ordered points intersecting the ray:

$$\hat{C} = \sum_{i\in N} c_i\alpha_i T_i, \quad \hat{D} = \sum_{i\in N} d_i\alpha_i T_i, \tag{1}$$

where $T_i = \prod_{j=1}^{i-1}(1-\alpha_j)$, which is the transmittance, concretely defined as multiplying the previous Gaussian opacity values intersecting the ray. During training, the parameters of each Gaussian are optimized with gradient descent to minimize the following photometric loss betweeen a ground truth and rendered image: as $\mathcal{L} = (1-\lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{SSIM}}$, where $\mathcal{L}_1$ is the absolute loss between the RGB values of the rendered and ground truth image, and $\mathcal{L}_{\text{SSIM}}$ is the **S**tructural **S**imilarity **I**ndex **M**easure (SSIM) loss. Further, we can extend the loss to include $\mathcal{L}_{depth}$, which uses the ground truth and rendered depth image.

### B. Next Best View Selection

We use FisherRF [13] for next best view selection. The problem of selecting the next best view is formulated as maximizing the Fisher information gain between candidate RGB camera poses $x_i^{\text{acq}}$ and views $y_i^{\text{acq}}$ and captured training RGB views $D^{\text{train}}$, given Gaussians' parameters $w^*$.

$$\begin{aligned}&\mathcal{I}[w; \{y_i^{\text{acq}}\}|\{x_i^{\text{acq}}\}, D^{\text{train}}]\\ &= H[w^*|D^{\text{train}}] - H[w^*|\{y_i^{\text{acq}}\}, \{x_i^{\text{acq}}\}, D^{\text{train}}]\end{aligned} \tag{2}$$

Here, $H[\cdot]$ is the entropy. From [22], the next best view objective is to maximize the reduction in entropy. In [23], the entropy of the model parameters can be approximated using a second order expansion.

$$H[\omega^*] = -\frac{1}{2}\log det\, H''[w^*] \tag{3}$$

Using the inequality $\log\det(A + Id) < tr(A)$, the final objective function for the next best view is [23]:

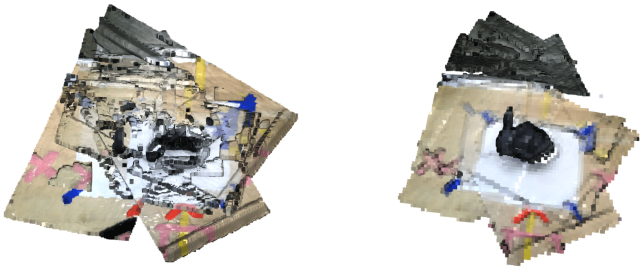

Fig. 2: Mesh of Incorporating Lifted Depths (left) and Lifted **SAM2** Depths (right). A semantic alignment provides a robust initialization for 3DGS.

$$\underset{x_i^{\text{acq}}}{\text{argmax}}\, \text{tr}(\mathbf{H}''[y_i^{\text{acq}}|x_i^{\text{acq}}, w^*]\mathbf{H}''[w^*|D^{\text{train}}]^{-1}) \tag{4}$$

The Hessian matrix $\mathbf{H}''$ can be computed just from the Jacobian matrix, which only requires a single backward pass on a given view. In practice, we apply Laplace approximation that approximates the Hessian matrix with its diagonal values plus a prior regularizer [13], [24].

$$\begin{aligned}\mathbf{H}''[y|x, w^*] &= \nabla_w f(x; w^*)^T \nabla_w f(x; w^*) \\ &\approx \text{diag}(\nabla_w f(x; w^*)^T \nabla_w f(x; w^*)) + \lambda I\end{aligned} \tag{5}$$

We present our extension to FisherRF for depth later in the methods section.

## III. METHOD

We now present *Next Best Sense*. The method is divided into three components: 1. **Few Shot Gaussian Splatting**, 2. **Next Best View for Robotics**, and 3. **Next Best Touch**. We leverage the real-time rendering capabilities of 3DGS to propose new views and touches in real-time.

### A. Few-Shot Gaussian Splatting

In environments where robots are constrained to efficiently utilize a limited number of views, traditional Gaussian Splatting training collapses, suffering from poor reconstruction and overfitting. Specifically, 3DGS is known to be highly sensitive to initialization in the few-view case, requiring extra care to align the initialization points. We improve the performance of few-shot Gaussian Splatting by optimizing the manner in which depth is handled during initialization and training.

*1) SAM2 Depth Alignment:* We posit that depth alignment should be done *semantically*, and supervision should be done *relatively*. When aligning a depth image and camera image, the depth of objects and components of a scene is based on the fact that the objects are semantically different. Concretely, we run the Segment Anything Model 2 automatic mask generator, which outputs a list of masks $M$ in image $I$. The alignment can be observed in Fig. 3. With an metric depth map from a depth sensor $D_{\text{real}}$, corresponding image $I$, monocular depth $D_{\text{MDE}}$, and list of masks $M$, we perform a mask-aware depth alignment as follows:

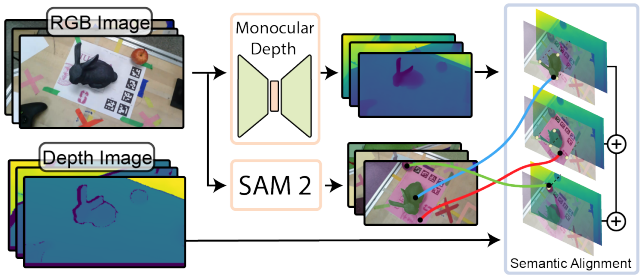


Fig. 3: **SAM2 Alignment.** Given an RGB image and depth image, we provide the RGB as input to a monocular depth model to get relative depths, and run the SAM2 automatic mask generator to get object and scene masks. We then align each object in the monocular depth with the corresponding sensor depth.

$$s^*_{M_i}, t^*_{M_i} = \arg\min_{s,t} \sum_{p \in D_{\text{real}}} ||D_{\text{real},M_i}(p) - D_{\text{MDE},M_i}(p; s, t)||^2 \tag{6}$$

where $s^*_{M_i}$ is the scale factor on the pixels in mask $M_i$, $t^*_{M_i}$ is the offset, and $p$ is a depth keypoint from sensor depth image $D_{\text{real}}$. An example of the few shot depths back-projected into 3D and into a mesh can be found in Fig. 2. Finally, the user is asked to use SAM2's point prompt to mark the selected object in a single frame. We then propagate this to all other frames, which computes a mask. We add this mask to our list of masks to ensure object alignment. This method maintains the metric accuracy of a depth sensor but the geometric quality of monocular depth models.

We lift the SAM2 aligned monocular depths into 3D and take a random percentage of them to initialize 3DGS, where a Gaussian of random color is constructed from each depth pixel. We find that this guides the few views to learn smoother geometries in the scene as compared to initialization from sensor depth data.

*2) Geometric Depth Guidance:* Drawing from prior works in few-shot GS, we introduce *Pearson Relative Depth Loss* to gently guide depth in our scene. We use the monocular depth output from a model such as DepthAnythingV2 [25] or Metric3DV2 [26] and measure the distribution difference between rendered and monocular depth maps.

We also incorporate scale regularization in our work, detailed in PhysGaussian [27], to reduce highly anisotropic Gaussians in our scene. Normal regularization is also lightly used, in which we use work from DN-Splatter [28] to derive normals, where we can approximate the surface normals from the monocular depth map, and during training, we minimize one of the scaling coefficients during training to encourage the Gaussians to become a flat like disc. Finally, we apply camera optimization on our views by adjusting the translation and rotation in $SO(3) \times \mathbb{R}^3$ for each view.

*3) Learning a 3D SAM2 Object Mask:* We render a separate image from GS for our object mask, which we can then perform binary cross entropy with the SAM2 Mask. This allows us to compute the 3D Gaussians that are associated

with the object, which we propose to use for touch.

### B. Next Best View for Robotics

Real world robot environments are visually feature-rich and full of color. FisherRF will select views that have a high Hessian with respect to the rendered color image, meaning that candidate views with high gradients of color are more likely to be selected. In the case of color-rich, opaque surfaces, FisherRF will select new views around these areas simply because of the high change in color. We assert that FisherRF should be extended to encode **depth uncertainty**, which is a strong indicator of geometric accuracy, and may be an even stronger metric of uncertainty as compared to color uncertainty. To achieve this, we formulate a **depth-based** negative log-likelihood objective for the radiance field similar to [13] as follows:

$$-\log p(y_D|x,\mathbf{w}) = (y_D - f_D(x,\mathbf{w}))^T(y_D - f_D(x,\mathbf{w})) \tag{7}$$

where $y_D$ is the ground truth depth, and $f_D$ is the depth rasterizer that outputs depth given camera pose $x$ and Gaussian Splat parameters $\mathbf{w}$. As this objective simply replaces color terms with depth terms in FisherRF, we follow the derivation in [13] and formulate the depth information gain objective as

$$\underset{x_i^{\text{acq}}}{\text{argmax}}\ \text{tr}(\mathbf{H}''[y_{D,i}^{\text{acq}}|x_i^{\text{acq}}, w^*]\mathbf{H}''[w^*|D^{\text{train}}]^{-1}), \tag{8}$$

where $y_{D,i}^{\text{acq}}$ is the depth. However, the Hessian $\mathbf{H}''$ only depends on the model input and parameters, so it can be reduced to the Jacobian as $\text{diag}(\nabla_w f_D(x;w^*)^T \nabla_w f_D(x;w^*)) + \lambda I$, where we compute the gradient of the *depth rasterization* for a given view. We then define the information gain as a linear combination of the color and depth information gain:

$$\mathcal{I}[w] = \alpha\mathcal{I}[w;f_C] + \beta\mathcal{I}[w;f_D], \tag{9}$$

where $\alpha$ weights the color information gain, and $\beta$ weights the depth information gain, and $f_C$ and $f_D$ are the color and depth rasterizers, respectively. This addition is computationally efficient as it requires one more backward pass on depth per view. Our formulation allows to encode the uncertainty as how much depth changes in a given view; discontinuous depths in and floaters are more likely to be removed.

*1) Robotic Next Best View:* With the formulation of FisherRF, we perform our robotic procedure as follows.

**Collect Random Views.** First, $n$ ($n < 9$) random views are collected by sampling poses in a sphere of radius uniformly sampled from $r_{min} < r < r_{max}$, where the center camera ray points directly at the object center. For each view, we compute the monocular depth, SAM2 aligned monocular depth, and store these with the camera pose, color image, and SAM2 mask.

**View Selection During Training**: During training, the Gaussian Splat queries the robot for a list of feasible candidate views; we then run FisherRF to compute the next best view. The robot moves to the highest view score. If any errors are encountered in inverse kinematics or trajectory planning, we select the next highest scoring view.

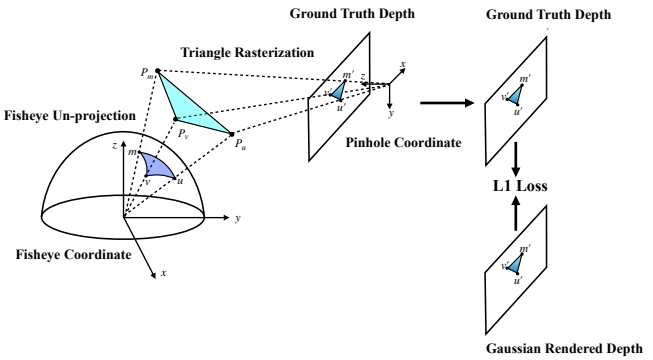


Fig. 4: **Tactile Data Supervision**. We backproject points from a fisheye coordinate to form a triangle face $P_u, P_v, P_m$, which is projected onto image plane and rasterizes the corresponding area $u', v', m'$ for touch depth supervision.

**Add New View**: We then add the view, monocular depth, SAM2 aligned depth, camera pose, color image, and SAM2 object mask. We store the point prompt on the first view and propagate it to SAM2 as new views are added.

### C. Next Best Touch

Touch provides accurate local geometries of a surface, regardless of lightning condition. It is then favorable to integrate it into our work as a tool to *refine* a scene. To do this, we propose to use FisherRF depth after our vision phase to propose new touches for GS.

We select touches similarly to vision. We compute the next best touch by computing the expected information gain of a pseudo touch camera *only on depth*. This ensures that more spikey and hole-like areas along an object are selected. We first only compute the depth uncertainty of Gaussians from the SAM2 object mask, ensuring spurious Gaussians not part of the object are ignored. As we add touches, we add them to the list of training views with different intrinsics to represent the touch camera. However, the touch poses are often too close to the surface of the object, which we address by moving the candidate touch view away from the object along its z axis till a surface of non-zero depth can be rendered.

We then collect a tactile image $I_t$ and its pose $p_i$. We first use a pretrained DenseNet model [29] to predict the depth along the ray, denoted as $I_d$. For each ray $\boldsymbol{t}_i$ from the fisheye camera, we unproject pixels into the point cloud point $\boldsymbol{p}_i = d \cdot \boldsymbol{t}_i$. We filter the points by their radial distance from the center and then by their z coordinate. Directly projecting the point cloud points to camera frame will lead to a sparse depth map. We convert the point cloud to triangle mesh using the connectivity from tactile image, and then rasterize the triangles into the image plane. The depth image is later used as depth supervision for Gaussian training, where we initialize Gaussians at the point of touch. We adopt $\mathcal{L}_1$ depth loss with smoothing around the point of touch, and the whole process is shown in Fig. 4.

| Method | PSNR↑ | SSIM↑ | LPIPS↓ |
|---|---|---|---|
| 3DGS w/o Depth | 15.66 | 0.49 | 0.53 |
| Dense-Depth [4] | 18.82 | 0.55 | 0.44 |
| Pearson Loss Depth | 18.32 | 0.54 | 0.45 |
| Lifted Depth, MSE Loss | 19.82 | 0.58 | **0.36** |
| **Lifted SAM2 Depth, Pearson Loss** | **20.21** | **0.65** | **0.43** |
| **Lifted SAM2 Depth, MSE Loss** | **20.33** | **0.67** | **0.41** |

TABLE I: Blender few-shot scene ablations with 3DGS

| Method | D-ABS↓ | D-ABS-O↓ |
|---|---|---|
| Dense-Depth [4] | 0.27 | 0.20 |
| Lifted Depth, MSE Loss | 0.18 | 0.035 |
| **Lifted SAM2 Depth, MSE Loss** | **0.16** | **0.033** |

TABLE II: **Blender simulated few-shot** ground truth depth comparison. D-ABS is the error across the whole view; D-ABS-O is the error for only the bunny.

## IV. RESULTS

### A. Few Shot Gaussian Splatting for Robotics

*1) Stanford Bunny Blender:* We demonstrate our improvements to Gaussian Splatting on the Touch-GS [14] dataset to highlight the improvements to vision for robotic tasks. To start, we perform ablations starting from vanilla 3DGS. We use the camera poses from Blender and use the perturbed ground truth depths with noise that quadratically increases with distance to simulate real depth. We use Nerfstudio [30] and implement depth supervision, normal supervision, and initialization of Gaussians from depth maps. We train on only 6 views for 15000 steps as convergence is reached early. We test PSNR, SSIM, and LPIPs for 34 unseen novel views on 3 trials and report the mean, and additionally the mean absolute *error for ground truth depth*.

As seen in Table I, depth supervision guides the geometric and visual quality of 3DGS; however, it still suffers from sparsity. Lifting the aligned Gaussians improves the density of the scene but is not as geometrically accurate as using SAM2 aligned depths for initialization. Across all visual metrics, SAM2 aligned depths improve the visual quality across test scenes besides LPIPS (shown as the green value). Further, the mean absolute error in depth in the SAM2 aligned depths is the best, highlighting its usefulness as a depth prior.

*2) Stanford Bunny Real and Challenge Objects:* With a 3D printed model of the Stanford bunny, we highlight our improvements in a real scene with low resolution visual and depth data. In Table III, metric depth loss on aligned depths becomes infeasible, and our method outperforms current baselines. We further include camera optimization, which improves the quality of the scene as the camera pose of the robot is not optimal. Finally, we test our method on a challenging mirror and prism, with normal supervision and camera optimization turned on as shown in Table IV. We refer interested readers to our website for more information.

### B. Offline FisherRF

We highlight the improvement of **FisherRF for depth** to our two bunny scenes, reporting the mean PSNR, SSIM, and LPIPS from five trials. We train GS for 15000 steps and select a new view every 2000 steps according to FisherRF, turning camera optimization off to see the direct effect of an added view and its pose. We verify on random, FisherRF depth, RGB (baseline), and a combination of the two with our formulation in Eq. 9. Selecting random views performs the worst, and while the baseline FisherRF on color is a significant improvement over random views, it does not select views that improve the geometric construction, as seen in Fig 5, where the geometries to the right of the back chair are incorrect. In fact, the head of the bunny is the sharpest in FisherRF depth. We posit that FisherRF on depth additionally receives the benefits of FisherRF color – we notice that very different colored Gaussians, which are higher areas of color uncertainty, often are associated with ambiguous depths. However, the converse is not necessarily true, which provides additional benefit to the scene. The difference is smaller between the two in the real world; however, we choose to use FisherRF on depth for the following experiments.

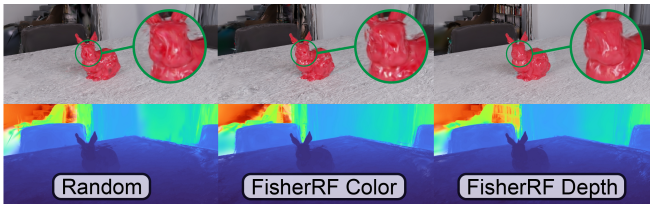


Fig. 5: **FisherRF Ablations** Qualitative results of Random (Left), FisherRF Color (Middle) and FisherRF Depth (Right).

| Method | PSNR↑ | SSIM↑ | LPIPS↓ |
|---|---|---|---|
| 3DGS | 10.63 | 0.43 | 0.58 |
| Dense Depth | 11.25 | 0.44 | 0.56 |
| Pearson Loss Depth | 11.68 | 0.47 | 0.54 |
| Lifted Aligned Depth Pearson | 11.79 | 0.47 | 0.53 |
| SAM2 Lifted Aligned Depth Pearson | 11.84 | 0.47 | 0.54 |
| Ours with Normal Supervision | 11.86 | 0.47 | 0.54 |
| **Ours with Camera Optimization** | **12.30** | **0.50** | **0.54** |

TABLE III: Few-Shot Bunny Real Results

### C. Robotic FisherRF

We apply our method to a real-world robotics scenario on challenging objects – the Stanford bunny, a toy Toyota Corolla, mirror, prism and 3D-printed ridged object. We use the Kinova Gen3 robot, and create an in-house pipeline in Docker using ROS, which is interfaced directly with Nerfstudio, allowing us to train a Splat and add views in real-time.

We start with only 8 random views. For each object, we keep the starting views the same for each method, and the list of candidate views every 2000 steps is kept the same (10 kinematically feasible poses). We train to 21000 steps. All experiments are run in real-time. We test each Splat on a held-out set per object of 25-40 views. The result is summarized in Table VI, with qualitative results in Fig 6. For all the objects, FisherRF performs better than the random baseline. We note that FisherRF on depth performs the best for the real world bunny – better than our baseline

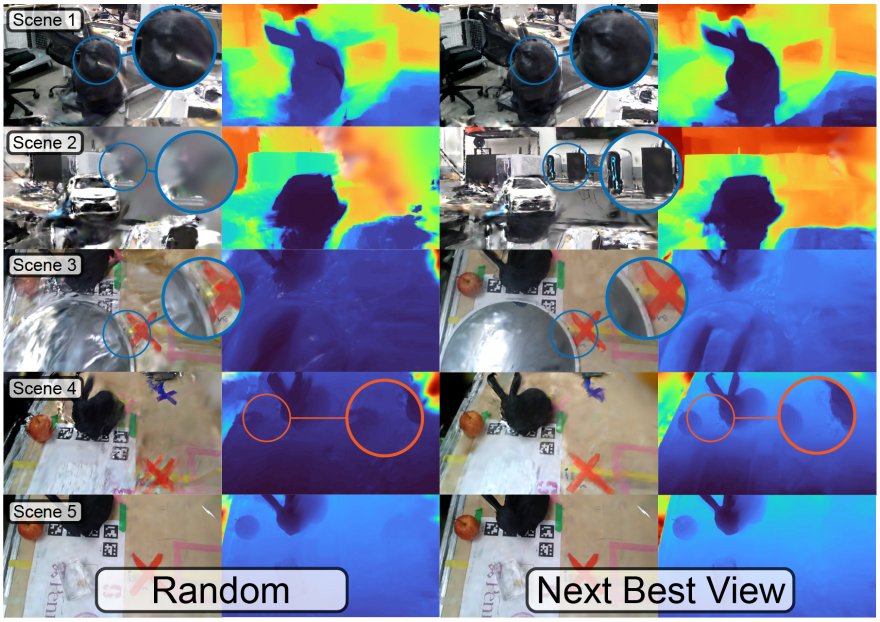


Fig. 6: **Qualitative Results of Next Best View**

| Object | Method | PSNR↑ | SSIM↑ | LPIPS↓ |
|---|---|---|---|---|
| Mirror | Pearson-Depth | 10.71 | 0.38 | 0.61 |
| | **Our Method** | **10.74** | **0.40** | **0.59** |
| Prism | Dense-Depth | 12.15 | **0.48** | 0.55 |
| | **Our Method** | **12.24** | **0.48** | **0.53** |

TABLE IV: Few-Shot GS on Challenge Objects

| Object | Method | PSNR↑ | SSIM↑ | LPIPS↓ |
|---|---|---|---|---|
| Bunny Blender | Random | 23.68 | 0.68 | 0.24 |
| | FisherRF RGB | 24.41 | 0.68 | 0.22 |
| | **FisherRF Depth** | **24.63** | **0.69** | **0.21** |
| | **FisherRF Combined** | **24.73** | **0.69** | **0.22** |
| Bunny Real | Random | 12.20 | 0.47 | 0.49 |
| | FisherRF RGB | 12.62 | 0.50 | 0.45 |
| | **FisherRF Depth** | **12.67** | **0.50** | **0.45** |
| | **FisherRF Combined** | **12.67** | **0.50** | **0.45** |

TABLE V: Offline FisherRF

of FisherRF on color – which comes from FisherRF depth encouraging new views to a robot to explore the *background* more, which originally is highly discontinuous. We see this in the toy car as well. On the mirror, prism, and ridged objects, FisherRF performs a depth smoothing effect by suggesting new views that learn the geometry of a scene more precisely, and visually generalizes to new views. We find that real world scenes are color rich but objects are geometrically smooth. Additionally, our few-shot method is showcased, as the scene remains visually and geometrically crisp.

### *D. FisherRF and Touch Data Supervision*

As a final experiment, we verify the usefulness of FisherRF for single **touch supervision** on a mirror example. We leverage a pretrained Splat from the Touch-GS mirror and perform FisherRF depth on a list of candidate touches every 100 steps until we have added 10 touches. The result is shown in Fig 7, where more touches are selected in the hole-like areas in the mirrors, which is due to the the holes having a higher Hessian with respect to depth.

| Object | Method | PSNR↑ | SSIM↑ | LPIPS↓ |
|---|---|---|---|---|
| Real-world Bunny | Random | 12.25 | 0.51 | 0.54 |
| | FisherRF RGB | 12.49 | 0.52 | 0.56 |
| | FisherRF Combined | 12.43 | 0.52 | 0.56 |
| | **FisherRF Depth** | **13.09** | **0.53** | **0.53** |
| Toy Car | Random | 11.40 | 0.45 | 0.55 |
| | **Our Method** | **11.75** | **0.47** | **0.52** |
| Mirror | Random | 14.66 | 0.72 | 0.50 |
| | **Our Method** | **16.50** | **0.76** | **0.41** |
| Prism | Random | 16.63 | 0.73 | 0.41 |
| | **Our Method** | **16.77** | **0.74** | **0.39** |
| Bunny/Ridged Object | Random | 17.24 | 0.75 | 0.39 |
| | **Our Method** | **17.47** | **0.76** | **0.38** |

TABLE VI: FisherRF Online Real-World Experiment

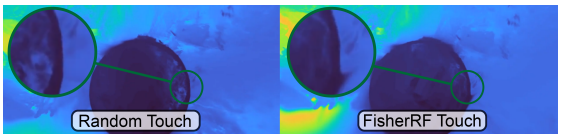


Fig. 7: **FisherRF Guided Touch Selection**

## V. CONCLUSION

In this work, we demonstrate the usefulness of few-shot active view and touch selection for robot manipulators using 3DGS. First, we utilize SAM2 and monocular depth models to generate high quality depth map as guidance for Gaussian Splatting training. Second, we use FisherRF to actively select both camera and touch poses and progressively reconstruct the object surface. Third, we fuse tactile data into 3DGS to represent a challenging object. Future work entails integrating our pipeline with both vision and touch **online** in order to reconstruct a digital twin.

## REFERENCES


[1] G. Lu, S. Zhang, Z. Wang, C. Liu, J. Lu, and Y. Tang, "Manigaussian: Dynamic gaussian splatting for multi-task robotic manipulation," *arXiv preprint arXiv:2403.08321*, 2024.

[2] O. Shorinwa, J. Tucker, A. Smith, A. Swann, T. Chen, R. Firoozi, M. D. Kennedy, and M. Schwager, "Splat-mover: Multi-stage, open-vocabulary robotic manipulation via editable gaussian splatting," in *8th Annual Conference on Robot Learning*, 2024.

[3] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics (ToG)*, vol. 42, no. 4, pp. 1–14, 2023.

[4] J. Chung, J. Oh, and K. M. Lee, "Depth-regularized optimization for 3d gaussian splatting in few-shot images," 2024.

[5] A. Paliwal, W. Ye, J. Xiong, D. Kotovenko, R. Ranjan, V. Chandra, and N. K. Kalantari, "Coherentgs: Sparse novel view synthesis with coherent 3d gaussians," 2024. [Online]. Available: https://arxiv.org/abs/2403.19495

[6] Wang, Shuzhe and Leroy, Vincent and Cabon, Yohann and Chidlovskii, Boris and Revaud Jerome, "DUSt3R: Geometric 3D Vision Made Easy," 2023.

[7] Z. Zhu, Z. Fan, Y. Jiang, and Z. Wang, "Fsgs: Real-time few-shot view synthesis using gaussian splatting," *arXiv preprint arXiv:2312.00451*, 2023.

[8] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer, "Sam 2: Segment anything in images and videos," 2024. [Online]. Available: https://arxiv.org/abs/2408.00714

[9] K. Lin and B. Yi, "Active view planning for radiance fields," in *Robotics Science and Systems*, 2022.

[10] X. Pan, Z. Lai, S. Song, and G. Huang, "Activenerf: Learning where to see with uncertainty estimation," in *ECCV*. Springer, 2022, pp. 230–246.

[11] S. Lee, L. Chen, J. Wang, A. Liniger, S. Kumar, and F. Yu, "Uncertainty guided policy for active robotic 3d reconstruction using neural radiance fields," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12 070–12 077, 2022.

[12] J. Shen, R. Ren, A. Ruiz, and F. Moreno-Noguer, "Estimating 3d uncertainty field: Quantifying uncertainty for neural radiance fields," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 2375–2381.

[13] W. Jiang, B. Lei, and K. Daniilidis, "Fisherrf: Active view selection and uncertainty quantification for radiance fields using fisher information," in *ECCV*, 2024.

[14] A. Swann, M. Strong, W. K. Do, G. S. Camps, M. Schwager, and M. Kennedy III, "Touch-gs: Visual-tactile supervised 3d gaussian splatting," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.

[15] M. Comi, A. Tonioni, M. Yang, J. Tremblay, V. Blukis, Y. Lin, N. F. Lepora, and L. Aitchison, "Snap-it, tap-it, splat-it: Tactile-informed 3d gaussian splatting for reconstructing challenging surfaces," *arXiv preprint arXiv:2403.20275*, 2024.

[16] E. Smith, D. Meger, L. Pineda, R. Calandra, J. Malik, A. Romero Soriano, and M. Drozdzal, "Active 3d shape reconstruction from vision and touch," *Advances in Neural Information Processing Systems*, vol. 34, pp. 16 064–16 078, 2021.

[17] S. Suresh, H. Qi, T. Wu, T. Fan, L. Pineda, M. Lambeta, J. Malik, M. Kalakrishnan, R. Calandra, M. Kaess, J. Ortiz, and M. Mukadam, "Neural feels with neural fields: Visuo-tactile perception for in-hand manipulation," 2023.

[18] F. Yang, C. Feng, Z. Chen, H. Park, D. Wang, Y. Dou, Z. Zeng, X. Chen, R. Gangopadhyay, A. Owens *et al.*, "Binding touch to everything: Learning unified multimodal tactile representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 26 340–26 353.

[19] L. Fu, G. Datta, H. Huang, W. C.-H. Panitch, J. Drake, J. Ortiz, M. Mukadam, M. Lambeta, R. Calandra, and K. Goldberg, "A touch, vision, and language dataset for multimodal alignment," *arXiv preprint arXiv:2402.13232*, 2024.

[20] D. Watkins-Valls, J. Varley, and P. Allen, "Multi-modal geometric learning for grasping and manipulation," in *2019 International conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 7339–7345.

[21] E. Smith, R. Calandra, A. Romero, G. Gkioxari, D. Meger, J. Malik, and M. Drozdzal, "3d shape reconstruction from vision and touch," *Advances in Neural Information Processing Systems*, vol. 33, pp. 14 193–14 206, 2020.

[22] N. Houlsby, F. Huszar, Z. Ghahramani, and M. Lengyel, "Bayesian active learning for classification and preference learning," *CoRR*, vol. abs/1112.5745, 2011. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1112.html#abs-1112-5745

[23] A. Kirsch and Y. Gal, "Unifying approaches in active learning and active sampling via fisher information and information-theoretic quantities," *Trans. Mach. Learn. Res.*, vol. 2022, 2022. [Online]. Available: https://openreview.net/forum?id=UVDAKQANOW

[24] E. Daxberger, A. Kristiadi, A. Immer, R. Eschenhagen, M. Bauer, and P. Hennig, "Laplace redux–effortless Bayesian deep learning," in *NeurIPS*, 2021.

[25] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao, "Depth anything v2," *arXiv preprint arXiv:2406.09414*, 2024.

[26] M. Hu, W. Yin, C. Zhang, Z. Cai, X. Long, H. Chen, K. Wang, G. Yu, C. Shen, and S. Shen, "Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation," *arXiv preprint arXiv:2404.15506*, 2024.

[27] T. Xie, Z. Zong, Y. Qiu, X. Li, Y. Feng, Y. Yang, and C. Jiang, "Physgaussian: Physics-integrated 3d gaussians for generative dynamics," *arXiv preprint arXiv:2311.12198*, 2023.

[28] M. Turkulainen, X. Ren, I. Melekhov, O. Seiskari, E. Rahtu, and J. Kannala, "Dn-splatter: Depth and normal priors for gaussian splatting and meshing," 2024.

[29] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," *CoRR*, vol. abs/1608.06993, 2016. [Online]. Available: http://arxiv.org/abs/1608.06993

[30] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, J. Kerr, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, D. McAllister, and A. Kanazawa, "Nerfstudio: A modular framework for neural radiance field development," in *ACM SIGGRAPH 2023 Conference Proceedings*, ser. SIGGRAPH '23, 2023.