2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Reinforcement Learning with Generalizable Gaussian Splatting

Jiaxu Wang[1,*], Qiang Zhang[1,2,*], Jingkai Sun[1], Jiahang Cao[1], Gang Han[2], Wen Zhao[2], Weining Zhang[2], Yecheng Shao[3,4], Yijie Guo[2], Renjing Xu[1,†]

***Abstract*— An excellent representation is crucial for reinforcement learning (RL) performance, especially in vision-based reinforcement learning tasks. The quality of the environment representation directly influences the achievement of the learning task. Previous vision-based RL typically uses explicit or implicit ways to represent environments, such as images, points, voxels, and neural radiance fields. However, these representations contain several drawbacks. They cannot either describe complex local geometries or generalize well to unseen scenes, or require precise foreground masks. Moreover, these implicit neural representations are akin to a "black box", significantly hindering interpretability. 3D Gaussian Splatting (3DGS), with its explicit scene representation and differentiable rendering nature, is considered a revolutionary change for reconstruction and representation methods. In this paper, we propose a novel Generalizable Gaussian Splatting framework to be the representation of RL tasks, called GSRL. Through validation in the RoboMimic environment, our method achieves better results than other baselines in multiple tasks, improving the performance by 10%, 44%, and 15% compared with baselines on the hardest task. This work is the first attempt to leverage generalizable 3DGS as a representation for RL.**

I. Introduction

In reinforcement learning (RL), obtaining a high-quality representation is crucial for problem-solving [1]–[7]. This challenge becomes even more pronounced in vision-based RL tasks, where the ability to derive effective representations from complex visual scenes is essential for developing successful downstream strategies. Particularly in application scenarios like robotic manipulation, high-quality environmental representations would influence the success rate of tasks. To tackle this issue, we extract information from high-dimensional visual data and convert it into representations that are compatible with deep RL algorithms. This process demands not only the skill to comprehend and process high-dimensional data but also the ability to capture the fundamental characteristics of the environment. Such capabilities allow robots or other automated systems to make precise decisions and actions based on these characteristics. Consequently, developing a versatile representation method that can accurately capture environmental features.

In previous research, representation methods in RL have primarily been divided into two types: low-dimensional and high-dimensional representations. Low-dimensional representation methods focus on expressing environmental information in a structured form, such as object positions, postures, or even shapes from visual inputs [4]. Even though low-dimensional representation methods can explicitly and accurately express environmental information, they are impractical to obtain. Conversely, high-dimensional representation methods tend to adopt an end-to-end approach to handle high-dimensional visual information. This method directly transforms visual information into high-dimensional features by using pre-trained models, such as [8]–[10].

However, only 2D images are not capable of perceiving 3D real-world structures. Thus researchers adopt 3D-aware explicit representations in RL, such as RGB-D images [11], [12], multiview images [13], [14], voxels [15], and point clouds [16], [17]. Unfortunately, these explicit scene representations cannot describe complex 3D local geometries due to the limitation of their resolutions, therefore, they cannot be considered an expected representation for RL. With the advancement of the Neural Radiance Field (NeRF) [18]–[22], which is a novel implicit, 3D-consistent scene representation. Researchers have applied such implicit NeRF formats to environmental representation [23]–[25]. Nevertheless, these NeRF-based methods either require high-quality foreground masks to distinguish the target objects or need high-level semantic feature maps produced by other large-scale deep learning models such as stable diffusion. More importantly, this type of method cannot smoothly generalize to unseen scenes because they tend to use a single vector to represent the whole scene. Therefore, an important research question arises: can we develop a new environmental representation that can both explicitly express 3D-aware structural information and capture 3D-consistent local geometry?

3D Gaussian Splatting (3DGS) [26] provides an approach to answer the question. This technique not only enables 3D-consistent content expression but also explicitly represents detailed local geometries. Owing to its differentiable properties, it can be integrated into many deep learning frameworks. However, conventional 3DGS requires per-scene optimization that obstacles its usage in RL. Motivated by this, we introduce a generalizable 3DGS framework that is pretrained for obtaining 3D-aware prior knowledge of specific tasks, enabling the extraction of high-quality 3DGS from the visual observations for RL tasks. We select RoboMimic [27] as our training and evaluation environment. RoboMimic is a sophisticated dataset and benchmark for robotics research. To put it in a nutshell, our contributions can be summarized as follows:

* are equal contributors, † are the corresponding authors
[1]The authors are with The Hong Kong University of Science and Technology (Guangzhou), China. {jwang457, qzhang749, jsun444, }@connect.hkust-gz.edu.cn, renjingxu@ust.hk
[2]The authors are with Beijing Innovation Center of Humanoid Robotics Co., Ltd. jack.guo@x-humanoid.com
[3]The author is with Center for X-Mechanics, Zhejiang University, China.
[4]The author is with Institute of Applied Mechanics, Zhejiang University, China. shaoyecheng@zju.edu.cn

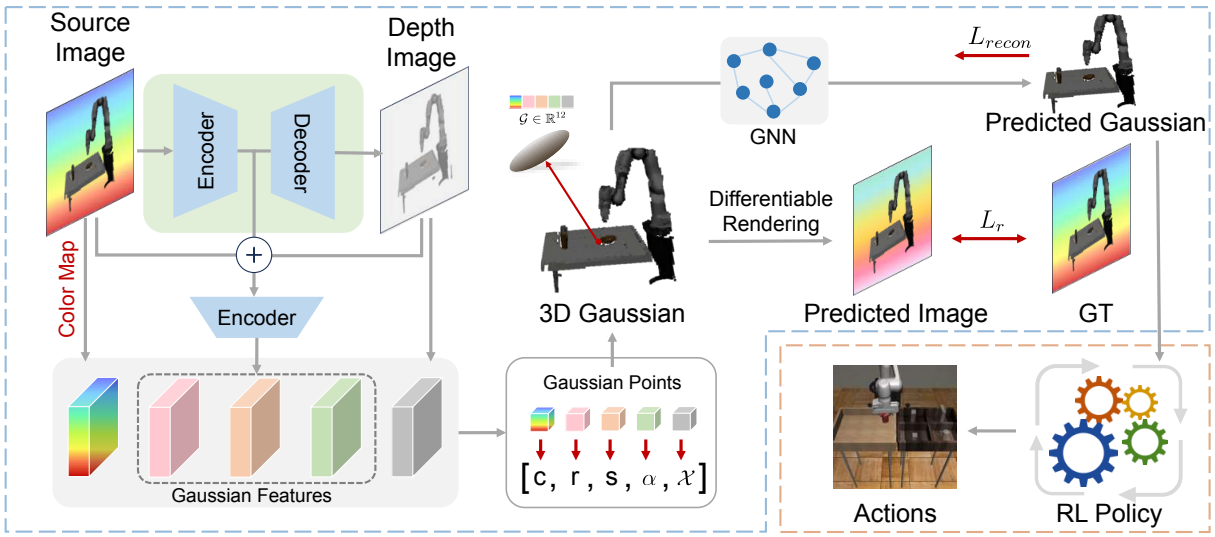

Fig. 1: The overview of the main pipeline. Contents in the blue dashed line represent the training of the generalizable Gaussian prediction module. This module converts image observation into a 3D-consistent and geometry-aware 3D Gaussian cloud. Contents in the orange dashed line denote the RL training module, which is fed with the reconstructed 3D Gaussians to predict the target actions.

1) To the best of our knowledge, we propose to adopt the pretrained generalizable 3D Gaussian as a representation for RL for the first time. Our approach is illustrated in Fig. 1. This innovation not only introduces an efficient way of environmental representation to the RL field but also broadens the application prospects of 3DGS technology in various RL tasks.
2) Our approach has demonstrated outstanding performance by validating our method across multiple tasks in the RoboMimic benchmark. This achievement not only proves the effectiveness of our proposed representation method but also provides new insights for future vision-based RL.

II. Related Work

A. 3D Scene Representation: Implicit and Explicit Way

Traditional methods that directly optimize explicit 3D geometries, such as mesh [28], voxel [29] and point cloud [30]. These representations use unified primitives to describe 3D structures. However, they cannot effectively describe detailed local geometries due to the limited resolutions. Recently the use of networks for implicitly representing scenes has prevailed [29], [31]. Among them, the Neural Radiance Field (NeRF) [18] attracts the most attention. Recent studies have employed NeRF for various purposes, including dynamic reconstruction [32], [33], physical reasoning [34], and reinforcement learning [23]–[25]. More recently, [26] proposed the 3D Gaussian Splatting to realize a remarkable rendering speed and high-quality 2D images by combining the advantages of both implicit and explicit representations.

B. Learning Representations for Reinforcement Learning

In the realm of RL, environments are typically described using intuitive, explicit methods, such as detailing the position [35]–[37], posture [38], [39], and motion state [4] of objects. However, these techniques have two significant challenges: first, in complex scenarios—such as complex shapes or soft materials—it becomes challenging to form structured representations; second, in real-world applications, these representations cannot be directly obtained from the physical world. With the progress in computer vision, researchers employ vision-based representations to train RL, such as [11]–[14]. However, 2D visions hardly reflect the real structures of objects because we live in the 3D world. Therefore, researchers investigate 3D-based representations including explicit and implicit ones. For the explicit ones, [15] train RL with voxel-based representation. [16] and [17] propose the benchmarks of point-based RL learning. [24], [25], and [23] propose to adopt implicit NeRF for scene modeling, encapsulating scene information within implicit vector features. Nonetheless, methods based on NeRF generally exhibit limited generalization to unseen scenes and usually require foreground masks.

3DGS [26] inherits from the point cloud, but with per-point geometry features to describe more detailed local structures. However traditional 3DGS does not match the requirement of RL. In this paper, we introduce a pretrained generalizable 3DGS pipeline for addressing this issue.

III. Preliminary

A. Reinforcement Learning

Our RL method is conceptualized within the framework of Markov Decision Process (MDP). This MDP is described by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, p, \gamma)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ signifies the action space, $\mathcal{R}$ represents the reward function, p delineates the transition probabilities from the current state to the subsequent state, and $\gamma \in [0, 1]$ stands for the discount factor applied to rewards. At each time step t, the agent interacts with the environment by receiving an observation.

Subsequently, the agent outputs an action $a_t \in \mathcal{A}$ based on a policy $\pi(a_t|s_t)$. According to the action, the state of the robot transitions from s_t to s_{t+1} based on the transition function $s_{t+1} \sim p(s_{t+1}|s_t, a_t)$. Additionally, the agent receives a reward $r_t = \mathcal{R}(s_t, a_t)$ at each time step. The objective is to maximize the return, achieved by optimizing the parameters θ of the policy:

$$\arg\max_{\theta} \mathbb{E}_{(s_t,a_t)\sim p_\theta(s_t,a_t)} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right] \tag{1}$$

where T denotes the time horizon of MDP.

B. 3D Gaussian Splatting

3DGS parameterize a 3D scene as 3D Gaussian primitives, each of primitives includes a mean (μ_k), a covariance ($\sum_k$), an opacity (o_k) and spherical harmonics coefficients ($\mathbf{SH}_k$) that represents the color information. These primitives can be rendered and accessed to produce novel views via Gaussian rasterization. To facilitate optimization by backpropagation, the covariance matrix can be decomposed into a rotation matrix (**R**) and a scaling matrix (**S**):

$$\Sigma = RSS^TR^T \tag{2}$$

The camera parameters and poses can be estimated via structure from motion such as Colmap, the projection of the 3D Gaussian to 2D image plane can be transformed by the view transformation (**W**) and the projection transformation. However, the projection matrix usually destroys the shape of Gaussian primitives due to its nonlinearity. To solve this issue, the Jacobian of the affine approximation **J** of the projective transformation is applied, as in:

$$\Sigma^{'} = JW\Sigma W^TJ^T \tag{3}$$

where the $\Sigma^{'}$ is the projected 2D covariance. After all Gaussians are transformed to the 2D planes, the final pixel color can be obtained by α-blend.

$$C = \sum_{i\in\mathcal{N}} c_i\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j) \tag{4}$$

in which c_i is computed from the spherical harmonics coefficients **SH**. α_i denotes the soft occupation Gaussian point i at 2D space, which can be calculated by $\alpha_i(x) = o_i exp(-\frac{1}{2}(x-\mu_i)^T\Sigma_i^T(x-\mu_i))$. In this work, we replace the **SH** with a single RGB color vector to facilitate simplicity.

IV. Method

Conventional 3DGS requires per-scene optimization from multiview images to adjust the Gaussian properties initialized randomly, therefore it can not be utilized as the state representation of RL. This is because it is impractical and extremely time-consuming to optimize a 3D Gaussian model at each moment of getting observations. Therefore, we propose to directly estimate 3D Gaussian clouds from multiview images in a generalizable way. The intuition is that the scenes and observations for a certain RL task are similar, therefore they share similar mappings from 2D images to 3D local geometries, and those local geometries can be seamlessly described by the properties of 3D Gaussians, i.e. the shape of each Gaussian point. Hence, the generalizable GS framework inherently learns the prior knowledge to map 2D patches to 3D local structures. In this case, RL algorithms can directly operate on this representation. The main pipeline of the method is demonstrated in Fig. 1.

This image-conditioned generalizable Gaussian representation has two-fold advantages. On the one hand, 3D Gaussian inherits the nature of the point cloud. Furthermore, the properties of 3D Gaussian allow more detailed descriptions of 3D local geometry, thus producing better geometry-aware feature representation. On the other hand, 3D Gaussians could be 3D-consistent because they are constructed from multiview images, thereby being robust to occlusions and generating 3D-consistent features. There are two steps in our framework. First, we train the image-conditioned 3DGS estimator network. This network can predict a 3D Gaussian cloud from given single or multiple images. In addition, we adopt a GNN network as a smoothness regularizer to enhance the 3D consistency. Second, we integrate and freeze the pretrained Gaussian estimator into the RL environment to convert the output observation of the environment into 3D Gaussians in real-time on which the RL policy is trained.

A. Generalizable GS framework

This section introduces the pretrained 3D Gaussian encoder. Given N images to describe a scene $\{I_n \in R^{H\times W\times 3}\}_1^N$ and their corresponding camera parameters $\{C_n = \{K_n, P_n\}\}_1^N$, we aim to reconstruct the 3D Gaussian representation conditioned by the given images, and the reconstructed Gaussian cloud can be accessed to render novel images with arbitrary viewpoints. We divide this image-conditioned Gaussian encoder into three main components: a depth estimator module, a Gaussian regressor module, and a Gaussian refinement module. In the training paradigm, when an image is randomly selected from the dataset (marked as I_t), we intentionally select the two closest nearby images of it (I_{s1} and I_{s2}). Then the two source views synthesize the target view used to optimize all networks via the difference with the I_t, which can be described as follows:

$$I_t^{'} = \mathcal{R}(\mathcal{G}(I_{s1}, I_{s2})|K_t, P_t) \tag{5}$$

where $\mathcal{R}$ denotes the rendering function of 3DGS, which is inspired by the original [40], $\mathcal{G}$ refers to the 3D Gaussian encoder. K_t, P_t are the camera intrinsic and pose of target view t, and $I_t{'}$ is the predicted target view via 3DGS. In the depth estimation module, we predict the absolute depth value for each pixel to transform a 2D image grid into a 3D coordinate space. The 3D coordinate of each point is regarded as the μ in the Gaussian properties. Then the Gaussian regressor predicts the rest of the Gaussian properties in a pixel-wise manner, which are transformed to 3D space accompanied by the depth value. Last, to improve the consistency of features, we define the Gaussian refinement operation to smooth the features in 3D space.

Depth Estimation. Estimating the depth map is crucial to bridge 2D images and 3D Gaussians. It is noted that the input of this module is a pair of stereo images. Therefore, depth prediction is equivalent to disparity prediction, and any alternative depth estimation theories can be adapted to this step. First, the two source views are fed to two extractors to extract semantic features.

$$F_{s1}, F_{s2} = Ex(I_{s1}), Ex(I_{s2}) \tag{6}$$

Ex refers to the feature extractor and the two networks share the same parameter. The extracted features are used to build cost volume by homography transformation [41]. The disparity value for each pixel is predicted through the cost volume. In practice, our model predicts the normalized log disparity map. The eventual absolute depth value can be converted from the prediction via:

$$D_{pred} = exp(D_{max} \cdot \sigma(\mathcal{D}(F_{s1}, F_{s2}))) \tag{7}$$

Here $\mathcal{D}$ refers to the disparity prediction network that we introduced above. σ is the Sigmoid activation. D_{max} indicates the maximum value of disparity across the dataset. We swap I_{s1} and I_{s2} in Eq. 7 to obtain all their per-pixel depth. Notably, we also train another depth predictor network with single image input because in some experiments it only allows one image as the observation, and the results show minimal discrepancy compared to the paired image input. When the camera set includes a global camera and a robot hand camera, we train two depth predictors for the two cameras separately.

Gaussian Properties Prediction. Each 3D Gaussian is parameterized by five independent properties to define its shape and appearance, i.e. $G = \{\mathbf{X}, \mathbf{R}, \mathbf{S}, \mathbf{c}, o\}$. As we stated before, the color attribute in conventional Gaussians is defined by spherical harmonics, but in this work, we replace it with the RGB vector to reutilize the image pixel color, thereby $\mathbf{c} = I_s$. Furthermore, we can obtain the $\mathbf{X}$ from the predicted depth map via:

$$\mathbf{X} = P_t \cdot D_{pred} \cdot K_t^{-1} \cdot \mathbf{u} \tag{8}$$

where $\mathbf{u} = \{u, v, 1\}$ is homogeneous 2D plane grid coordinates. The other symbols remain the same meaning as the previous content. Hence, this Gaussian regressor module aims to predict the rest properties in a per-pixel manner. We reutilize the extracted feature in the previous module and concatenate it with the source image and predicted depth to obtain a fused feature map.

$$\mathcal{F}_R = E_\phi(D_s \oplus F_s \oplus I_s) \tag{9}$$

in which E_ϕ is a UNet-like encoder parameterized by ϕ, $\mathcal{F}_R \in R^{H \times W \times D_R}$ denotes the fused feature map which is in the full image resolution. Then the F_R is sent to different prediction heads to regress corresponding Gaussian properties.

$$\mathbf{R},\ \mathbf{S},\ o = norm(\mathcal{H}_r(f_R)),\ exp(\mathcal{H}_s(f_R)),\ \sigma(\mathcal{H}_o(f_R)) \tag{10}$$

$\mathcal{H}_r$, $\mathcal{H}s$, and $\mathcal{H}_o$ represent the corresponding decoder heads for the Gaussian parameters, which are formulated by three full convolutional layers with 1×1 convolution kernel. The predicted parameter maps have the same spatial resolution as the source image, i.e. $\mathbf{R} \in R^{H \times W \times 4}, \mathbf{S} \in R^{H \times W \times 3}, \alpha \in R^{H \times W \times 1}$. Different functions activate each parameter map. $norm$ indicates the normalization function along the channel dimension. exp is the exponential activation.

Gaussian Refinement. After obtaining those properties, we can directly render novel views by Gaussian rasterization. However, due to inherent biases in the image, such as color biases related to different view directions, this might lead to some inconsistent noise. To address this issue, we adopt a GNN-based Autoencoder architecture to smooth these Gaussian properties and filter out inconsistent noises. This procedure is formulated as follows.

$$\mathbf{R}^{'}, \mathbf{S}^{'}, \mathbf{c}^{'}, o^{'} = D_\theta(E_\theta(\mathbf{R}, \mathbf{S}, \mathbf{c}, o|\mathbf{X})) \tag{11}$$

We implement the encoder E_θ and the decoder D_θ as graph-based networks, similar to [42]. We select the KNN neighbor number $K = 16$ to construct subgraphs. The encoder contains 3 MLPs and outputs the graph with 128-dimensional node features. The decoder also includes 3 layers to restore their respective original parameters. The GNN can be considered a regularization term to alleviate high-frequency noises caused by the discrepancy between neighboring views due to its smooth nature. After this operation, we obtain the smoothed 3D Gaussian representation which can be used to render the target view by Eq. 5

B. Training Strategy

We first pretrain the depth estimation module by using the $L1$ loss function. After the depth estimator converges sufficiently, we freeze it and train the Gaussian regressor and refinement module jointly. The following losses that guide the second training stage are formulated in the following.

$$\begin{aligned} L_r &= ||\mathcal{R}_{\theta,\phi}(I_{s1,2}|C_t) - I_t||_2^2 \\ L_{recon} &= ||D_\theta E_\theta(G) - G||_2^2 \\ L_{total} &= L_r + \lambda L_{recon} \end{aligned} \tag{12}$$

L_r denotes the rendering loss. $\mathcal{R}_{\theta,\phi}$ includes the Gaussian rendering and all learnable modules introduced above. I_t and C_t are the target image and its corresponding camera parameters. L_{recon} is an auxiliary loss to supervise the reconstruction of Eq. 11. λ refers to the coefficient to balance the two items and we experimentally set it to 0.15.

V. Experiments

We evaluate the proposed novel scene representation on the robomimic [27] robot learning platform on various environments and reinforcement learning methods. Robomimic is a framework that can learn RL policy from demonstrations. We select four tasks, namely Lift, Can, Square, and Transport, which are illustrated in Figure 2, and three Offline RL algorithms BCQ [43], IQL [44], and IRIS [45] with four different vision modalities including images, point clouds, voxels, and Gaussian points. All the demonstration data are claimed to be collected by a Proficient Human

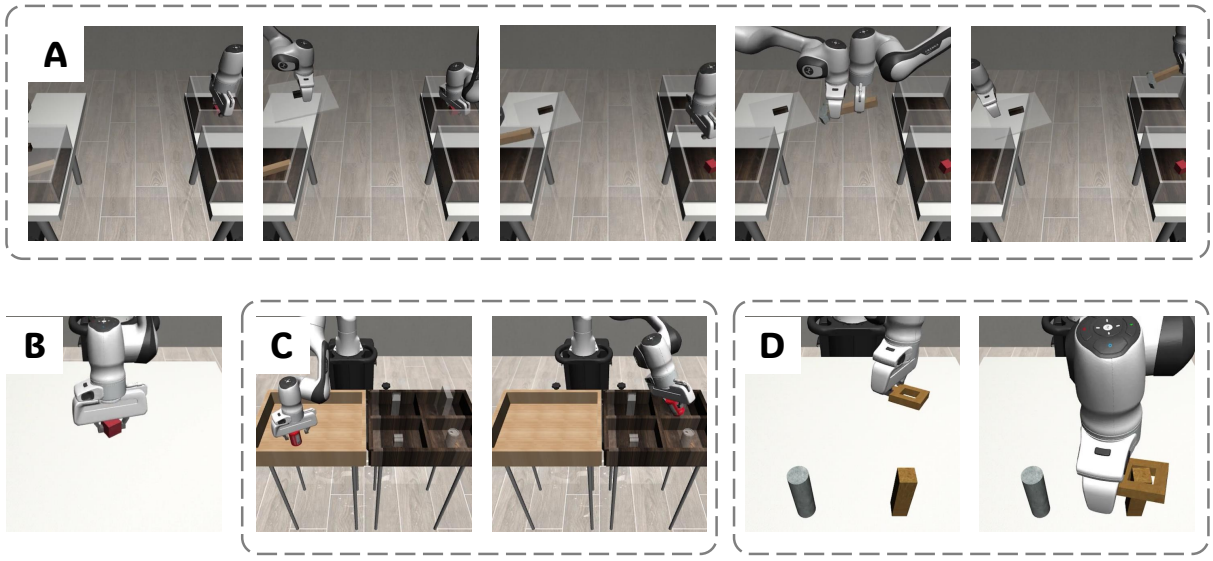

Fig. 2: We evaluate our method in four tasks. **A** is Transport, meaning two robot arms must take the red box into the target container and transport the hammer to the opposite collaboratively. **B** is Lift, which forces the arm to take the red box up. **C** is Can. In this task, the robot arm needs to take the can into a specific area showing the can symbol. **D** is Square task designed for placing the hollow square object with a handle on the square pillar.

Representation	Lift			Can			Square			Transport		
	BCQ	IQL	IRIS	BCQ	IQL	IRIS	BCQ	IQL	IRIS	BCQ	IQL	IRIS
Image	98.0±2.0	97.0±1.0	100.0±0.0	71.0±2.0	75.0±2.0	98.0±0.0	42.0±6.0	**4.0±2.0**	**70.0±3.0**	0.0±0.0	0.0±0.0	33.0±3.0
Point	97.5±1.5	**99.0±1.0**	100.0±0.0	68.5±4.5	75.0±1.0	97.5±1.5	28.0±4.0	0.0±0.0	58.5±10.5	0.0±0.0	0.0±0.0	25.0±5.0
Voxels	97.5±1.5	98.5±1.5	99.5±0.5	**71.5±0.5**	73.0±3	98.5±0.5	42.0±6.0	0.0±0.0	69.0±3.0	0.0±0.0	0.0±0.0	31.5±2.5
Gaussians(ours)	**98.5±1.5**	99.0±1.0	**100.0±0.0**	70.0±2.0	**76.5±1.5**	**98.5±0.5**	**48.5±3.5**	0.0±0.0	70.0±3.0	0.0±0.0	0.0±0.0	**36.0±2.0**

TABLE I: Quantitative Comparisons between different visual observation modalities on four tasks with three RL algorithms.

Sample number	2048	4096	8192
Lift	98.5	99.0	98.5
Can	61.0	68.0	70.0
Square	35.0	47.5	48.5

TABLE II: Quantitative Ablation of the effect of the number of 3D Gaussians on the performance of the BCQ algorithm.

operator with 200 successful trajectories. The generalizable Gaussian module is trained with some image-depth pairs and their associated camera parameters rendered by the robosuite simulator [46].

A. Experimental Settings

This work proposes the effectiveness of the novel scene representation on RL. Therefore, we compare the performance of different modalities on different RL algorithms. For each environment, we set several cameras to observe the scenario including the agentview camera and the robot hand and shoulder cameras. We set the baselines as the other three explicit modalities, namely multiview image, point cloud, and voxels for all algorithms. We exclude the implicit NeRF representation because it cannot be generalized to different scenes thereby needing to retrain the model on each task, and requires additional masks. The multiview images denote that we stack these images along the channel dimension and input to the RL network. We choose a simple ResNet18 as the encoder. The point cloud is produced by these ground-truth depth maps. Both points and 3DGS utilize the SetTransformer as the encoder. We set the voxel grid as $V \in R^{64\times3}$, and trilinearly interpolate the point clouds into it. We adopt a 3D convolutional Resnet to encode it. We set the batch size of voxel representation as 32, but that of the other modalities is 100. This is because 3D Convolution could consume a lot of memories. We train all models on an NVIDIA A6000 GPU with the Adam optimizer. It is noted that our goal is not to maximize the performance of each RL algorithm on specific tasks. In contrast, we aim to demonstrate the scene representation is general and effective. Therefore we do not search for the optimal hyperparameters for every task. As stated in Fig. 1, after the Generalizable Gaussian model is trained, we fixed it to be an encoder to transfer multiview image observations into the 3D-consistent and geometry-aware Gaussian representation. The RL network takes the Gaussians as input to predict proper actions and then obtain the next state. The new state produces new observations which will be iteratively transformed into the 3DGS representation in the next loop.

B. Results Analysis

We report all quantitative comparisons across different modalities and offline algorithms in Table I. The success rates

Training step	PSNR	Lift	Can	Square
3500	25.9	97.5	70.5	42.0
9500	28.8	97.5	71.0	43.0
15000	31.5	97.0	71.5	44.5
25000	33.7	98.5	70.0	48.5

TABLE III: Quantitative Ablation study to show the effect of 3DGS rendering quality on the performance.

	PSNR	SSIM	LPIPS
w/o feat_inp	31.62	0.979	0.081
w/o refine	32.24	0.971	0.062
prediction depth	33.69	0.985	0.041
real depth	33.91	0.987	0.038

TABLE IV: Ablations of some main components design.

are evaluated with randomly initialized shapes and positions in the online simulation environment. It is observed that our model delivers the most satisfactory results in most cases. In contrast, the cases in which the point cloud representation is only used show under-average performance compared with the other benchmarks. This indicates that these extra Gaussian properties improve the ability to represent the scene on account of detailed local geometry descriptions. In Lift, Square, and Transport tasks, our representation outperforms all counterparts across all three baseline algorithms. Our performance in the Square environment for BCQ method improves 15%, 70%, and 15% compared with the other two modalities. In the most difficult scenario Transport, BCQ and IQL algorithms fully collapse for all modalities due to the nature of those methods. But our method is 10%, 44%, and 15% better than the rest three representations in IRIS experiments. Besides, the covariance of our success rate is relatively smaller than others, which indicates that our method is more stable. We additionally evaluate the influence of the number of points on our performance. We downsample the original point set to 8192, 4096, and 2048 points respectively, and test them on the BCQ algorithm. The results are shown in Table II. The metrics in this Table are the average success rate. Notably, the proposed approach is overall not sensitive to the number of points. The performance is likely to be improved further as the point number increases, but the trend is not obvious. Interestingly, 4096 points performed just as well as 8192 points, and sometimes even better. On the contrary, when the number of points decreases to 2048, the simple task Lift is unaffected but the performance on relatively hard tasks, i.e. Can and Square, drops by 12% and 26% respectively. Therefore, this should be a trade-off to determine how many points are used in scene representation, and harder tasks naturally require more points to perceive. Furthermore, we test the effect of the quality of 3DGS reconstruction on RL performance. We stopped the GS model training to obtain models with different reconstruction qualities. Then we test each model on the BCQ algorithm and the report is listed in Table III. The rendering quality, depicted by PSNR, indeed affects the RL performance, especially on relatively harder tasks. More precise reconstruction leads to better RL performance.

According to this point, we, in addition, evaluate the effectiveness of some basic designs in the generalizable GS framework and report their results in Table IV. We use PSNR, SSIM [47], and vgg-based LPIPS [48] to evaluate the reconstruction quality. In this Table, w/o feat_inp means the regressor module does not reuse the pre-extracted features (F_{s1}, F_{s2} in Eq. 6), it replaces features with images. w/o refine denotes that we remove the Gaussian refinement module. "prediction depth" and "real depth" are the depth maps we use in subsequent modules because the Robomimic can also provide the real depth map. It can be seen that both cascaded architecture in feature space and the Gaussian refinement are effective in improving the reconstruction quality. Moreover, even though we replace the input depth map with the real one, there is only a minimal improvement, which indicates that the proposed approach can work well whether the observation includes depth information or not.

VI. Conclusion

We propose to use a novel scene representation i.e. 3DGS, for vision-based RL algorithms. Traditional 3DGS is not suited to be an environment representation due to the need for per-scene optimization. To address this, we introduce the learning framework that directly predicts 3D Gaussians from visual observations in a generalizable manner, which contains three main procedures, a depth estimator, a Gaussian regressor, and a Gaussian refinement module. This framework can effectively capture task-specific priors of image-to-local structures. To evaluate the effectiveness of the novel scene representation, we compare it with other explicit vision representations on the Robomimic platform with four different tasks and across three different RL algorithms. The results show that our generalizable GS representation overall outperforms these counterparts, and improves the Success Rate by 10%, 44%, and 15% respectively on the most challenging task.

References

[1] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel, "Deep spatial autoencoders for visuomotor learning," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 512–519.

[2] D. Dwibedi, J. Tompson, C. Lynch, and P. Sermanet, "Learning actionable representations from visual observations. in 2018 ieee/rsj international conference on intelligent robots and systems (iros)," 2018.

[3] M. Vecerik, J.-B. Regli, O. Sushkov, D. Barker, R. Pevceviciute, T. Rothörl, R. Hadsell, L. Agapito, and J. Scholz, "S3k: Self-supervised semantic keypoints for robotic manipulation via multi-view consistency," in *Conference on Robot Learning*. PMLR, 2021, pp. 449–460.

[4] R. Jonschkowski, R. Hafner, J. Scholz, and M. Riedmiller, "Pves: Position-velocity encoders for unsupervised learning of structured state representations," *arXiv preprint arXiv:1705.09805*, 2017.

[5] T. D. Kulkarni, A. Gupta, C. Ionescu, S. Borgeaud, M. Reynolds, A. Zisserman, and V. Mnih, "Unsupervised learning of object keypoints for perception and control," *Advances in neural information processing systems*, vol. 32, 2019.

[6] M. Laskin, A. Srinivas, and P. Abbeel, "Curl: Contrastive unsupervised representations for reinforcement learning," in *International conference on machine learning*. PMLR, 2020, pp. 5639–5650.

[7] L. Manuelli, Y. Li, P. Florence, and R. Tedrake, "Keypoints into the future: Self-supervised correspondence in model-based reinforcement learning," *arXiv preprint arXiv:2009.05085*, 2020.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[9] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[11] C. Sun, Y. Jia, Y. Guo, and Y. Wu, "Global-aware registration of less-overlap rgb-d scans," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6357–6366.

[12] Y. Gao, P. Zhou, Z. Liu, B. Han, and P. Hui, "Fras: Federated reinforcement learning empowered adaptive point cloud video streaming," *arXiv preprint arXiv:2207.07394*, 2022.

[13] J. Fan and W. Li, "Dribo: Robust deep reinforcement learning via multi-view information bottleneck," in *International Conference on Machine Learning*. PMLR, 2022, pp. 6074–6102.

[14] H. Yang, D. Shi, G. Xie, Y. Peng, Y. Zhang, Y. Yang, and S. Yang, "Self-supervised representations for multi-view reinforcement learning," in *The 38th Conference on Uncertainty in Artificial Intelligence*, 2022.

[15] Y. Ze, N. Hansen, Y. Chen, M. Jain, and X. Wang, "Visual reinforcement learning with self-supervised 3d representations," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2890–2897, 2023.

[16] Z. Ling, Y. Yao, X. Li, and H. Su, "On the efficacy of 3d point cloud reinforcement learning," *arXiv preprint arXiv:2306.06799*, 2023.

[17] Y. Qin, B. Huang, Z.-H. Yin, H. Su, and X. Wang, "Dexpoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 594–605.

[18] 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," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[19] J. Wang, Z. Zhang, and R. Xu, "Learning to generate and manipulate 3d radiance field by a hierarchical diffusion framework with clip latent," in *Computer Graphics Forum*, vol. 42, no. 7. Wiley Online Library, 2023, p. e14930.

[20] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Mip-nerf 360: Unbounded anti-aliased neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5470–5479.

[21] J. Wang, Z. Zhang, and R. Xu, "Learning robust generalizable radiance field with visibility and feature augmented point representation," *arXiv preprint arXiv:2401.14354*, 2024.

[22] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar, "Block-nerf: Scalable large scene neural view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8248–8258.

[23] D. Driess, I. Schubert, P. Florence, Y. Li, and M. Toussaint, "Reinforcement learning with neural radiance fields," *Advances in Neural Information Processing Systems*, vol. 35, pp. 16 931–16 945, 2022.

[24] Y. Ze, G. Yan, Y.-H. Wu, A. Macaluso, Y. Ge, J. Ye, N. Hansen, L. E. Li, and X. Wang, "Gnfactor: Multi-task real robot learning with generalizable neural feature fields," in *Conference on Robot Learning*. PMLR, 2023, pp. 284–301.

[25] D. Shim, S. Lee, and H. J. Kim, "Snerl: Semantic-aware neural radiance fields for reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2023, pp. 31 489–31 503.

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

[27] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," in *arXiv preprint arXiv:2108.03298*, 2021.

[28] S. Liu, T. Li, W. Chen, and H. Li, "A general differentiable mesh renderer for image-based 3D reasoning," vol. 44, no. 1, pp. 50–62, 2020.

[29] V. Sitzmann, J. Thies, F. Heide, M. Nießner, G. Wetzstein, and M. Zollhofer, "Deepvoxels: Learning persistent 3d feature embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2437–2446.

[30] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[31] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," vol. 33, pp. 7537–7547, 2020.

[32] Z. Yan, C. Li, and G. H. Lee, "Nerf-ds: Neural radiance fields for dynamic specular objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 8285–8295.

[33] A. Cao and J. Johnson, "Hexplane: A fast representation for dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 130–141.

[34] J. Wang, J. He, Z. Zhang, and R. Xu, "Physical priors augmented event-based 3d reconstruction," *arXiv preprint arXiv:2401.17121*, 2024.

[35] G. Du, K. Wang, S. Lian, and K. Zhao, "Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review," *Artificial Intelligence Review*, vol. 54, no. 3, pp. 1677–1734, 2021.

[36] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "Cosypose: Consistent multi-view multi-object 6d pose estimation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*. Springer, 2020, pp. 574–591.

[37] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox, "Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 13 438–13 444.

[38] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6d object pose estimation by iterative dense fusion," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 3343–3352.

[39] Y. He, H. Huang, H. Fan, Q. Chen, and J. Sun, "Ffb6d: A full flow bidirectional fusion network for 6d pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3003–3013.

[40] S. Zheng, B. Zhou, R. Shao, B. Liu, S. Zhang, L. Nie, and Y. Liu, "Gps-gaussian: Generalizable pixel-wise 3d gaussian splatting for real-time human novel view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 680–19 690.

[41] 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)*, 2018, pp. 767–783.

[42] Y. Yang, C. Feng, Y. Shen, and D. Tian, "Foldingnet: Point cloud auto-encoder via deep grid deformation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 206–215.

[43] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *International conference on machine learning*. PMLR, 2019, pp. 2052–2062.

[44] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," *arXiv preprint arXiv:2110.06169*, 2021.

[45] A. Mandlekar, F. Ramos, B. Boots, S. Savarese, L. Fei-Fei, A. Garg, and D. Fox, "Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4414–4420.

[46] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu, "robosuite: A modular simulation framework and benchmark for robot learning," *arXiv preprint arXiv:2009.12293*, 2020.

[47] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[48] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.