3D Superquadric Splatting

Daniel MacSwayne Aleš Leonardis Jianbo Jiao
The MIx Group, School of Computer Science, University of Birmingham
dxm378@student.bham.ac.uk, {a.leonardis, j.jiao}@bham.ac.uk

Abstract

Gaussian Splatting has proven to be an effective algorithm for novel view synthesis and 3D reconstruction from multi-view images. However, the underlying volumetric primitive—the ellipsoidal Gaussian—has limited expressive capabilities, leading to difficulties in 3D modelling (especially geometry such as edges, corners, and high curvature). To address this limitation, in this paper, we introduce Superquadric Splats (SQS), an extended class of volumetric primitives, as a superset of Gaussian Splats, to model more detailed geometry. We treat superquadrics as volumetric distance functions rather than level-set surfaces. A non-trivial differentiable rendering pipeline is developed to support this. Extensive experimental analysis on multiple datasets validates the effectiveness of the proposed SQS approach, showing both enhanced visual and geometric performance compared to Gaussian-based splatting (with more than 1dB in PSNR and prominent geometric improvement). Project page can be found at: https://daniel-macswayne.github.io/3DSQS/

1. Introduction

The problem of Novel View Synthesis (NVS) is commonly defined as follows: For a static scene, given a set of images from posed cameras, render an image of the scene from a previously unseen view (*i.e.* camera pose). Performing well on this task necessitates constructing a 3D model of the scene. This is a challenging task because depth information is irreversibly lost when geometry is projected onto an image plane. The ability to reconstruct 3D scenes from visual inputs is immensely useful and has applications in robotics and spatial computing [35, 39, 40].

There are various ways to reconstruct 3D scenes from visual inputs, ranging from conventional techniques such as stereo vision [15, 19, 21], structure from motion [32, 36], multi-view stereo [7, 33], *etc.* in earlier years, as well as neural radiance fields (NeRF) [27] and 3D Gaussian Splatting (3DGS) [22] in more recent literature. As a neural representation, NeRF showed a novel way to represent 3D via a

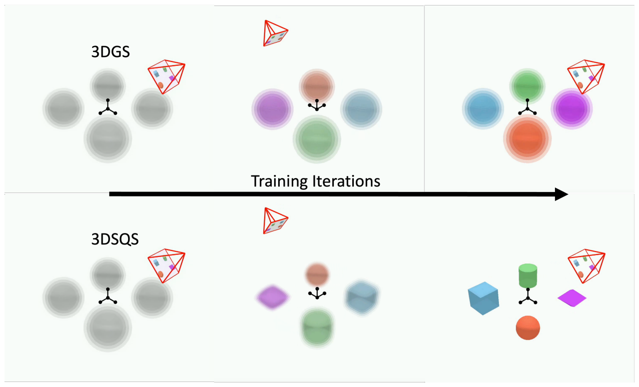

Figure 1. **Illustration of conventional Gaussian splats (3DGS) *vs.* the proposed Superquadric Splats (3DSQS).** A few 3D shapes (*e.g.* cube, cylinder, sphere, and pyramid) are modelled by regular 3DGS (**top**) parametrised by Eq. (4), and the proposed 3DSQS (**bottom**) parametrised by Eq. (6) (corresponding dynamic videos are shown on the project page.

deep neural network, allowing high-quality novel view synthesis. However, NeRF suffers from computationally costly training and rendering, while faster NeRF methods trade off speed for quality. As an alternative 3D modelling approach, 3DGS [22] has recently been shown to be an effective NVS solution. By modelling a scene as a composition of parameterised volumetric primitives, a visually accurate model can be reconstructed, in a computationally efficient way—addressing the above-mentioned issues with NeRF.

However, even with the promising performance, there are still weaknesses. We noticed a fundamental deficiency of 3DGS in 3D modelling by nature, which is its basic constructing element—*Gaussian*. Such Gaussian-based underlying 3D representation has a limited expressive capability by its definition, making high curvature (*e.g.* edges and corners) modelling inefficient. Usually, we need a large number of Gaussians to approximate shapes like boxes, while still struggling with the accuracy around edges. An example is shown in Fig. 2, from which we can see that although a Gaussian splat is able to represent a circle, it struggles to reconstruct various other shapes (*e.g.* square, polygon, dia-

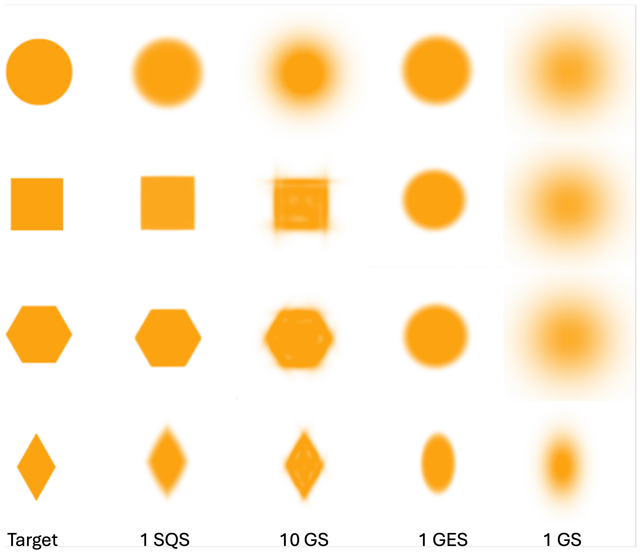

Figure 2. **The ability of each primitive** to adjust its parameters in order to match a visual target. **Left to right**: Target shape, One Superquadric splat (SQS), Ten regular Gaussian splats, One generalised exponential splat (GES) [11], One regular Gaussian splat [22]. It can be seen that the proposed SQS has a much higher expressive capacity than regular Gaussian splats (3DGS).

mond), even with multiple splats. This is due to the limitations that come from its fundamental definition of ellipsoid-shaped 3D Gaussian. To this end, we are interested in asking: *is there a more appropriate 3D primitive representation that could fit a more general family of shapes by nature?*

To answer this question, in this paper, we propose, to our knowledge, the first attempt towards this direction, and find a neat representation for 3D modelling, better fitting more general shapes compared to 3DGS by its fundamental definition. Specifically, we revisit an established primitive — **superquadrics**, and introduce *Superquadric Splat (3DSQS)*, an alternative volumetric primitive which can model an extended family of shapes. An illustration of 3DSQS versus regular 3DGS is shown in Fig. 1. Superquadrics, by its definition [2], is a parametric family of shapes that can represent cubes, cylinders, spheres, ellipsoids *etc*. in a single continuous parameter space. With its simple yet effective representation ability, we can easily fit various shapes of objects (or parts of the objects), with a smaller number of primitives than Gaussians. As illustrated in Fig. 2, we show that a single Gaussian fails to approximate a target square (and other shapes) and even with an increased number of Gaussians, it still struggles to model the shape. Whereas for superquadrics, only one is needed to model the shape very well. This stems from the fact that superquadrics are parametrically defined to represent the shape in a more precise way than having to approximate it as in Gaussians. The main contributions of this work are summarised as follows:

- We present Superquadrics Splat (SQS), a new approach to represent 3D using a generalised family of atomic shapes. This extended volumetric primitive (*i.e*. SQS) is better suited to model geometry, which is a crucial perspective in 3D modelling.
- A non-trivial differentiable rendering pipeline is proposed to support the superquadric projections. The proposed SQS can easily serve as a plug-and-play component for existing 3DGS-based pipelines.
- Extensive experimental analysis validates the effectiveness of the proposed SQS, evidenced by state-of-the-art performance for both visual reconstruction ($\sim$ 1dB improvement on average) and geometric reconstruction (5-10% improvement on average).

2. Related Work

2.1. Neural Radiance Fields

Neural Radiance Fields (NeRF) [27] were a significant advancement for NVS. In NeRF, the entire scene is represented with a single function that assigns each point in the scene an opacity and a view-dependent colour. This function is parameterised by an MLP which is trained independently to fit each new scene. NeRF is trained using a self-supervised framework whereby the model attempts to synthesise a realistic render from a viewpoint that is shared by one of the images in the training set. The comparison between the training image and the synthetic render acts as the supervisory signal that is back-propagated to train the MLP. This only works because the entire rendering pipeline is differentiable. During rendering, the radiance field is sampled many times along virtual rays to form an image. This is an expensive procedure due to the many MLP forward passes. It is often wasteful because it can involve sampling empty regions which do not contribute to the final image. Follow-up works aimed to sample the radiance field more efficiently and reduce the size of the MLP [29–31, 34].

2.2. Gaussian Splatting and the Variants

Different from NeRF, a Gaussian-based representation has increasingly been leveraged in recent advances. The seminal work [22] introduced 3D Gaussian Splatting (3DGS), enabling real-time neural rendering with high visual fidelity. Building upon this, several methods extend or adapt Gaussian-based techniques for improved geometric accuracy and broader applications. For instance, Huang et al. [16] explored 2D Gaussian Splatting for geometrically accurate radiance fields. In pursuit of richer shape modelling, concurrent work [8] fuses 2D Gaussians with superquadrics to learn part-aware 3D representations. Beyond Gaussian, [4] introduced linear kernel splatting for faster and high-fidelity rendering, while [14] represented radiance fields using smooth convex primitives. To enhance surface detail, [42] proposed quadratic Gaussian splatting, while [25] combined explicit Gaussians with implicit signed distance fields. Orthogonal to these, [10] presented a Lagrangian volumetric mesh-based representation via Tetsphere Splitting. Recent innovations also include LinPrim [37], which uses linear primitives for differentiable volumetric rendering, and deformable beta splatting [26], which generalises Gaussian splatting with learnable Beta kernels for improved deformation modelling.

3. Preliminaries

3.1. Gaussian Splatting

In Gaussian Splatting [22], each splat has its own parameterised radiance field modelled with an ellipsoidal radial distance function d. It provides the characteristic shape of the volumetric primitive. This distance is fed into an exponential decay to get the Gaussian weighting. The distance field is parameterised by the position $\vec{m}$ and the covariance matrix Σ that encodes scale S and orientation R:

$$\Sigma = RSS^TR^T \tag{1}$$

$$d = (\vec{x} - \vec{m})^T\Sigma^{-1}(\vec{x} - \vec{m}) \tag{2}$$

$$G = e^{-\frac{1}{2}d}. \tag{3}$$

The view-dependent colour is modelled with a spherical harmonic series. The coefficients that describe the shape, orientation, position, opacity and colour of each splat are learned during the training process. During rendering, the covariance matrix is rotated and projected from the 3D world coordinates to the 2D image coordinates using an efficient projection method from [43].

There are two main advantages of a parametric particle-based radiance field. Firstly, the radiance field is sampled only at the known particle locations, thus empty regions with no splats are automatically avoided. Secondly, the opacity and view-dependent colour can be analytically evaluated rather than an expensive MLP forward pass. This facilitates efficient opacity and colour accumulation when forming an image.

3.2. Superquadrics

Quadrics are a family of second-degree polynomials that generalise the ellipse to the higher-dimensional ellipsoid that characterises 3D Gaussian Splatting. In simplified form, the distance function can be expressed as:

$$d(\vec{x}_s) = \left(\frac{x_s}{a_x}\right)^2 + \left(\frac{y_s}{a_y}\right)^2 + \left(\frac{z_s}{a_z}\right)^2, \tag{4}$$

where d is a radial distance function. An ellipsoid's surface is implicitly defined by a level set of d. The parameters a_x, a_y, a_z scale the ellipsoid along its principal axes.

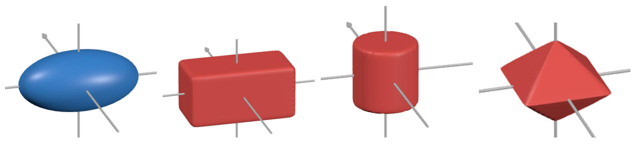
Figure 3. Different forms of superquadrics including ellipsoids (the left blue one) *i.e.* Gaussian, and others (red ones on the right).

By adding two exponential terms ϵ_1, ϵ_2 into the radial distance function, the shape family is extended to also include cuboids, cylinders and more. Thus superquadrics are a *superset* of regular quadrics. See Fig. 3 for some examples of the various forms a superquadric can take. Specifically, the superquadric distance function takes the form:

$$d(\vec{x}_s) = \left[\left(\frac{x_s}{a_x}\right)^{\frac{2}{\epsilon_2}} + \left(\frac{y_s}{a_y}\right)^{\frac{2}{\epsilon_2}}\right]^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{z_s}{a_z}\right)^{\frac{2}{\epsilon_1}}. \tag{5}$$

Superquadrics have been a focus of 3D computer vision for decades and first lntroduced in [2]. In particular, the ability to segment a scene into its composite superquadrics has been extensively studied [1, 28]. However, these works attempt to model the surface of an object with a specific level-set instance of the implicit superquadric distance function. Instead, we propose to use the entire superquadric distance field to model the volume density of the object.

Other attempts appear in a recent work [11] at modifying the volumetric primitive by adding a single exponential term to the ellipsoid distance field. This extra parameter varies the attenuation of the distance function so that the Gaussians can more easily model high-frequency features such as high-contrast edges. However, it is still based on the ellipsoidal distance function.

4. The Proposed Approach

4.1. Superquadrics as a Volumetric Primitive

Our approach utilises the same self-supervised training framework popularised by NeRF. We adopt the same volumetric particle-based splat rendering from Gaussian Splatting. However, we introduce *superquadrics* as the new volumetric primitive due to their increased expressive capabilities. We also borrow the third exponential, ϵ_3, term used in Generalised Exponential Splatting [11]. It is hoped that this will allow the volumetric primitives to more naturally represent common geometric features such as high contrast edges by amplifying the distance function. For the ablation study in Sec. 5.4, we refer to SQS (Base) and SQS (Full) defined by whether they include the ϵ_3 term (please refer to Sec. 3.4 in the supplementary material for clarity). In the following section, we will elaborate on the new differentiable rendering pipeline that is required to support this.

The distance function for exponential superquadrics is defined as:

$$d(\vec{x}_s) = \left[\left[\left(\frac{x_s}{a_x} \right)^{\frac{2}{\epsilon_2}} + \left(\frac{y_s}{a_y} \right)^{\frac{2}{\epsilon_2}} \right]^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{z_s}{a_z} \right)^{\frac{2}{\epsilon_1}} \right]^{\epsilon_3}. \quad (6)$$

4.2. Coordinate Systems

First, we introduce notation to clarify which coordinate systems are being discussed. The *Shape* coordinate system has its origin at the centre of the volumetric shape primitive, and its axes are aligned with the principal axes of the shape. The *World* coordinate system refers to the shared space that all shape primitives and cameras inhabit. The *Camera* coordinate system has its origin at the focal point of the camera, and the z axis is aligned with the viewing direction. The *Image* coordinate system is the result of a perspective projection applied to the camera space. The *Pixel* coordinate system describes the same space but measured in pixels instead. The spaces are denoted by s, w, c, i, p respectively. For example, the matrix $R_{w\to s}$ rotates points from world space to shape space.

4.3. Challenges in Rendering

To reiterate, rendering a Gaussian splat requires transforming the 3D ellipsoidal distance function into a 2D projected version. By doing this, the 2D distance function d' can be evaluated at each pixel position $\vec{x}_p$. To achieve our proposed SQS goal, we need to extend this operation to superquadrics. However, this is non-trivial because merely augmenting the previous transformation is not viable analytically. Therefore, the need to formulate an alternate rendering procedure for superquadrics is crucial. Instead, the new method evaluates the 3D superquadric distance function by first recovering the missing 3D information for the pixel canvas.

4.4. Rim Projection

To render a 3D shape, we must project its shadow onto an image canvas. For example, a cube viewed diagonally has a hexagonal shadow. The contour of this shadow is called the rim. The rim of the shape describes the boundary between the visible and occluded sections when viewed from a certain direction. An illustration is shown in Fig. 4. For simplicity, we use the orthographic rim as an approximation of the perspective rim. This is defined as the set of points where the surface normal is perpendicular to the viewing direction [18], which provides an analytic rim constraint:

$$\tfrac{r_{13}}{a_x} \cos^{2-\epsilon_1} \eta \cos^{2-\epsilon_2} \omega + \tfrac{r_{23}}{a_y} \cos^{2-\epsilon_1} \eta \sin^{2-\epsilon_2} \omega + \tfrac{r_{33}}{a_z} \sin^{2-\epsilon_1} \eta = 0. \quad (7)$$

In this case, the viewing vector is the z axis of the camera space expressed in the shape coordinate system. Thus r_{13}, r_{23}, r_{33} are components of the rotation matrix $R_{c\to s}$.

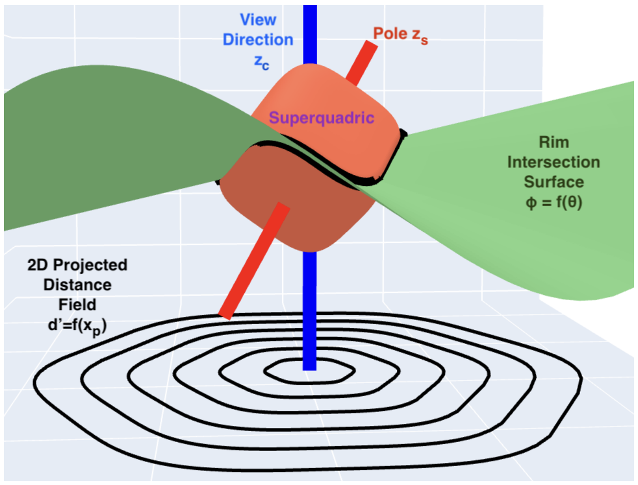

Figure 4. **A diagram to show the projection of a superquadric (red) using the rim intersection surface (green).** The 2D hexagonal distance function (black) is represented using isocontours.

Note, the rim constraint is radially equivariant, meaning that the level set that it defines, scales the entire radial distance field. This defines a surface with the polar form $f(\omega, \eta) = 0$ that intersects the superquadric along the rim. For each pixel, we are aiming to find its corresponding position on this surface. This rim constraint solves an explicit relation:

$$\eta = \arctan \left[\left(-\tfrac{a_z}{r_{33}} \left(\tfrac{r_{13}}{a_x} \cos^{2-\epsilon_2} \omega + \tfrac{r_{23}}{a_y} \sin^{2-\epsilon_2} \omega \right) \right)^{\frac{1}{2-\epsilon_1}} \right]. \quad (8)$$

The rim is sampled uniformly to generate a set of (ω, η) tuples. Note, sampling ω uniformly causes the η values to non-uniformly cluster at the corners. For the edge cases where $r_{33} \to 0$, the degenerate solutions are used instead (please see Sec. 1.1 in the supplementary material). The rim points are converted to Cartesian shape coordinates $\vec{r}_s$ and are then rotated to align with the camera coordinate system:

$$\vec{r}_s = \begin{bmatrix} a_x \cos^{\epsilon_1} \eta \cos^{\epsilon_2} \omega \\ a_y \cos^{\epsilon_1} \eta \sin^{\epsilon_2} \omega \\ a_z \sin^{\epsilon_1} \eta \end{bmatrix} \quad (9)$$

$$\vec{r}_c = \begin{bmatrix} r_{xc} \\ r_{yc} \\ r_{zc} \end{bmatrix} = R_{s\to c} \cdot \vec{r}_s. \quad (10)$$

Finally, the rim is expressed in a polar form, which is centred at the shape origin and aligned with the camera axes:

$$\theta_r = \arctan 2(r_{yc}, r_{xc}) \quad (11)$$

$$\phi_r = \arctan \left(\frac{r_{zc}}{\sqrt{r_{xc}^2 + r_{yc}^2}} \right), \quad (12)$$

where θ is the longitude around the z axis of the camera coordinates and ϕ is the latitude above the plane.

Discretising the rim is only necessary because there is no known analytic solution $\phi = f(\theta)$. If this existed, then the 2D projection of a superquadric could be derived entirely analytically. Instead, we use the set of tuples (θ_r, ϕ_r) as representative samples relating ϕ and θ.

4.5. Pixel Canvas

In this section, we will introduce how we use the rim to recover the 3D points $\vec{x}_s$, $\vec{x}_c$ from a 2D-pixel canvas point $\vec{x}_p$. First, we convert back to image space by re-scaling the pixel canvas by the focal length f (measured in pixels per meter) and shifting by the focal centre $\vec{c}_p$:

$$\vec{x}_i = \frac{1}{f}(\vec{x}_p - \vec{c}_p). \tag{13}$$

Next, the image canvas is re-scaled to undo the perspective projection of the splat. Hence, we multiply by the known view-depth m_{zc} of the splat:

$$\vec{x}_c = m_{zc} \cdot \vec{x}_i. \tag{14}$$

The canvas is then converted to polar coordinates that are centred at the splats lateral position m_{xc}, m_{yc}. Note, with only 2D canvas information, it is impossible to calculate the ϕ component unless there is additional information about the shape. Fortunately, the rim points can be leveraged. For each pixel at angle $\theta = \arctan 2(y_c - m_{xc}, x_c - m_{xc})$, we can find the two adjacent θ_r values from the rim mesh and interpolate the missing ϕ values:

$$\phi = Interpolate(\theta_r, \phi_r, \theta). \tag{15}$$

However, directly interpolating the polar values leads to visual artefacts when the rim points are unevenly sampled. To resolve this, it is better to instead linearly interpolate between the Cartesian points. As the rim intersection surface is radially equivariant, we can use similar triangles to recover the missing 3D information ϕ and z_c for each pixel point:

$$\widehat{r}_c = Interpolate(\theta_r, \vec{r}_c, \theta) \tag{16}$$

$$\tan\phi = \frac{z_c}{\sqrt{x_c^2 + y_c^2}} = \frac{\widehat{r}_{zc}}{\sqrt{\widehat{r}_{xc}^2 + \widehat{r}_{yc}^2}}. \tag{17}$$

Finally, the restored $\vec{x}_c$ point is converted to shape coordinates $\vec{x}_s$ where it is used to evaluate the 3D superquadric distance field and find the Gaussian weighting G:

$$\vec{x}_s = R_{c\to s} \cdot \vec{x}_c \tag{18}$$

$$G = e^{-\frac{1}{2}d(\vec{x}_s)}. \tag{19}$$

To summarise, we have constructed a Gaussian where the underlying distance function is now characterised by a superquadric. By leveraging the rim intersection surface, we showed how the distance function could be evaluated for pixels on a 2D image canvas. The alpha blending and colouring process is left unchanged as in the original 3DGS.

5. Experiments

5.1. Experimental Setup

Implementation details. Following recent work [6], in this experiment section, DUSt3R [38] is used to simultaneously estimate a depth map and the pose of each sparse viewpoint, thus producing an initial global point cloud. During rendering, the image canvas is split into 16×16 tiles. For each tile, the splats are first culled based on their visibility. The rim is sampled using 30 points per splat. The training is supervised by a weighted combination of RGB L1 loss and SSIM. There are a fixed number of splats per scene, and there is no adaptive density control. Each experiment is run with 4×10^5 splats for 1,000 training iterations using a learning rate of 10^{-3}. Code and models are available on the project page. We created an efficient PyTorch renderer that follows the same tile-based parallelisation and early stopping logic as 3DGS [22]. This runs on both CPU and GPU. For a typical render, the memory cost is 1.1GB and the render takes 10s. Since splats are loaded in depth-ordered batches, the memory cost is fixed regardless of the number of splats in view. However, render time is proportional to the number of splats stacked per pixel. However, these pixels will likely saturate and terminate early. See section 2 in the supplementary material for more details. All experiments were performed based on a high-performance computing cluster equipped with NVIDIA A100 GPUs. Each training job was run on a compute node with 40GB of memory and 36 cores.

Datasets. To evaluate the proposed SQS, we use both synthetic and real-world data. We opted to use synthetic datasets because the ground truth depth map for each image can be accurately obtained. This allows us to measure the geometric quality through the estimated depth map from the model, to establish a more reliable evaluation pipeline. Note that although we have access to the ground truth depth map, this is not used as a supervisory signal. Specifically, the synthetic datasets we used include: Chair [12] and NeRDS360 [17]. On the other hand, real-world benchmark datasets were also included for evaluation: 12 scenes from Deep Blending [13], 6 scenes from Tanks&Temples [23], and 8 scenes from the MipNeRF360 dataset [3].

Metrics. The visual reconstruction performance is measured by the evaluation metrics of SSIM, PSNR, and LPIPS [41] on the rendered RGB images in the test set. However, the quality of geometric reconstruction is measured using evaluation metrics based on the depth map. Specifically, we use the mean absolute relative error (rel), mean $\log_{10}$ error (log10), root mean squared error (rms), rms(log), and the accuracy under threshold ($\delta < 1.25^i, i = 1, 2, 3$), following the depth estimation literature [5, 9, 20, 24]. Note that we

Table 1. **Quantitative performance comparison for novel view synthesis across real-world datasets**: DeepBlending [13], MipNeRF360 [3], and Tanks&Temples [23]. The proposed SQS is compared against other alternative rendering approaches. SQS is shown to achieve the best performance for all metrics over the three standard real-world benchmark datasets. The best performances are shown in **bold** and the second best is <u>underlined</u>.

	DeepBlending			MipNeRF360			Tanks&Temples		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
3DGS [22]	21.66	0.759	0.170	21.79	0.739	0.167	25.88	0.826	0.147
GES [11]	21.78	0.765	0.172	22.20	0.772	0.164	26.19	0.830	0.145
SQS Base (Ours)	<u>22.17</u>	<u>0.785</u>	<u>0.160</u>	<u>22.39</u>	<u>0.773</u>	<u>0.161</u>	<u>26.34</u>	<u>0.836</u>	<u>0.139</u>
SQS Full (Ours)	**22.54**	**0.797**	**0.152**	**22.85**	**0.786**	**0.153**	**26.60**	**0.843**	**0.138**

Table 2. **Quantitative performance on the Chair dataset** [12] for both novel view synthesis (left) and geometry (right).

	Novel View Synthesis			Geometric Quality						
Method	PSNR↑	SSIM↑	LPIPS↓	rel↓	log10↓	rms↓	rms(log)↓	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$
3DGS [22]	32.00	0.9289	0.0662	0.057	0.027	0.239	0.151	0.969	0.985	0.991
GES [11]	<u>32.48</u>	**0.9341**	<u>0.0548</u>	0.054	0.026	0.234	0.149	0.974	0.988	0.991
SQS Base (Ours)	**32.94**	<u>0.9336</u>	0.0555	0.054	0.026	0.233	0.149	**0.974**	**0.988**	0.991
SQS Full (Ours)	32.35	0.9318	**0.0547**	**0.049**	**0.023**	**0.222**	**0.132**	0.973	0.987	**0.992**

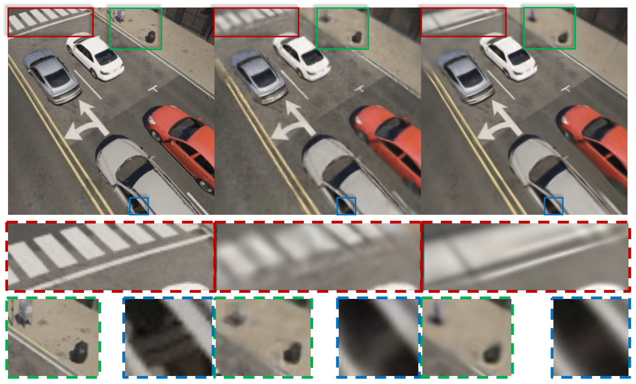

Figure 5. **Qualitative comparison on NeRDS360 [17]**. From left to right: Ground Truth, our SQS, and 3DGS. Example regions are zoomed in for moreldetailed comparison, *e.g.* the zebra crossing and other structures are better modelled with the proposed SQS.

are aiming to measure the geometry reconstruction error. It is unreasonable to test depth reconstruction on far-field with no discernible multi-view discrepancy. Therefore, we only measure the depth reconstruction accuracy for pixels within the near-field mask. For outside open-ended scenes, the near-field threshold is set to 24m. As there is no ground-truth depth for real-world scenes, quantitative measurement was not applied for real-world data, but qualitative comparisons are presented.

5.2. Novel View Synthesis Performance

Here in this section, we evaluate the performance of the proposed SQS against the conventional 3DGS approach and state-of-the-art technique GES [11], for the task of novel view synthesis. Tab. 1 shows the quantitative performance comparison across three real-world benchmark datasets, while Tab. 2 and Tab. 3 present the quantitative comparisons on the above-mentioned synthetic datasets.

From the results, we can see that the proposed superquadrics-based representation is a better primitive than Gaussian splats. This supports the aforementioned (Sec. 4) limited expressivity of Gaussian splats, which are unable to reconstruct certain visual/geometric features efficiently. Fig. 2 shows how a single Gaussian struggles in modelling high-contrast features with sharp corners. Even when 10 Gaussians are used, they are still unable to model this geometry without producing undesirable artefacts. However, with just 1 single superquadric, the artefacts are greatly diminished, and the target shapes are very well reconstructed. A recent work [11] managed to push the 3DGS boundary to be sharper, but when representing non-ellipsoid shapes, it fails to represent them due to the limited expressivity by definition. Examples of more realistic scenes are shown in Fig. 5, in which the effectiveness of SQS over 3DGS is further validated. In addition to synthetic data, we also present the performance of SQS on real-world scenes with comparison to the 3DGS [22] and GES [11], in Fig. 6 (more video examples are included in the supplementary material). We can see that in real scenes, the advantage of the proposed SQS over 3DGS further supports its effectiveness in reconstructing better structures, *e.g.* boundary of the letters, textures on the wall, and the door-lock details.

Table 3. **Quantitative performance on the NeRDS360 dataset** [17] for both novel view synthesis (left) and geometry (right) datasets.

Method	Novel View Synthesis PSNR↑	SSIM↑	LPIPS↓	Geometric Quality rel↓	log10↓	rms↓	rms(log)↓	$\delta < 1.25 \uparrow$	$\delta < 1.25^2 \uparrow$	$\delta < 1.25^3 \uparrow$
3DGS [22]	22.88	0.7981	0.2288	0.248	0.116	2.664	0.334	0.462	0.862	0.964
GES [11]	23.12	0.8067	0.2075	0.244	0.114	2.660	**0.328**	0.479	**0.871**	**0.966**
SQS (Base)	**23.14**	<u>0.8093</u>	<u>0.2030</u>	**0.230**	**0.113**	**2.627**	0.341	**0.509**	0.870	0.963
SQS (Full)	<u>23.13</u>	**0.8104**	**0.2008**	0.233	0.115	2.671	0.351	0.501	0.860	0.959

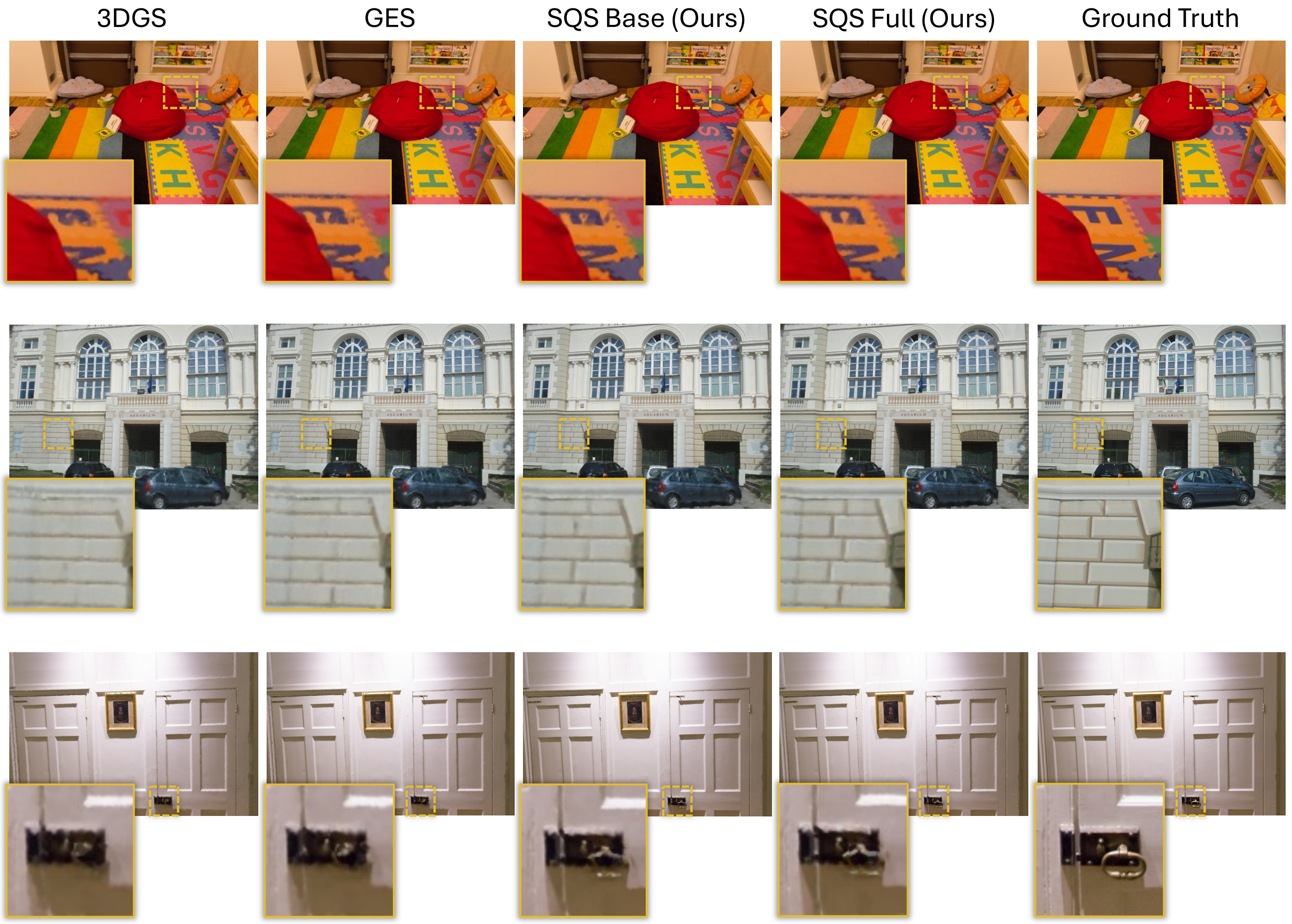

Figure 6. **Qualitative performance of novel view synthesis** on the proposed method (SQS Base and Full) compared with 3DGS [22], GES [11], together with the Ground Truth, across various real-world scenes. Clear structures (*e.g.* letters, bricks, and door-lock) are better reconstructed by the proposed SQS when compared to other methods.

5.3. Geometric Reconstruction Performance

Unlike existing NeRF-based and 3DGS-based literature that only evaluate on novel view synthesis, *i.e.* the visual reconstruction quality, of different 3D modelling methods, here in this paper, we argue that the quality of geometry reconstruction is crucial for various 3D applications, and evaluate the geometric reconstruction quantitatively and qualitatively. Specifically, we use the estimated depth as a geometry representation, and assess the quality according to the evaluation metrics mentioned in Sec. 5.1. The corresponding results are reported in Table 2 and Table 3 (right side). We can see that the proposed SQS recovers better geometric information when compared to other popular 3D reconstruction methods. Qualitative examples of rendered depth on real-world scenes are shown in Fig. 7, where we can see the advantages of our SQS over other methods, *e.g.* edge regions and the cup/bowl on the table.

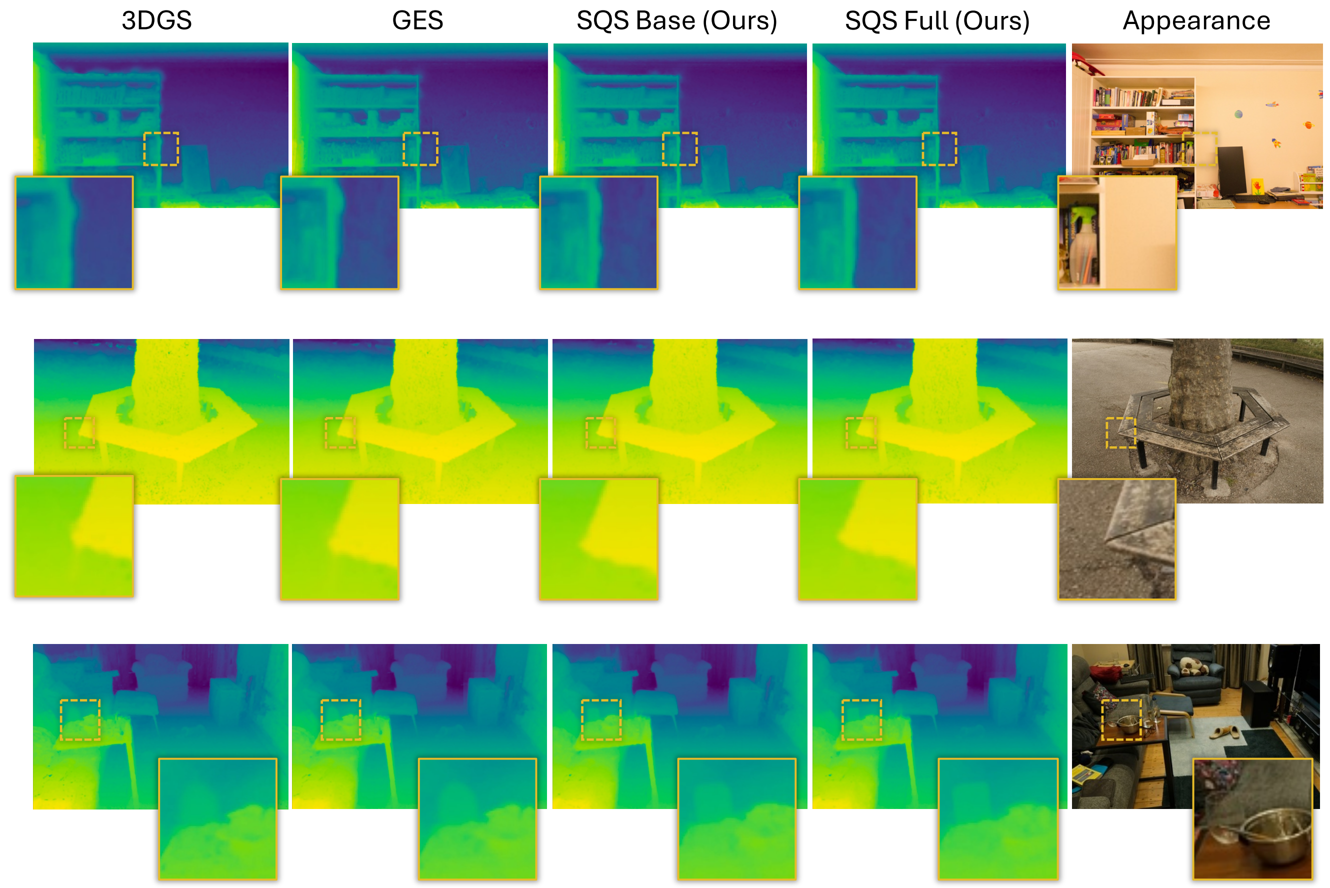

Figure 7. **Qualitative performance of depth reconstruction** on the proposed method (SQS Base and Full) compared with 3DGS [22], GES [11], together with the Ground Truth, across various real-world scenes. The geometry (*e.g.* shelf edges, bench corners, glass cup and bowl, *etc.*) is better reconstructed by the proposed SQS when compared to other methods.

5.4. Ablation Study and Analysis

We analyse the role of each exponential parameter. Recall that ϵ_1, ϵ_2 define the superquadric shape, enabling sharp corners, while ϵ_3 attenuates the distance function, allowing high-contrast edges. We compare Gaussian Splatting (3DGS) [22] (no ϵ), Generalized Exponential Splatting (GES) [11] (ϵ_3 only), and our Superquadric Splatting (SQS) (ϵ_1, ϵ_2, ϵ_3). We also ablate ϵ_3 (denoted SQS Base) to isolate contributions. All methods are trained under identical settings with fixed splats and no adaptive density control to test representational power only. Details are in Sec. 3 of the supplementary material.

6. Conclusion and Discussion

In this paper, we extended Gaussian Splatting to incorporate superquadrics as an alternate volumetric primitive, by introducing a new representation—*SQS*. We derived a new differentiable rendering pipeline to accommodate this. By adding only 3 extra parameters per primitive, we have greatly increased the class of shapes that can be approximated. We demonstrated that superquadrics achieved a consistent $5-10\%$ improvement in depth reconstruction, especially at edges and sharp corners. We also showed an average improvement in visual reconstruction over classic Gaussian Splatting of $\sim$ 1dB. The idealised scene for this method is primarily comprised of sharp edges and corners, typically man-made structures. If there is limited texture, then a small number of superquadric splats can efficiently represent the geometry.

Limitations. Due to the unsolved explicit relation $\phi = f(\theta)$, we were unable to find an analytic solution for the superquadric rim projection. To address this, we stored a set of θ, ϕ values to represent this relation. Specifically, given a pixel's θ_r value, we find the nearest two θs on the rim using memory-efficient binary search. In future work, a better analytic relation could be derived to reduce rendering cost per splat, and a CUDA implementation could considerably increase the speed as well.

Acknowledgements

This project is partially supported by the Royal Society grants (SIF\R1\231009, IES\R3\223050), and is supported by the Amazon Research Award "PCo3D: Physically Plausible Controllable 3D Generative Models". The computations in this research were supported by the Baskerville Tier 2 HPC service. Baskerville was funded by the EPSRC and UKRI through the World Class Labs scheme (EP\T022221\1) and the Digital Research Infrastructure programme (EP\W032244\1) and is operated by Advanced Research Computing at the University of Birmingham.

References

[1] Stephan Alaniz, Massimiliano Mancini, and Zeynep Akata. Iterative superquadric recomposition of 3D objects from multiple views. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18013–18023, 2023.

[2] Alan H Barr. Superquadrics and angle-preserving transformations. *IEEE Computer graphics and Applications*, 1(01): 11–23, 1981.

[3] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. *CoRR*, abs/2111.12077, 2021.

[4] Haodong Chen, Runnan Chen, Qiang Qu, Zhaoqing Wang, Tongliang Liu, Xiaoming Chen, and Yuk Ying Chung. Beyond gaussians: Fast and high-fidelity 3D splatting with linear kernels. *arXiv preprint arXiv:2411.12440*, 2024.

[5] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, pages 2650–2658, 2015.

[6] Zhiwen Fan, Wenyan Cong, Kairun Wen, Kevin Wang, Jian Zhang, Xinghao Ding, Danfei Xu, Boris Ivanovic, Marco Pavone, Georgios Pavlakos, Zhangyang Wang, and Yue Wang. Instantsplat: Sparse-view SfM-free gaussian splatting in seconds, 2024.

[7] Yasutaka Furukawa, Carlos Hernández, et al. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148, 2015.

[8] Zhirui Gao, Renjiao Yi, Yuhang Huang, Wei Chen, Chenyang Zhu, and Kai Xu. Self-supervised learning of hybrid part-aware 3D representation of 2D gaussians and superquadrics. *arXiv preprint arXiv:2408.10789*, 2025.

[9] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 270–279, 2017.

[10] M. Guo, B. Wang, K. He, and W. Matusik. Tetsphere splitting: Representing high-quality geometry with lagrangian volumetric meshes. *arXiv preprint arXiv:2405.20283*, 2024.

[11] Abdullah Hamdi, Luke Melas-Kyriazi, Guocheng Qian, Jinjie Mai, Ruoshi Liu, Carl Vondrick, Bernard Ghanem, and Andrea Vedaldi. Ges: Generalized exponential splatting for efficient radiance field rendering. *arXiv preprint arXiv:2402.10128*, 2024.

[12] hbb1. torch-splatting. https://github.com/hbb1/torch-splatting/blob/main/B075X65R3X.zip, 2023.

[13] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. 37(6):257:1–257:15, 2018.

[14] J. Held, R. Vandeghen, A. Hamdi, A. Deliege, A. Cioppa, S. Giancola, and M. Van Droogenbroeck. 3D convex splatting: Radiance field rendering with 3D smooth convexes. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21360–21369, 2025.

[15] Heiko Hirschmuller and Daniel Scharstein. Evaluation of cost functions for stereo matching. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007.

[16] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2D gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, 2024.

[17] Muhammad Zubair Irshad, Sergey Zakharov, Katherine Liu, Vitor Guizilini, Thomas Kollar, Adrien Gaidon, Zsolt Kira, and Rares Ambrus. Neo 360: Neural fields for sparse view synthesis of outdoor scenes. 2023.

[18] Aleš Jaklič, Aleš Leonardis, and Franc Solina. *Superquadrics and Their Geometric Properties*, pages 13–39. Springer Netherlands, Dordrecht, 2000.

[19] Jianbo Jiao, Ronggang Wang, Wenmin Wang, Shengfu Dong, Zhenyu Wang, and Wen Gao. Local stereo matching with improved matching cost and disparity refinement. *IEEE MultiMedia*, 21(4):16–27, 2014.

[20] Jianbo Jiao, Ying Cao, Yibing Song, and Rynson Lau. Look deeper into depth: Monocular depth estimation with semantic booster and attention-driven loss. In *Proceedings of the European conference on computer vision (ECCV)*, pages 53–69, 2018.

[21] Takeo Kanade and Masatoshi Okutomi. A stereo matching algorithm with an adaptive window: Theory and experiment. *IEEE transactions on pattern analysis and machine intelligence*, 16(9):920–932, 1994.

[22] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023.

[23] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 36(4), 2017.

[24] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, pages 239–248. IEEE, 2016.

[25] Kunyi Li, Michael Niemeyer, Zeyu Chen, Nassir Navab, and Federico Tombari. G2SDF: Surface reconstruction from explicit gaussians with implicit SDFs. 2024.

[26] Rong Liu, Dylan Sun, Meida Chen, Yue Wang, and Andrew Feng. Deformable beta splatting. *arXiv preprint arXiv:2501.18630*, 2025.

[27] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: representing scenes as neural radiance fields for view synthesis. *Commun. ACM*, 65(1):99–106, 2021.

[28] Tom Monnier, Jake Austin, Angjoo Kanazawa, Alexei Efros, and Mathieu Aubry. Differentiable blocks world: Qualitative 3D decomposition by rendering primitives. *Advances in Neural Information Processing Systems*, 36:5791–5807, 2023.

[29] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022.

[30] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. Derf: Decomposed radiance fields, 2020.

[31] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny MLPs. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14335–14345, 2021.

[32] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[33] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard 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)*, pages 519–528. IEEE, 2006.

[34] Shihe Shen, Huachen Gao, Wangze Xu, Rui Peng, Luyang Tang, Kaiqiang Xiong, Jianbo Jiao, and Ronggang Wang. Disentangled generation and aggregation for robust radiance fields. In *European Conference on Computer Vision*, pages 218–236. Springer, 2024.

[35] William Shen, Ge Yang, Alan Yu, Jansen Wong, Leslie Pack Kaelbling, and Phillip Isola. Distilled feature fields enable few-shot language-guided manipulation. *arXiv preprint arXiv:2308.07931*, 2023.

[36] Shimon Ullman. The interpretation of structure from motion. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 203(1153):405–426, 1979.

[37] Nicolas von Lützow and Matthias Nießner. Linprim: Linear primitives for differentiable volumetric rendering. *arXiv preprint arXiv:2501.16312*, 2025.

[38] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy, 2024.

[39] Zhen Wang, Shijie Zhou, Jeong Joon Park, Despoina Paschalidou, Suya You, Gordon Wetzstein, Leonidas Guibas, and Achuta Kadambi. Alto: Alternating latent topologies for implicit 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 259–270, 2023.

[40] Chao-Yuan Wu, Justin Johnson, Jitendra Malik, Christoph Feichtenhofer, and Georgia Gkioxari. Multiview compressive coding for 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9065–9075, 2023.

[41] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[42] Ziyu Zhang, Binbin Huang, Hanqing Jiang, Liyang Zhou, Xiaojun Xiang, and Shunhan Shen. Quadratic gaussian splatting for efficient and detailed surface reconstruction. *arXiv preprint arXiv:2411.16392*, 2024.

[43] M. Zwicker, H. Pfister, J. van Baar, and M. Gross. Ewa splatting. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):223–238, 2002.