

Splat & Sculpt: Single-View 3D Reconstruction via Gaussian Sculpting Networks

Ruihan Xu¹, Anthony Opipari¹, Joshua Mah¹, Stanley Lewis¹
Haoran Zhang¹, Hanzhe Guo¹, and Odest Chadwicke Jenkins¹

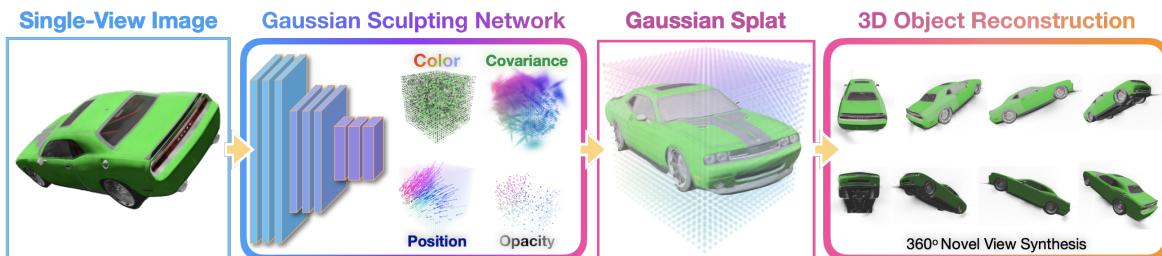


Fig. 1: Overview of the Gaussian Sculpting Network (GSN) for single-view 3D reconstruction. **Left:** Using a single observed image, the GSN transforms a canonical Gaussian splat with predicted colors, covariances, positions, and opacities. **Right:** Output from the GSN is combined to form a Gaussian splat representing the reconstructed 3D surface geometry and texture of the observed object.

Abstract—This paper introduces Gaussian Sculpting Networks as an approach for 3D object reconstruction from single-view image observations. Gaussian sculpting networks (GSNs) take a single observation as input to generate a Gaussian splat representation describing the observed object’s shape and texture (Fig. 1). By using a shared feature extractor before decoding Gaussian colors, covariances, positions, and opacities, GSNs achieve extremely high throughput ($>150\text{FPS}$). Experiments demonstrate that GSNs can be trained efficiently using a multi-view rendering loss and are competitive, in quality, with expensive diffusion-based reconstruction algorithms. The GSN model is validated on multiple benchmark experiments. Moreover, we demonstrate the potential for GSNs to be used within a robotic manipulation pipeline for object-centric grasping.

I. INTRODUCTION

Autonomous robots tasked with operating in complex, unstructured environments must be able to perceive their environment and the objects within it. In particular, perceiving the shape and visual properties of objects is needed in order for robots to effectively plan and act using those objects. At a given instant, a robot can typically only observe part of an object’s surface and must rely on 3D reconstruction algorithms to perceive the remaining occluded surface. Moreover, the ability to understand and interact with novel objects that were previously unobserved during model training remains a critical requirement for unstructured settings.

Novel view synthesis offers a promising direction for robots to reconstruct detailed object models using only sparse viewpoints from their environment. For example, Neural Radiance Fields (NeRFs), is a technique that enables novel view synthesis and 3D reconstruction by using neural implicit

representations to render new views of a scene based only on a small set of observed images [1, 2]. While NeRFs have demonstrated impressive results and received a surge of interest within robotic use cases, their implicit nature often poses challenges for direct integration with robotic planning and control modules [3, 4, 5, 6, 7].

Against this backdrop, Gaussian Splatting has emerged as an explicit alternative for representing scenes and objects while also enabling novel view synthesis [8]. This approach uses a collection of Gaussian-shaped primitives to approximate a continuous scene volume along with a differentiable, highly efficient rendering technique to optimize each Gaussian against the observed views. The explicit nature of Gaussian Splatting representations has the potential to be used within robotic perception by downstream planning and control algorithms. Recent work has demonstrated the potential for Gaussian Splatting within robotic manipulation [9, 10] and navigation [11, 12]. While Gaussian Splatting has enabled novel view synthesis with an explicit representation useful for robotics, for 3D reconstruction it requires multiple diverse image observations, which are not always available. In this project, we set out to enable high throughput 3D reconstruction using single-view images with Gaussian Splatting. More specifically, we set out to develop a model that can generate an object-centric Gaussian splat reconstruction given only a single image.

The present paper makes the following contributions:

- 1) Propose and develop Gaussian Sculpting Networks (GSNs), to generate Gaussian splats from single-view image observations in real time.
- 2) Perform quantitative and qualitative evaluations of GSNs to understand relative tradeoffs in comparison to existing 3D reconstruction approaches.
- 3) We demonstrate the potential for GSN to be used within a robotic manipulation pipeline for grasping.

¹All authors are affiliated with the Department of Robotics, University of Michigan, Ann Arbor, MI, USA, 48109.

*R. Xu and A. Opipari are the corresponding authors: rrxu@umich.edu, topipari@umich.edu

II. RELATED WORK

A. Multi-View 3D Reconstruction

Multi-view 3D reconstruction is a foundational topic within computer vision and robotic perception that seeks to infer the three-dimensional structure of a scene using numerous images taken from different vantage points throughout the scene. Classical approaches, such as Structure from Motion (SfM) and Multi-View Stereo enable the derivation of 3D points or volumetric representations by triangulating matched features across multiple images [13, 14]. End-to-end deep learning-based approaches have been considered including MVSNet [15] and DeepMVS [16] for depth map estimation as well as DeepSDF [17] as a continuous shape representation. More recently, neural radiance fields (NeRFs) are proposed as an approach to implicitly reconstruct volumetric scenes from sparse image observations [1, 2, 18]. Following the surge of interest in NeRFs, Gaussian splatting is developed as an explicit approach to novel view synthesis from multi-view images [8]. Notably, Gaussian splatting represents a scene as a mixture of Gaussians distributed throughout the scene and optimized using gradient descent. In contrast to these approaches, the present paper sets out to reconstruct 3D scenes using only a single observed image.

B. Single-View 3D Reconstruction

In single-view 3d reconstruction, algorithms have access to only a single image of the object whose shape they set out to estimate. Early approaches used shading cues to estimate geometric surface contours [19, 20]. Following the rise of deep learning, convolutional neural networks have been proposed for voxel-grid reconstruction from single images [21]. Image-conditioned reconstruction into point cloud representations has been proposed using convolution [22] and diffusion-based architectures [23, 24]. Following the introduction of Gaussian splatting, 3D reconstruction using splats as a 3D representation has gained interest [25, 26, 27]. Watson et al. introduce 3DiM to generate novel views with a single image using image-based diffusion, then generate 3D splats using those generated images [26]. Similarly, Feng et al. employ image-level diffusion with additional geometric constraints to improve the reconstruction speed and quality. Szymanowicz et al. propose a convolutional architecture to directly regress an output Gaussian Splat representation offering substantial speedups [25]. We set out to further improve the efficiency of single-view reconstruction based on Gaussian splatting.

III. METHOD

A. Gaussian Primitive: 3D Gaussian Splatting

A Gaussian Primitive \mathcal{G} is parameterized by 4 variables: mean $\mu \in \mathbb{R}^3$, covariance $\Sigma \in \mathbb{R}^{3 \times 3} : \Sigma \succ \mathbf{0}$, RGB color $c \in \mathbb{R}^3 : \mathbf{0} \leq c \leq \mathbf{1}$, and opacity $\alpha \in \mathbb{R} : 0 < \alpha \leq 1$. To simplify optimization, we follow [8] and decompose Σ into scale $s \in \mathbb{R}^3 : s > 0$ and rotation $r \in \mathbb{R}^4 : |r| = rr^* = 1$ represented using rotation quaternions. The resulting Gaussian will have the form of $\mathcal{G} = (\mu, s, r, c, \alpha)$. A scene or object can

thus be represented by a collection of Gaussians that form a Gaussian Splat

$$\mathcal{S} := \{\mathcal{G}_i, i = 1, \dots, N\} = \{(\mu_i, s_i, r_i, c_i, \alpha_i), i = 1, \dots, N\}$$

The 3D Gaussian Splatting [8] renderer \mathcal{R} takes a 3D Gaussian Splat \mathcal{S} and maps it into a 2D image \mathcal{I} using camera pose $p \in \mathbb{R}^3$ and rotation $\mathbf{R} \in \mathbb{R}^{3 \times 3} : \mathbf{R}^T \mathbf{R} = \mathbf{I}$. Here, we want to find the function \mathcal{F} that inverts the process \mathcal{R} such that $\mathcal{F}(\mathcal{I}) = \mathcal{R}^{-1}(\mathcal{I}) = \mathcal{S}$. The reconstruction from this single image should preserve the correct 3D representation of the object, allowing it to render novel views of the object from viewpoints different to the input image.

Instead of letting the neural network directly predict the 5 parameters for each Gaussian, our method predicts a deviation of the parameters from a canonical cube. The canonical cube refers to a collection of Gaussians, $\bar{\mathcal{G}}_i$, that are evenly spaced apart on a unit cube with covariances $\bar{\Sigma}_i = \delta \mathbf{I} \forall i = 1, \dots, N$ where $\delta \in \mathbb{R} : \delta > 0$ is a small scalar constant, with color and opacity values that are randomly sampled without any constraints. The resultant initial Gaussian Splat would thus have predefined $(\bar{\mu}_i, \bar{s}_i) \forall i = 1, \dots, N$ and would resemble a point cloud since the covariance scale is isotropic in \mathbb{R}^3 . To generate the predicted Gaussian Splat \mathcal{S}_{pred} , our network \mathbf{f} predicts an error term $\Delta\mu_i, \Delta s_i$ and values r_i, c_i, α_i for a fixed number of Gaussians N such that

$$\mathbf{f}(\mathcal{I}) = \{(\Delta\mu_i, \Delta s_i, r_i, c_i, \alpha_i), i = 1, \dots, N\}$$

The final predicted Gaussian Splat has the form

$$\mathcal{S}_{pred} = \{(\bar{\mu}_i + \Delta\mu_i, \bar{s}_i + \Delta s_i, r_i, c_i, \alpha_i), i = 1, \dots, N\}$$

B. Network Architecture

As illustrated in Fig. 2, the network feature extractor takes the image as input and encodes it into a latent vector followed by a decoding step into Gaussian parameters.

1) *Encoder*: We adopted ResNet [28] as the backbone of the encoder with some modifications. Instead of directly using the 2048 size latent vector output from the ResNet, we take the output from the second last layer that has a size of $2048 \times 7 \times 7$ latent tensor. This tensor is then fed into two 2D convolutions followed by ReLU activation and finally gets a vector of size 1×2352 after flattening the tensor. The design was chosen since, experimentally, taking the output from the ResNet with a smaller receptive field allows the network to capture more fine-grained details to reconstruct the Gaussian Splat. We discuss this choice in more detail in the ablation section.

2) *Decoder*: Parallel MLPs are used to decode the latent vector into Gaussian parameters $\Delta S = \{(\Delta\mu_i, \Delta s_i, r_i, c_i, \alpha_i), i = 1, \dots, N\}$. For each of the 5 parameters, we have an independent MLP layer to output $N \times \text{size of parameter}$ number of values, where N is predefined as the number of Gaussian. For example, to describe $\Delta\mu$ for one Gaussian, we need three values, $\Delta\mu_x, \Delta\mu_y, \Delta\mu_z$, so the output of the MLP for $\Delta\mu$ will have $N \times 3$ output size. For each MLP, there consists of two fully connected linear layers and the first linear layer is followed

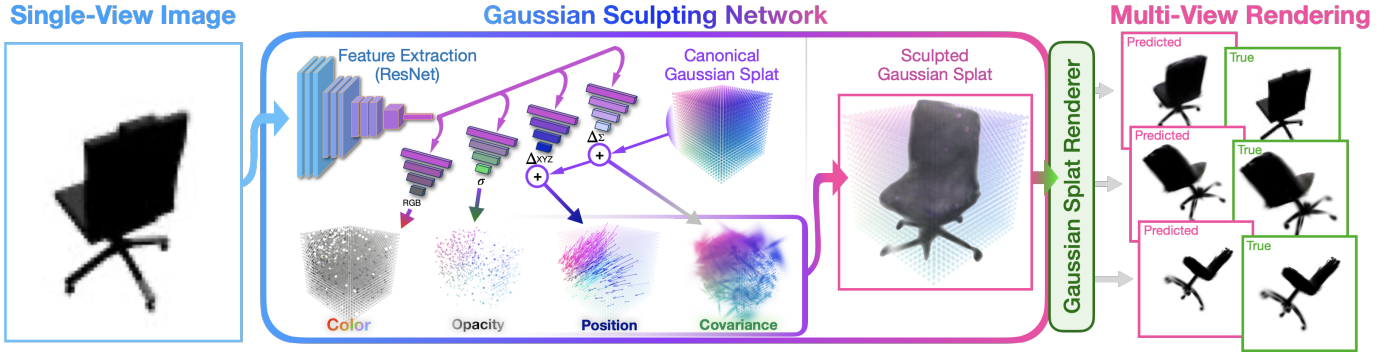


Fig. 2: **Gaussian Sculpting Network Architecture.** Our Encoder-Decoder style network takes an input image and encodes it into a latent vector. Subsequently, a decoder with parallel MLPs decodes the latent vector into Gaussian parameters, sculpting a canonical Gaussian Splat into a 3D object presented in the input image. Finally, we perform multi-view rendering to obtain various novel views for loss calculation.

by a ReLU activation. For predicted scale $\bar{s}_i + \Delta s_i$, we apply ReLU activation followed by addition with a small scalar constant $\epsilon \in \mathbb{R} : \epsilon > 0$ to ensure that the covariance is positive definite. For predicted rotation r_i , we normalize r_i without making the real quaternion term positive, whereas for color c_i and opacity α_i , we apply a dimensionwise sigmoid function.

C. Loss Formulation

To train the neural network \mathcal{F} , we aim to minimize the difference between a set of ground truth images with different camera poses and the rendered images from a predicted Gaussian Splat. Specifically, given an input image of an object, the network predicts a 3D Gaussian Splat conditioned on this input image. We then render the Splat from different views and compare it with available ground truth images taken from those viewpoints.

$$\begin{aligned} \tilde{\mathcal{I}}_i &:= \mathcal{R}(\tilde{\mathcal{S}}, p_i, \mathbf{R}_i) \\ \mathcal{I}_i &= \mathcal{R}(\mathcal{F}(\mathcal{I}_{input}), p_i, \mathbf{R}_i) \\ \mathcal{L}_{loss} &= \frac{1}{k} \sum_{i=1}^k f_{loss}(\tilde{\mathcal{I}}_i, \mathcal{I}_i) \end{aligned}$$

where $\tilde{\mathcal{I}}_i$ denotes the ground truth image i , $\tilde{\mathcal{S}}$ denotes some hypothetical ground truth Gaussian Splat that would generate $\tilde{\mathcal{I}}$ and k represents the number of sampled images for each object.

For the actual loss calculation, we adopt 4 different losses: L2, L1, Lpips, and DSSIM. The total loss is a combination of all 4 losses. Our specific loss formulation comes from a combination of the Splatter Image loss [25] and from the original 3D Gaussian Splatting work [8].

$$\mathcal{L}_{lpips} = (1 - \lambda_{lpips})\mathcal{L}_{L2} + \lambda_{lpips}\mathcal{L}_{lpips} \quad (1)$$

$$\begin{aligned} \mathcal{L}_{DSSIM} &= (1 - \lambda_{DSSIM})\mathcal{L}_{L1} \\ &\quad + \lambda_{DSSIM}\mathcal{L}_{DSSIM} \end{aligned} \quad (2)$$

$$\mathcal{L}_{loss} = \mathcal{L}_{lpips} + \mathcal{L}_{DSSIM} \quad (3)$$

Method	Chairs			Cars		
	PSNR \uparrow	SSIM \uparrow	Lpips \downarrow	PSNR \uparrow	SSIM \uparrow	Lpips \downarrow
SRN	22.89	0.89	0.104	22.25	0.88	0.129
CodeNeRF	23.66	0.90	0.166	23.80	0.91	0.128
PixelNeRF	23.72	0.90	0.128	23.17	0.89	0.146
Splatter*	23.03	0.91	0.101	23.72	0.91	0.107
GSN (ours)*	24.33	0.92	0.123	23.27	0.91	0.120

TABLE I: **ShapeNet-SRN: Single-view 3D reconstruction.** Methods marked * are trained on our local machine with RTX A6000 for a fixed amount of iteration in a limited amount of time before the convergence. While the statistics for other methods are adopted from experiments in prior works. The result shows that our model performs better than some of the prior methods even before the training is converged.

Method	FPS \uparrow
Splatter	50
GSN (ours)	164

TABLE II: Performance during inference. Our method is significantly faster than the recent baseline for generating Gaussian Splat from a single input image.

Qualitatively, we note that L2 and DSSIM [29] seemed to contribute to shaping the overall structure and color of the object. Using only these losses, the final output would appear fuzzy, a phenomenon noted in [30], hence L1 and lpips were added to help learn more fine-grained details.

In practice, we randomly sample 3 other images on top of the original input image to compute the loss for each data point. The original input image would be used to assess how well the model can encode and reconstruct observable object details while the final 3 images would capture how well the model can generalize for unseen portions of the object.

IV. EXPERIMENT

To evaluate the performance of our proposed framework, we trained our model against the chairs and cars dataset provided by [25]. We trained our model and image-splatter [25] for roughly the same number of iterations (200k).

A. Single-View 3D reconstruction on Chairs and Cars

In this task, the model predicts a 3D representation (Gaussian Splat in this case) of an object given a single input view of that object. Novel target views are then rendered from this representation and the performance is measured by comparing these rendered images with the ground truth images. We follow the practice of prior works, such as [31] [1] [32] [33], and

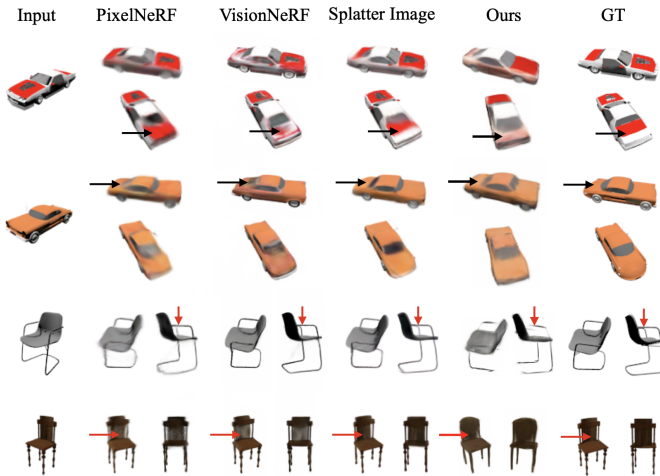


Fig. 3: Qualitative single-view reconstruction comparison.

measure the performance in terms of the Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM) and perceptual quality (LPIPS). For PSNR and SSIM, higher values indicate better quality, whereas for LPIPS, lower values are preferable. For datasets, we adopt the ShapeNet-SRN dataset for cars and chairs from [32] which aligns with the recent single-view Gaussian Splat generation model [25].

Results. We trained our model on the chair dataset for 8 epochs and the cars dataset for 15 epochs over two days, using a batch size of 8. To ensure direct comparability with the Splatter Image model [25], we converted epochs to iterations, maintaining a similar number of training iterations for both datasets and trained the Splatter Image for the same amount of iteration time. For the chair dataset, one epoch corresponds to 28,825 iterations; thus, 8 epochs require 230,600 iterations. Similarly, for the car dataset, one epoch comprises 15,363 iterations, resulting in a total of 230,445 iterations for 15 epochs. Both models are trained and evaluated on the same hardware setting with RTX A6000. Table I and Figure 3 show how our model performs quantitatively and qualitatively to other baselines. As demonstrated, our method performs roughly 3 times faster than the next fastest method (splatter-image) while having comparable quantitative image reconstruction quality to other recent baselines. Qualitatively, we note that while our model can accurately capture larger shapes, it still struggles with extremely fine-grained details, especially on the chair dataset which consists of many skinny leg and handle structures.

B. Comparative Study

We consider three key areas: the size of the ResNet architecture used, the selection of the latent vector, and the selection of the origin of the unit cube used to generate the initial set of Gaussian. We compare between ResNet50 and ResNet152 for the ResNet Architecture. For the latent vector, we consider either taking the final output of the ResNet (size 2048) or the second last layer (size 100352) and compressing it with 2 convolution layers (size 2352), ensuring the size of the latent

Method	ResNet50		ResNet152	
	Train	Val	Train	Val
Zero-Centered (2048)	4.37×10^{-3}	5.44×10^{-3}	4.30×10^{-3}	5.32×10^{-3}
Off-Centered (2048)	5.07×10^{-3}	5.64×10^{-3}	5.59×10^{-3}	6.07×10^{-3}
Zero-Centered (2352)	3.8×10^{-3}	5.35×10^{-3}	4.02×10^{-3}	4.68×10^{-3}
Off-Centered (2352)	4.62×10^{-3}	5.58×10^{-3}	4.68×10^{-3}	5.64×10^{-3}

LPIPs results for comparative studies.

LPIPs results for comparative studies.

TABLE III: weighted L2 +

LPIPs results for comparative studies.

vector in either case to be roughly similar. For the cube origin we consider one where the camera is centered on the origin and the cube 1 meter ahead from the camera, and the other where the unit cube is on the origin and the camera is 1 meter away from the origin. In both cases, the Gaussian splat has been rotated so that the camera remains upright in both frames of reference. This implies that the deformation of the Gaussians on the unit cube directly encodes the 3D shifts required to capture the shape of the object with respect to the camera viewpoint.

Results. For each of the scenarios listed above, we trained with a smaller data set containing 30 car objects with 150 views each. We use LPIPs as the quantitative measure. Table IV-B shows the results of the comparative study. The greatest impact comes from centering the unit cube on the origin of the frame. The low impact of the latent vector and architecture might however be due to the small dataset and further studies would be conducted given more time.

C. Grasp Generation

We also evaluate the effectiveness of our generated 3D Gaussian Splat for practical robot manipulation tasks to gauge its suitability for real-world applications. In this experiment, the model predicts a Gaussian splat based on an input image of an object. Subsequently, the 3D object is rendered into a novel view, which is utilized for robot grasp prediction by Dex-Net 2.0 [34]. Specifically, based on this novel view, we conduct depth estimation employing MiDaS [35] to generate a depth map, and Segment Anything (SAM) [36] to produce a binary mask. Utilizing these outputs, Dex-Net generates a grasp pose along with a q-value, where a higher q-value indicates better grasp quality. During the trials, Dex-Net can generate grasps for a chair sample with a q-value=0.882 and for a car sample with a q-value=0.614. This result is shown in Figure 4.

V. CONCLUSION

This paper makes three central contributions: (1) a Gaussian Sculpting Network for real-time single-image 3D reconstruction, (2) experiments validating the sculpting network design, and (3) demonstrations to highlight how GSN is applicable to a robotic grasping pipeline. Experiments demonstrated the high-throughput of GSNs and suggest their potential for application to robotic perception tasks. Results from this study suggest future directions to further improve reconstruction quality by focusing on fine-grained texture detail.





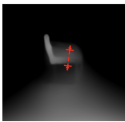





Input View	Novel View	Depth Map	Binary Mask	Generated Grasp
				
				

Fig. 4: Generated grasp using Dex-Net 2.0 from a novel view render from 3D Gaussian Splat generated by our model. The Objects are scaled down to a suitable size for grasping. In these examples, the Dex-Net 2.0 can generate grasps for the chair with q-value=0.882 and for cars with q-value=0.614

REFERENCES

- [1] 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.
- [2] S. Pan, L. Jin, H. Hu, M. Popović, and M. Bennewitz, "How many views are needed to reconstruct an unknown object using nerf?," *arXiv preprint arXiv:2310.00684*, 2023.
- [3] J. Kerr, L. Fu, H. Huang, Y. Avigal, M. Tancik, J. Ichnowski, A. Kanazawa, and K. Goldberg, "Evo-nerf: Evolving nerf for sequential robot grasping of transparent objects," in *6th Annual Conference on Robot Learning*, 2022.
- [4] M. Adamkiewicz, T. Chen, A. Caccavale, R. Gardner, P. Culbertson, J. Bohg, and M. Schwager, "Vision-only robot navigation in a neural radiance world," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4606–4613, 2022.
- [5] Y. Li, S. Li, V. Sitzmann, P. Agrawal, and A. Torralba, "3d neural scene representations for visuomotor control," in *Conference on Robot Learning*, pp. 112–123, PMLR, 2022.
- [6] A. Rosinol, J. J. Leonard, and L. Carlone, "Nerf-slam: Real-time dense monocular slam with neural radiance fields," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3437–3444, IEEE, 2023.
- [7] S. Lewis, B. Aldeeb, A. Opipari, E. A. Olson, C. Kisailus, and O. C. Jenkins, "Nerf-frenemy: Co-opting adversarial learning for autonomy-directed co-design,"
- [8] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, July 2023.
- [9] J. Abou-Chakra, K. Rana, F. Dayoub, and N. Sinderhauf, "Physically embodied gaussian splatting: Embedding physical priors into a visual 3d world model for robotics," in *Conference on Robot Learning*, no. 7th, 2023.
- [10] Y. Zheng, X. Chen, Y. Zheng, S. Gu, R. Yang, B. Jin, P. Li, C. Zhong, Z. Wang, L. Liu, *et al.*, "Gaussiangrasper: 3d language gaussian splatting for open-vocabulary robotic grasping," *arXiv preprint arXiv:2403.09637*, 2024.
- [11] N. Keetha, J. Karhade, K. M. Jatavallabhula, G. Yang, S. Scherer, D. Ramanan, and J. Luiten, "Splatam: Splat, track map 3d gaussians for dense rgb-d slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.
- [12] G. Chen and W. Wang, "A survey on 3d gaussian splatting," *arXiv preprint arXiv:2401.03890*, 2024.
- [13] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.
- [14] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1, pp. 519–528, 2006.
- [15] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan, "Mvsnet: Depth inference for unstructured multi-view stereo," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 767–783, 2018.
- [16] P.-H. Huang, K. Matzen, J. Kopf, N. Ahuja, and J.-B. Huang, "Deepmvs: Learning multi-view stereopsis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2821–2830, 2018.
- [17] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "DeepSDF: Learning continuous signed distance functions for shape representation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 165–174, 2019.
- [18] J. Reizenstein, R. Shapovalov, P. Henzler, L. Sbordone, P. Labatut, and D. 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*, pp. 10901–10911, 2021.
- [19] B. K. Horn, "Shape from shading: A method for obtaining the shape of a smooth opaque object from one view," 1970.
- [20] J.-D. Durou, M. Falcone, and M. Sagona, "Numerical methods for shape-from-shading: A new survey with benchmarks," *Computer Vision and Image Understanding*, vol. 109, no. 1, pp. 22–43, 2008.
- [21] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee, "Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision," *Advances in neural information processing systems*, vol. 29, 2016.
- [22] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "Pu-net: Point cloud upsampling network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2790–2799, 2018.
- [23] L. Zhou, Y. Du, and J. Wu, "3d shape generation and completion through point-voxel diffusion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5826–5835, 2021.
- [24] A. Nichol, H. Jun, P. Dhariwal, P. Mishkin, and M. Chen, "Point-e: A system for generating 3d point clouds from complex prompts," 2022.
- [25] S. Szymanowicz, C. Rupprecht, and A. Vedaldi, "Splatter image: Ultra-fast single-view 3d reconstruction," 2023.
- [26] D. Watson, W. Chan, R. Martin-Brualla, J. Ho, A. Tagliasacchi, and M. Norouzi, "Novel view synthesis with diffusion models," 2022.
- [27] Q. Feng, Z. Xing, Z. Wu, and Y.-G. Jiang, "Fdgaussian: Fast gaussian splatting from single image via geometric-aware diffusion model," *arXiv preprint arXiv:2403.10242*, 2024.
- [28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.
- [29] A. H. Baker, A. Pinard, and D. M. Hammerling, "Dssim: a structural similarity index for floating-point data," 2023.
- [30] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," 2018.
- [31] 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," 2020.
- [32] V. Sitzmann, M. Zollhoefer, and G. Wetzstein, "Scene representation networks: Continuous 3d-structure-aware neural scene representations," in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.
- [33] K.-E. Lin, L. Yen-Chen, W.-S. Lai, T.-Y. Lin, Y.-C. Shih, and R. Ramamoorthi, "Vision transformer for nerf-based view synthesis from a single input image," 2022.
- [34] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," 2017.
- [35] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," 2020.
- [36] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," 2023.