6DGS: 6D Pose Estimation from a Single Image and a 3D Gaussian Splatting Model

Bortolon Matteo[1,2,3] , Theodore Tsesmelis[1] , Stuart James[1,4] ,
Fabio Poiesi[2] , and Alessio Del Bue[1]

[1] PAVIS, Fondazione Istituto Italiano di Tecnologia (IIT), Genoa, IT
[2] TeV, Fondazione Bruno Kessler (FBK), Trento, IT
[3] Università di Trento, Trento, IT
[4] Durham University, Durham, UK

Abstract. We propose 6DGS to estimate the camera pose of a target RGB image given a 3D Gaussian Splatting (3DGS) model representing the scene. 6DGS avoids the iterative process typical of analysis-by-synthesis methods (*e.g.* iNeRF) that also require an initialization of the camera pose in order to converge. Instead, our method estimates a 6DoF pose by inverting the 3DGS rendering process. Starting from the object surface, we define a radiant Ellicell that uniformly generates rays departing from each ellipsoid that parameterize the 3DGS model. Each Ellicell ray is associated with the rendering parameters of each ellipsoid, which in turn is used to obtain the best bindings between the target image pixels and the cast rays. These pixel-ray bindings are then ranked to select the best scoring bundle of rays, which their intersection provides the camera center and, in turn, the camera rotation. The proposed solution obviates the necessity of an *"a priori"* pose for initialization, and it solves 6DoF pose estimation in closed form, without the need for iterations. Moreover, compared to the existing Novel View Synthesis (NVS) baselines for pose estimation, 6DGS can improve the overall average rotational accuracy by **12**% and translation accuracy by **22**% on real scenes, despite not requiring any initialization pose. At the same time, our method operates near real-time, reaching **15** fps on consumer hardware.

1 Introduction

Neural and geometrical 3D representations for Novel View Synthesis (NVS) have recently surged in popularity [18,33], and they have been quickly integrated into daily applications, *e.g.* mapping services [1]. The change in 3D representation creates new challenges on how to solve classical problems, such as 6D pose estimation, and on how to leverage NVS implicit advantages [25, 29, 34, 44, 46].

The method of iNeRF [46] pioneered 6D pose estimation using an NVS model by proposing an iterative analysis-by-synthesis, as illustrated in the left panel of Fig. 1. Given a nearby pose initialization (iteration *#1*), the NVS model
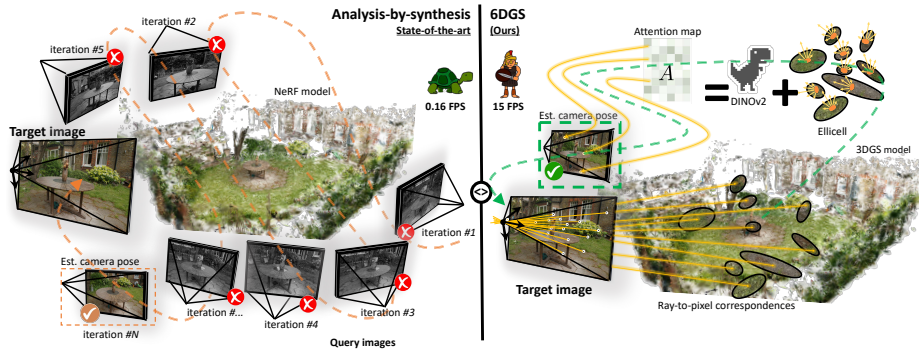

Fig. 1: Our 6DGS method introduces a novel approach to 6DoF pose estimation, departing from conventional analysis-by-synthesis methodologies. Standard NeRF-based methods **(left)** employ an iterative process, rendering candidate poses and comparing them with the target image before updating the pose, which often results in slow performance and limited precision. In contrast, 6DGS **(right)** estimates the camera pose by selecting a bundle of rays projected from the ellipsoid surface (a *radiant Ellicell*) and learning an attention map to output ray/image pixel correspondences (based on DINOv2). The optimal bundle of rays should intersect the optical center of the camera and then are used to estimate the camera rotation in closed-form. Our 6GDS method offers significantly improved accuracy and speed, enabling the recovery of the pose within a one-shot estimate.

is used to render the image related to the initial pose. Then iteratively, the rendered image is compared with the target image using a photometric loss, and the initial pose guess is updated so that the two views achieve the best image overlap at the final step (iteration $\#N$). The authors in iNeRF [46] use the popular NeRF [33] NVS model where backpropagation updates every new pose guess. This procedure leverages the remarkable NeRF capabilities in synthesizing realistic novel views, however, at the computational expense of synthesizing a newly rendered image at each iteration. This limitation restricts iNeRF to offline use while requiring a close initial pose estimate for a successful convergence.

Recent works in 3D Gaussian Splatting (3DGS) [18, 28, 45] are an alternative to Neural NVS models, providing fast rendering speed through the use of explicit geometric primitives that do not require the optimization of a neural network. 3DGS represents a 3D scene as a set of ellipsoids paired with photometric information, such as color and opacity. The ellipsoids are first initialized using Structure from Motion (SfM), and then they are optimized to reduce the photometric error between the rasterized ellipsoids and a set of known images. During the rasterization stage, the 3DGS model is projected onto the image plane as ellipses and for each pixel the algorithm computes its photometric contribution.

By leveraging the 3DGS model properties, we design a novel 6DoF pose estimation method (6DGS) that surpasses the limitations of NeRF-based iterative approaches. 6DGS does not require any pose initialization, and it estimates the camera translation and rotation without an iterating analysis-by-synthesis walk-

through. This is a key factor for achieving near real-time performance ($15fps$), also due to the quick rendering capabilities of 3DGS. The right panel of Fig. 1 presents the gist of our approach for 6DoF pose estimation. If we knew the camera pose, the first NVS step of 3DGS would be to project the ellipsoid centers onto the image plane. Practically, this is a ray casting through the camera's optical center. Our 6DGS attempts to invert this process and, by doing so, to estimate the camera pose. If the target image camera pose is unknown, and thus neither where the optical center is, we are unable to cast the single ray from each ellipsoid that passes through the correct target image pixels. For this reason, instead, we radiate uniformly distributed rays from each ellipsoid through the introduction of a novel casting procedure named Ellicell. Only one radiated ray per ellipsoid would be accurate, *i.e.*, the one that renders the pixel photometrically by projecting the correct ellipse onto the target image plane.

Now, the 6DGS problem is to select, given all the casted rays from the Ellicells, the correct bundle of rays that can generate most of the target image pixels with high confidence. This selection stage is addressed by binding pixels and rays through the learning of an attention map. Notice that this step is also unsupervised, as it leverages the known camera poses and images used to compute the 3DGS model to obtain the pixel and ray pairs used for training. After the bundle of rays is selected, the intersection of these rays identifies the camera center, which is solved using weighted Least Squares (wLS), with the weights being the scores from the previous selection stage. After the optical center is estimated, the optical axis can be used to obtain the camera rotation degrees of freedom from the rays bundle, thus solving the 6DoF pose. By design, 6DGS eliminates the need for an initial camera pose, which is one of the limitations of analysis-by-synthesis pose estimation methods [34, 44, 46], as well as the tendency to converge to local minima during the iteration procedure, especially if the initial pose is initialized far from the optimal position.

We evaluate 6DGS on datasets featuring real-world objects and scenes, comparing against the current NVS state-of-the-art approaches such as iNeRF [46], Parallel iNeRF [25] and NeMO + VoGE [44]. Our experimental results show that 6DGS is competitive, especially if the initial pose is not provided *"a priori"*. Finally, we achieve near real-time 6DoF pose estimation on consumer hardware, which is one rather challenging limitation in the practical application of NVS-based approaches for camera pose estimation. To summarize, 6DGS contributions are threefold:

- Our approach for 6DoF camera pose estimation eliminates the need for an initial camera pose and iterations to converge, which is typically required in analysis-by-synthesis approaches;
- 6DGS employs a novel ray casting pipeline, *i.e.* Ellicell, and an attention-based mechanism that efficiently matches pixel-level image information with 3DGS ellipsoids;
- The proposed method is state-of-the-art in the NVS benchmarks for camera pose estimation both for accuracy and real-time performance.

2   Related works

We review relevant works on 6DoF camera pose estimation based on Neural Radiance Fields (NeRF) models, ellipsoid-based approaches, and correspondence matching methods that are related to key components of 6DGS.

Pose estimation from neural radiance fields. iNeRF [46] pioneered NeRF-based 6D camera pose estimation, using iterative alignment of target and rendered images based on photometric error. However, iNeRF is prone to local minima in the optimization function, leading to recent developments like Parallel iNeRF [25], which employs parallel optimization of multiple candidate poses. While these approaches rely on NeRF-based models, NeMo+VoGe [43,44] have explored 6D camera pose estimation using object models based on volumetric Gaussian reconstruction kernels as geometric primitives. The rendering strategy (VoGE) differs from 3DGS as it is based on ray marching. Therefore, NeMo+VoGe iteratively aligns learned features from target and rendered images. Notably, NeMo+VoGe's training requires multiple objects, in contrast to our method, which leverages a single object 3DGS model. Alternatively, CROSS-FIRE [34] addresses the local minima issue by integrating learned local features, which describes not only the visual content but also the 3D location of the scene in the NeRF model. Despite these advancements, analysis-by-synthesis approaches often struggle with inefficient pose updates due to the nature of the optimization refinement and the dependence on accurate initial pose priors. These factors can limit their real-world applicability. Recently, IFFNeRF [6] utilized a method that inverts the NeRF model to re-render an image to match a target one. However, unlike our approach, it does not consider the specificities of 3DGS, which include ellipsoid elongation and rotation, and their non-uniform distribution across the scene surface.

Pose estimation from ellipsoids. Recovery of the camera pose from ellipsoids has been explored for both SfM [8, 9, 12–14, 37] and SLAM [11, 16, 21, 24, 32, 47] scenarios, where methods frequently recover the object's ellipsoid representation as well as the camera 6DoF. Such approaches typically solve linear systems to recover the pose, most commonly minimizing a loss of the projection to and from an object detection. However, this methodological framework often presents limitations when confronted with large numbers of ellipsoids, as they are more indicated for handling few large ellipsoids that model a single object occupancy, 3D position and orientation.

Correspondences Matching. In traditional 6DoF image matching, feature-based approaches are used, which often rely on hand-crafted features, *e.g.*, SIFT [27] or more recent deep approaches such as SuperGlue [36] and Transfor-Matcher [19]. SuperGlue utilizes a Graph Neural Network (GNN) for feature attention and Sinkhorn [39] for matching, while LightGlue replaces the GNN with a lightweight transformer. Unlike these, TransforMatcher [19] performs global match-to-match attention, allowing for accurate match localization. In addition, there is a body of work around feature equivariance [22,23] for improving the robustness of matching. However, these methods rely on the hypothesis that both feature sets exist in a homogeneous feature space, *i.e.* extracted from the image,

while in 6DGS we have the specific problem to match pixel to rays emitted from the Ellicells. Therefore, we rely on the proposed attention model to handle these ray-to-pixel bindings. OnePose++ [15] instead adopts a multi-modal approach matching a point cloud with an image. Another proposed alternative is to regress directly the pose parameters, as in CamNet [10]. Nevertheless, these approaches require a large amount of training data (≈ 500 or more images), sometimes across multiple scenes and, like with CamNet, these need to be available also at inference time. 6DGS however, requires only ≈ 100 or less images, which are utilized only once during training.

3 Preliminaries

We first review 3D Gaussian Splatting (3DGS) [18] to understand the underlying principles and provide the mathematical formalization of the model. 3DGS objective is to synthesize novel views of a scene by optimizing the position, the orientation and the color of a set of 3D Gaussians approximated as ellipsoids $\mathcal{Q} = \{\mathbf{Q}\}_{i=1}^{K}$ from a given set of input images $\mathcal{I} = \{\mathbf{I}\}_{i=1}^{J}$ and their corresponding camera projection matrices $\mathcal{P} = \{\mathbf{P}\}_{i=1}^{J} \in \mathbb{R}^{3\times4}$. A point $\mathbf{d}$ for being on the surface of an ellipsoid must satisfy the equation $(\mathbf{d} - \mathbf{x})\mathbf{\Sigma}(\mathbf{d} - \mathbf{x})^{T} = 1$, where $\mathbf{x} \in \mathbb{R}^{3}$ is the ellipsoid center and $\mathbf{\Sigma} \in \mathbb{R}^{3\times3}$ its covariance matrix. We can further decompose the covariance of the ellipsoid $\mathbf{\Sigma}$ as:

$$\mathbf{\Sigma} = \mathbf{R}\mathbf{U}\mathbf{U}^{T}\mathbf{R}^{T}, \tag{1}$$

where $\mathbf{R} \in \mathbb{R}^{3\times3}$ is the ellipsoid rotation matrix and $\mathbf{U}^{3\times3}$ denotes the scaling matrix. The projection matrix $\mathbf{P} \in \mathbb{R}^{3\times4}$ allows the projection of the ellipsoid $\mathbf{Q}$ onto the image plane generating the corresponding ellipse representation:

$$\check{\mathbf{y}} = \mathbf{P}\check{\mathbf{x}}^{T}, \;\; \check{\mathbf{E}} = \mathbf{P}\mathbf{\Sigma}\mathbf{P}^{T}, \tag{2}$$

where $\mathbf{y} \in \mathbb{R}^{2}$ and $\check{\mathbf{y}} \in \mathbb{R}^{3}$ correspond to the Euclidean and homogeneous coordinates of the ellipse center point. The homogeneous coordinates $\check{\mathbf{y}}$ originate from the projection of the corresponding ellipsoid center in the homogeneous coordinates $\check{\mathbf{x}} \in \mathbb{R}^{4}$. The matrix $\check{\mathbf{E}} \in \mathbb{R}^{3\times3}$ is the ellipse covariance in homogeneous space. The covariance of the ellipse $\mathbf{E} \in \mathbb{R}^{2\times2}$, is derived by selecting only the first two rows and columns of $\check{\mathbf{E}}$ and dividing by the last element on $\check{\mathbf{E}}$ diagonal.

The splatted ellipses, denoted as $\mathcal{B} = \{\langle\mathbf{y}, \mathbf{E}\rangle\}_{i=1}^{K}$, generate a pixel color with the rendering function ϕ using rasterization techniques [2, 18]. The function ϕ acts independently on every single pixel of the image $\mathbf{p}$. The pixel value depends on the neighboring projected ellipses, taking into account their center points' distances to the pixel coordinates, as well as their orientations and scales. ϕ assumes that the ellipses are ordered based on the depth, so they should be sorted. Formally, ϕ can be expressed as:

$$\phi(\mathcal{B}, \mathbf{p}) = \sum_{i=1}^{K} \rho_i \alpha_i e^{-\tau(\mathcal{B}_i, \mathbf{p})} \gamma(i, \mathbf{p}), \tag{3}$$
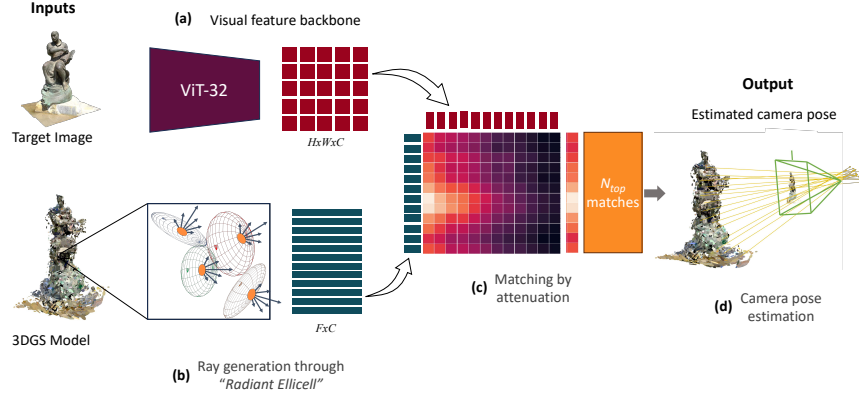
Fig. 2: The figure illustrates the pipeline of our 6DGS methodology. The image is encoded using a visual backbone (**a**). Concurrently, rays are uniformly projected from the center of the 3DGS ellipsoids (**b**), and their corresponding color is estimated. Subsequently, an attention map mechanism is employed to compare the encoded ray and image features (**c**). Following this comparison, the N_{top} matches are selected via attenuation, and the camera location is estimated (**d**) as the solution of a weighted Least Squares problem, resulting in a distinct 6DoF pose for the image.

where ρ and α represent the color and opacity attributes associated with the ellipsoid, which are inherited by the splatted ellipse. Similar to the volumetric rendering equation in NeRF, γ denotes the inverse of the volume density accumulated up to the ith ellipse on pixel $\mathbf{p}$ and is defined as:

$$\gamma(i, \mathbf{p}) = \prod_{j=1}^{i-1}(1 - \alpha_j e^{-\tau(\mathcal{B}_i, \mathbf{p})}).$$

(4)

The purpose of τ is to determine the light absorption by the ellipse when represented as a 2D Gaussian. Light absorption depends on the orientation and distance between the ellipse center, denoted as $\mathbf{y}$, and the pixel location, expressed as $\mathbf{d} = \mathbf{p} - \mathbf{y}$. Consequently, we can formally define τ as:

$$\tau(\mathbf{B}, \mathbf{p}) = \frac{1}{2}\left(\mathbf{1}_2 \mathbf{d}^T \mathbf{E} \mathbf{d} \mathbf{1}_2^T\right),$$

(5)

where $\mathbf{1}_2 \in \mathbb{R}^2$ denotes a vector filled with ones. Following the processing of all pixels onto the image plane, the rendering function ϕ generates an image $\hat{\mathbf{I}} \in \mathbb{R}_+^{H \times W}$, where W and H represent the width and height of the image.

4   Our approach

4.1   Overview

6DGS estimates the camera pose $\hat{\mathbf{P}} \in \mathbb{R}^{3 \times 4}$, given a target image $\mathbf{I}_t$ and a set of ellipsoids $\mathcal{Q}$ from a pre-computed 3DGS model (Fig. 2). To solve for the

(a) Ellicell components   **(b)** 3D Ellicell grid   **(c)** 3D radiant Ellicell
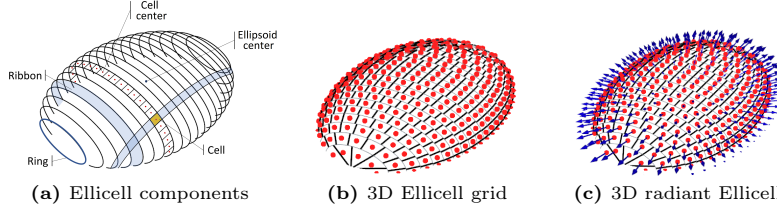
Fig. 3: The illustration depicts the three primary stages involved in the radiant Ellicell generation. Firstly, **(a)** delineates the formulation of components required to compute the geometric information for each cell. Secondly, **(b)** shows the resulting Ellicell grid positioned on the surface of the ellipsoid along with their respective center points. Finally, **(c)** demonstrates the generation of rays originating from the center point of the ellipsoid going through the Ellicell center.

camera pose, we propose a casting method from the ellipsoid's surface, called Ellicell, that divides it in equal area cells (Sec. 4.2). The ellipsoids cast a set of N rays, denoted as $\mathcal{V} = \{\langle \mathbf{v}_o, \mathbf{v}_d, \mathbf{v}_c \rangle\}_{i=1}^N$, one for each of the generated cell (Fig. 3c). Each ray is identified by *i)* the origin $\mathbf{v}_o \in \mathbb{R}^3$, *ii)* the center point of each ellipsoid, *iii)* the direction $\mathbf{v}_d \in \mathbb{R}^3$ originating from the ellipsoid center to the cell center and through the space, and *iv)* the color information $\mathbf{v}_c \in \mathbb{R}^3$ as RGB values. We synthesize the rays' color using the 3DGS rendering function ϕ (Eq. 3). A subset of these rays, depending on the view perspective, may intersect the camera's optical center. For binding the rays to the image pixels we compute the target image pixels features $\psi(\mathbf{I}_t)$ (Fig. 2a) and the rays features $\psi(\mathcal{V})$ (Fig. 2b). These features are used to identify the intersecting rays by using an attention map $\mathcal{A}$ (Fig. 2c), see Sec. 4.4. The higher the attention value for a ray-pixel pair is, the more likely the intersection on the image plane is a valid one. Lastly, we determine $\hat{\mathbf{P}}_t$ (Fig. 2d) by computing the intersection point of rays using the weighted Least Squares algorithm (Sec. 4.5).

4.2 Radiant Ellicell

We create rays spanning in every direction allowing 6DGS to recover the camera pose. We introduce the concept of radiant Ellicell for generating rays that uniformly emanate from the ellipsoid surface, as illustrated in Fig. 3. Ellicell generation is deterministic [5,31] and achieves higher precision with fewer rays [17,42] compared to other sampling methods like Monte-Carlo [30].

First, we compute the area of each Ellicell. This is achieved by calculating the ellipsoid surface area, using a computationally efficient approach, namely Ramanujan approximation [3]:

$$h = 4\pi \left(\frac{(ab)^{1.6} + (ac)^{1.6} + (bc)^{1.6}}{3} \right)^{\frac{1}{1.6}}, \tag{6}$$

where $a, b, c = diag(\mathcal{S})$ are the ellipsoid axis scales. Each Ellicell cell's target area equals $\mu = h/G$, with G being the number of cells dividing each ellipsoid.

Approximating each cell as a square with side $z = \sqrt{\mu}$ we slice the ellipsoids along the major axis into ribbons, each as wide as z (Fig. 3a). The extremity of each ribbon is called a ring. The total number of rings is $e = \lfloor \kappa(a,b)/(2z) \rfloor \in \mathbb{N}$, where $\kappa(a,b)$ computes the ring perimeter. Ignoring ellipsoid's rotation, we compute the ring perimeter by treating them as 2D ellipses, thus defining $\kappa(a,b)$ as:

$$\kappa(a,b) = \pi \left((a+b) + \frac{3(a-b)^2}{10(a+b) + \sqrt{a^2 + 14ab + b^2}} \right). \tag{7}$$

Given the total number of rings e it is possible to compute the ribbon's centerline geometric parameters. In particular, we compute the scale parameter of each ribbon as:

$$\varrho(n, \Delta r, a, b) = \sqrt{1 - \frac{(0.5\Delta r + n\Delta r - a)^2}{b^2}}, \tag{8}$$

where $\Delta r = a/e$ is the distance between two consecutive rings. This equation derives from the manipulation of the standard ellipse equation. While ribbon size z should be equal to Δr, these two values will likely differ due to the need for the number of rings being a natural number. Eq. 8 is also used to compute the other ribbon scaling parameter by replacing b with c. ϱ is then used to compute the number of cells inside each ribbon as:

$$\xi(n, e, a, b, c) = \left\lfloor \frac{\kappa\left(\varrho(n, e, a, b), \varrho(n, e, a, c)\right)}{z} \right\rfloor, \tag{9}$$

where ξ is the number of cells inside the ring. We compute the center of each cell, equally spaced along the ribbon's centerline, by sampling ξ points along it. This is challenging as the perimeter distance does not linearly correlate with the x and y variations. However, we can solve this by using a statistical method. Knowing a distribution's Cumulative Distribution Function (CDF) allows us to sample uniformly between 0 and 1 and then use the CDF inverse to map the sample to the distribution space. This approach applies to our case, where samples are distributed as follows:

$$ds^2 = dx^2 + dy^2, \tag{10}$$

and, by taking its inverse, we can retrieve the coordinates of each cell center. To simplify the equations, we define $r = \varrho(n, e, a, b)$ and $w = \varrho(n, e, a, c)$ to indicate the scale of the ellipse under consideration. Then we express Eq. 10 in polar coordinates to simplify the differentiation:

$$\frac{ds}{d\theta} = \sqrt{r^2 \sin^2 \theta + w^2 \cos^2 \theta}, \tag{11}$$

then, we can express the set of points on the perimeter of the ribbon centerline as an angular position in the polar coordinate system as:

$$\theta' = \left(\frac{ds}{d\theta} \right)^{-1} \left(g \cdot \frac{1}{\xi(n, e, a, b, c)} \right), \tag{12}$$

with g being the cell identifier. Given θ' we can use it inside the ellipse equation in polar coordinates to obtain the 3D position of each cell center:

$$\mathbf{u} = \begin{pmatrix} w\cos(\theta') \\ g\sin(\theta') \\ -a + n\Delta r \end{pmatrix}. \tag{13}$$

4.3   Ray generation

Once we have divided each ellipsoid of the 3DGS model into equidistant cells, we cast the rays originating from the center point of the ellipsoid *i.e.* $\mathbf{v}_o = \mathbf{x}$ and oriented towards the Ellicell center $\mathbf{v}_d = \mathbf{u} - \mathbf{x}$. We reduce the number of potential rays cast from each ellipsoid by considering only the rays oriented in the same hemisphere as the estimated surface normal of the ellipsoid. We obtain the surface normals by treating the ellipsoid centroids as a point cloud, and the surface normal is estimated using the nearby points [41].

Finally, each ray has also been associated with the color information $\mathbf{v}_c$, which we compute through the same pixel-level approach of 3DGS (Eq. 5). We note that the application of the volumetric rendering function of Eq. 5 produces a single pixel for each ray. The generated rays represent a collection of potential hypotheses, meaning that a subset of them will intersect the target image $\mathbf{I}_t$.

4.4   Binding by attenuation of rays to image

Given all the cast rays $\mathbf{v}$, we identify a subset of $\mathbf{v}$ correlating with the target image $\mathbf{I}_t$. A learned attention map $\mathcal{A}$ assigns scores $\hat{\mathbf{s}}$ based on the highest correlation to image pixels; higher similarity results in higher scores. Based on scores $\hat{\mathbf{s}}$, we select the top candidate's rays (N_{top}) that present maximal association and use them to recover the pose $(\hat{\mathbf{P}}_t)$.

To select rays with similar appearance and position, we use a Multi-Layer Perceptron (MLP) defined as $\mathbf{V} = \psi(\mathbf{v})$, where $\mathbf{V} \in \mathbb{R}^{N \times C}$ with C being the feature size and N the overall number of rays. The MLP input is enriched by incorporating Positional Encoding that maps the data in the Fourier domain [40] to better distinguish between similar data.

We generate features from $\mathbf{I}_t$ using DINOv2 [35] as a pre-trained backbone feature extractor. This results in a set of features $\mathbf{F}_t \in \mathbb{R}^{M \times C}$, where $M = W \times H$. Both the image and ray features sets are processed by a single attention module $\mathcal{A}(\mathbf{V}_f, \mathbf{F}_t) \in \mathbb{R}^{M \times N}$ producing a set of scores. Inside the attention module the ray features, $\mathbf{V}$, are used as queries and the image features, $\mathbf{F}_t$, as a key. We optimize the attention map by summing along the rows and converting it into a per-ray correlation score as follows $\hat{\mathbf{s}} = \sum_{i=1}^{M} \mathcal{A}_i$. The higher the score value given by $\hat{\mathbf{s}}$, the better the association between the rays and image pixels. At test-time we select the N_{top} rays with the highest ranking scores.

Because a ray and an image pixel should be associated with each other based on the distance between the camera origin and its projection onto the corresponding ray, we supervise the predicted scores $\hat{\mathbf{s}}$ using the same images used to

estimate the 3DGS model at training time. We compute the projection of the point on the line as $l = max((\mathbf{O} - \mathbf{v}_o)\mathbf{v}_d, 0)$, where $\mathbf{O}$ is the camera position, $\mathbf{v}_o$ the generated ray origin and $\mathbf{v}_d$ the corresponding direction. Rays are infinite only in one direction, so we restrict $l \in \mathbb{R}^+$ using the max operator. Then, we can compute the distance between the camera origin and its projection on the ray as $\mathbf{h} = \|(\mathbf{v}_o + l\mathbf{v}_d) - \mathbf{O}\|_2$. The value $\mathbf{h}$ can span from 0 to $+\infty$, with 0 indicating a ray that passes through the camera's optical center. We map distances to the attention map score using:

$$\delta = 1 - tanh\left(\frac{\mathbf{h}}{\lambda}\right), \quad \mathbf{s} = \delta\frac{M}{\sum \delta}, \tag{14}$$

where λ regulates the number of rays to assign to a specific camera. Lastly, the softmax inside the attention map computation requires we normalize the ground truth scores. We use the $L2$ loss to minimize the difference between the predicted $\hat{\mathbf{s}}$ and the computed ground truth $\mathbf{s}$ scores as:

$$\mathcal{L} = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\|\hat{\mathbf{s}}_{i,j} - \mathbf{s}_{i,j}\|_2, \tag{15}$$

where M, N are the size of the attention map $\mathcal{A}$. During each training iteration, we predict an image and a pose utilized for estimating the 3DGS model.

4.5   Test-time pose estimation

During the test phase, the predicted scores $\hat{\mathbf{s}}$ are used to select the top N_{top} rays, identified as the utmost relevant, and constrained to choose at most one ray per ellipsoid. Note that only a small set of rays is sufficient to estimate the camera pose. However, based on an ablation study we set $N_{top} = 100$, see Tab. 3a.

The camera position is found at the intersection of selected rays, solved as a weighted Least Squares problem. Since 3D lines usually do not intersect at a single point due to discretization noise introduced by the Ellicell, we minimize the sum of squared perpendicular distances instead.

For the selected ray $\mathbf{v}_j$ with $f = 1 \ldots N_{top}$, the error is given by the square of the distance from the camera position to predict $\hat{\mathbf{O}}$ to its projection on $\mathbf{v}_j$:

$$\sum_{f=1}^{N_{top}}\left((\hat{\mathbf{O}} - \mathbf{v}_{o,f})^T(\hat{\mathbf{O}} - \mathbf{v}_{o,f}) - ((\hat{\mathbf{O}} - \mathbf{v}_{o,f})^T\mathbf{v}_{d,f})^2\right), \tag{16}$$

where $\mathbf{v}_{o,f}$ indicating the origin of the f-th ray and $\mathbf{v}_{d,f}$ the respective direction. To minimize Eq. 16, we differentiate it with respect to $\hat{\mathbf{O}}$, resulting in

$$\hat{\mathbf{O}} = \sum_{f=1}^{N_{top}}\hat{\mathbf{s}}_f(\mathbb{I} - \mathbf{v}_{d,f}\mathbf{v}_{d,f}^T)\mathbf{v}_{o,f}, \tag{17}$$

where $\mathbb{I}$ is the identity matrix and $\hat{\mathbf{s}}_f$ are the predicted ray scores. This expression can be solved as a weighted system of linear equations.

Table 1: Evaluation of 6DoF pose estimation on the Mip-NeRF 360° [4] dataset. We report results in terms of Mean Angular Error (MAE) and Mean Translation Error (MTE) in terms of degrees and units, u, respectively. Where $1u$ is equal to the object's largest dimension. For both metrics lower is better. Best-performing results are highlighted in **bold** and green, while second best values are highlighted in orange.

| | Fixed pose prior (eval. protocol by [46]) | | | | | | Random pose prior | | | | | | No pose prior | |
| | iNeRF [46] | | NeMo + VoGE [44] | | Parallel iNeRF [25] | | iNeRF [46] | | NeMo + VoGE [44] | | Parallel iNeRF [25] | | 6DGS (Ours) | |
Scenes	MAE↓	MTE↓	MAE↓	MTE↓	MAE↓	MTE↓	MAE↓	MTE↓	MAE↓	MTE↓	MAE↓	MTE↓	MAE↓	MTE↓
Bicycle	39.5	0.116	43.8	0.015	35.9	0.116	76.6	0.217	111.8	0.038	44.4	0.150	**12.1**	**0.010**
Bonsai	51.3	0.228	52.5	**0.036**	41.1	0.223	96.7	0.385	98.9	0.073	58.2	0.298	**10.5**	0.038
Counter	40.7	0.324	45.6	0.072	24.7	0.212	70.3	0.487	98.1	0.139	42.1	0.435	**19.6**	**0.043**
Garden	31.0	0.121	31.8	0.026	**18.2**	0.090	72.8	0.210	89.2	0.038	60.0	0.144	37.8	**0.015**
Kitchen	38.2	0.113	41.6	0.042	37.3	0.109	100.2	0.266	122.2	0.082	65.0	0.193	**23.2**	**0.018**
Room	38.8	0.274	44.9	0.045	**30.7**	0.257	91.6	0.444	110.0	0.010	63.5	0.271	38.3	**0.019**
Stump	21.4	0.030	26.3	0.016	**14.8**	0.016	86.9	0.035	96.3	0.025	72.6	0.033	28.3	**0.009**
Avg.	37.3	0.172	40.9	0.036	28.9	0.146	85.0	0.292	103.8	0.058	58.0	0.218	**24.3**	**0.022**

5   Results

5.1   Experimental setup

We evaluate 6DGS and compare with other analysis-by-synthesis methods for 6D pose estimation, including iNeRF [46], Parallel iNeRF [25], and NeMo+VoGE [43, 44]. We reproduce the results using their published code. We follow iNeRF's evaluation protocol and test on two real-world datasets: Tanks & Temples [20] and Mip-NeRF 360° [4]. For each dataset, we use the predefined training-test splits and evaluate them with two pose initialization pipelines: *i)* the original iNeRF initialization, where the starting pose is sampled uniformly between $[-40°, +40°]$ degrees of errors and $[-0.1, +0.1]$ units of translation error from the ground-truth target pose; *ii)* by randomly choosing an initialization pose from the ones used to create the 3DGS mode. Although analysis-by-synthesis methods were tested with a prior, in reality it is rarely available, so we present a second scenario to assess them under more realistic conditions. We perform multiple ablation studies to assess the sensitivity of 6DGS to different hyperparameters and settings. We quantify pose estimation results in terms of mean angular (MAE) and translation (MTE) errors (see Tab. 1 and Tab. 2) and measure the inference time.

Implementation Details. 6DGS is implemented in PyTorch and the attention map was trained for 1.5K iterations (∼45mins) with an NVIDIA GeForce RTX 3090. We use the Adafactor optimizer [38] with weight decay of 10^{-3}. For speedup training, we uniformly sample 2000 3DGS ellipsoids at each iteration.

5.2   Datasets

To demonstrate the applicability of 6DGS, we test on two datasets featuring real world challenges. **Tanks&Temples** [20] was created to evaluate 3D reconstruction methods with challenging real-world objects of varying sizes, acquired from human-like viewpoints and with difficult conditions (illumination, shadows, and reflections). We use the five scenes (Barn, Caterpillar, Family, Ignatius, Truck) and the train test splits given in [7, 26]. The splits are object dependent, having

Table 2: Evaluation of 6DoF pose estimation on the Tanks&Temples [20] dataset. We show the same metrics and analysis as in Table 1.

	Fixed pose prior (eval. protocol by [46])						Random pose prior						No pose prior	
	iNeRF [46]		NeMo + VoGE [44]		Parallel iNeRF [25]		iNeRF [46]		NeMo + VoGE [44]		Parallel iNeRF [25]		6DGS (Ours)	
Objects	MAE ↓	MTE ↓	MAE ↓	MTE ↓	MAE ↓	MTE ↓	MAE ↓	MTE↓	MAE ↓	MTE ↓	MAE ↓	MTE ↓	MAE ↓	MTE ↓
Barn	26.5	0.208	51.2	0.752	**22.9**	**0.131**	89.2	0.682	92.5	0.684	85.2	0.572	30.3	0.162
Caterpillar	42.9	0.166	52.6	0.516	25.2	0.138	89.3	2.559	90.5	2.559	86.8	0.843	**14.5**	**0.027**
Family	42.8	0.794	58.4	1.130	22.9	0.507	93.9	1.505	97.0	1.506	99.0	2.028	**20.6**	**0.468**
Ignatius	31.4	0.723	51.2	1.193	23.4	0.604	84.1	1.489	85.4	1.491	86.9	1.326	**15.5**	**0.441**
Truck	31.6	0.370	54.6	1.236	29.4	0.351	94.4	1.042	97.7	1.045	97.6	0.883	**27.5**	**0.242**
Avg.	35.0	0.452	53.6	0.965	24.7	0.346	90.2	1.455	92.6	1.457	91.1	1.130	**21.7**	**0.268**

on average ≈ 247 training images (87%) and ≈ 35 testing images (12%). **Mip-NeRF 360°** [4] consists of seven scenes: two outdoors and four indoors, with a structured scenario and background. We use the original train-test splits [4], at a ratio of 1:8. Following [25], we resize all the objects to fit inside a unit box. The translation error is relative to the object size, defined as a unit.

5.3 Analysis

Quantitative Analysis: Tab. 1 and Tab. 2 present the results obtained across both datasets. 6DGS consistently outperforms baseline methods across all datasets and pose initialization pipelines. Notably, 6DGS achieves lower error rates than the second-best results, especially under identical comparison conditions (*i.e.*, random pose prior). Even when initialized from a fixed pose proximal to the known camera, 6DGS still excels over baselines in most scenes. Parallel iNeRF demonstrates improvement over iNeRF across all tested scenarios, consistent with its reported enhancements, but both methods' performance drops with random initialization. Likewise, NeMo+VoGE performs worst, especially with random pose prior due to the utilization of a smaller number of larger ellipsoids in their approach. In contrast, 6DGS leverages approximately 300,000 ellipsoids of varying sizes obtained via 3DGS, as opposed to their mesh-to-ellipsoid method, which utilizes only about 5,000 larger ellipsoids. This fundamental disparity in ellipsoid size and quantity is a crucial factor contributing to the performance difference. Additionally, 6DGS exhibits faster processing speeds, operating nearly in real-time at 15 *frames per second* (*fps*) compared to the 0.05*fps* of Parallel iNeRF and 0.16*fps* of iNeRF. Please refer to the supplementary material for the complete table on timings.

Qualitative Analysis: Figure 4 illustrates qualitative findings revealing notable observations. Particularly, we notice that the estimated poses exhibit proximity to the object relative to ground truth, attributable to the quantization effect introduced by the Ellicell. The qualitative findings verify the quantitative outcomes, albeit occasional inconsistencies in results, such as in the Counter scene, with the analysis-by-synthesis approaches showcasing a total incoherent output in regards to the overall scene (notice how the estimated poses are completely off the target). Moreover, the performance of 6DGS demonstrates consistency across varied scenarios, encompassing single-object instances and indoor settings, despite substantial variations in the models utilized.
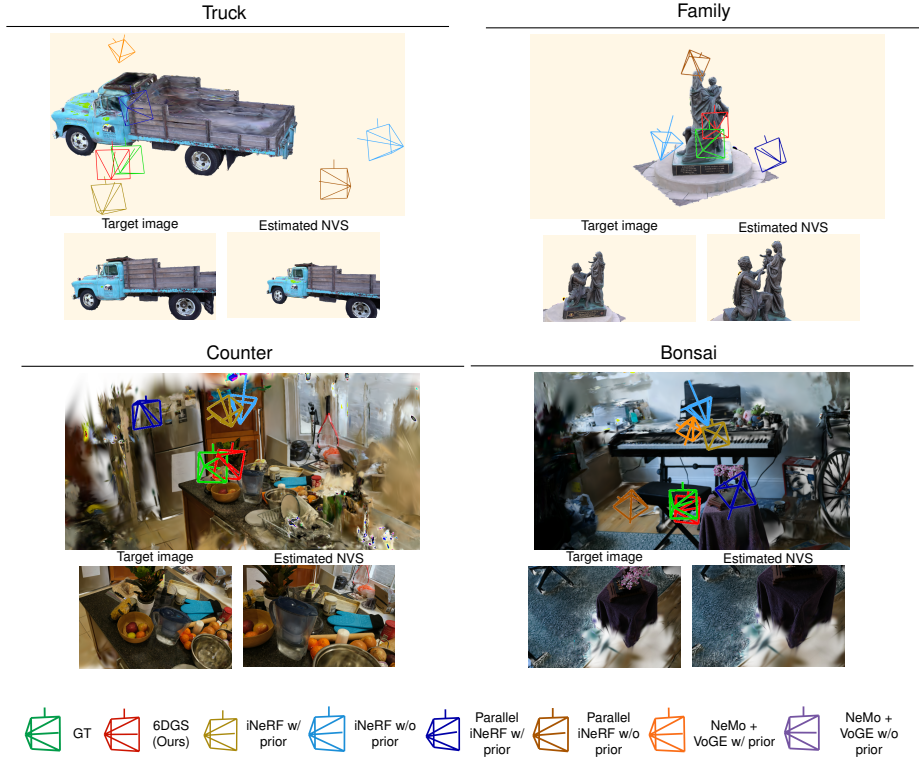
Fig. 4: The illustration presents qualitative results from Tanks & Temple (upper row) and Mip-NeRF 360° (lower row) datasets. Each scene showcases the target images (bottom left) along with their corresponding Novel View Synthesis (NVS) outputs (bottom right), derived from the camera poses estimated by 6DGS (located on the top). Furthermore, the estimated camera poses from the comparative baselines are visualized, with distinct colors as indicated in the image legend. The NVS of each scene is rendered based on the provided 3DGS model. Please check the supplementary material for more qualitative results.

5.4 Ablation studies

Our ablation studies involve the analysis of the number of rays selected for the pose estimation (Tab. 3a), the number of rays that we cast from a Ellicell (Tab. 3b) as well as the different feature size on the MLP channels (Tab. 3c). The supplementary material contains additional ablations that analyze 6DGS performance with low-quality 3DGS models.

We find that the number of selected rays mainly affects the angular error, while the translation error remains relatively stable. Increasing the number of rays decreases the angular error but slightly increases the translation error, likely due to less confident rays contributing to the pose estimation. The optimal balance between translation and angular errors is achieved between 100 to 150 rays, with 100 being the best. The slight increase in error with more N_{top} rays is due to

Table 3: Ablation study on the number of rays selected for pose estimation, on the rays cast from each ellipsoid and on the MLP channels using Mip-NeRF 360 [4]. <u>Underline</u> indicates the default values used.

(a) Number of rays used for pose estimation.

N_{top}	MAE (°)↓	MTE (u)↓	Time (s)
20	29.0	0.0235	0.03
50	26.3	0.0227	0.04
<u>100</u>	<u>24.3</u>	<u>0.0217</u>	<u>0.06</u>
150	24.4	0.0219	0.9
200	24.5	0.0222	0.11

(b) Number of cast rays per ellipsoid.

# of cast rays	MAE (°)↓	MTE (u)↓	Time (s)
20	29.0	0.0235	0.04
35	24.7	0.0220	0.04
<u>50</u>	<u>24.3</u>	<u>0.0217</u>	<u>0.06</u>
65	25.1	0.0218	0.09
80	25.2	0.0221	0.15

(c) MLP channel feature size.

MLP channels	MAE (°)↓	MTE (u)↓	Time (s)
256	29.4	0.0273	0.04
<u>512</u>	<u>24.3</u>	<u>0.0217</u>	<u>0.06</u>
1024	30.1	0.0228	0.27

introducing rays not pointing precisely to the camera's optical center. Similar to what we observed in the qualitative examples, the noisy rays make the weighted Least Squares estimating the camera closer to the object than it actually is.

Regarding the impact of the varying number of rays cast from the Ellicells, the angular error tends to remain relatively constant across different configurations. In contrast, the translation error decreases when 50 cast rays are used, and then increases again. This behavior is connected to network generalization capability. Increasing the number of rays allows the network to fit the training set better, but at test time, it makes the network more prone to noise and consequently selecting the wrong rays, thus increasing the error. We observe this generalization issue when increasing the MLP channels, see Tab. 3c, particularly given the limited and uneven distribution of training images (≈ 150). Moreover, the processing time increases proportionally with the number of rays and the MLP channels; upon exceeding the default values for rays and feature size, a notable surge in processing time is observed, reaching approximately $10 fps$ and $13 fps$, respectively.

6   Conclusions

In this study, we proposed a novel ray sampling by attention method for estimating 6DoF camera poses from a single image and a 3DGS scene model. Our analytical evaluation demonstrates its robustness and efficiency without requiring initialization, up to 22% in accuracy and while being faster by a big margin, approx. 94x faster. Furthermore, the proposed method formulates and utilizes a novel ray generation methodology in order to explore diverse camera pose hypotheses in accordance to an effective attention mechanism. Our method exhibits enhanced robustness across real-world datasets and holds promise for real-time deployment in robotics and other fields. Future research endeavors will focus on improving accuracy and extending applicability to diverse scenes and objects.
Limitations. The main constraint of 6DGS is the need for retraining with each new scene. This could be mitigated with meta-learning, particularly when similar objects or scenes are under consideration.

Acknowledgments

References

1. Google maps nerf integration. `https : / / blog . google / products / maps / sustainable-immersive-maps-announcements/`, accessed: 2024-03-07
2. Akenine-Mo, T., Haines, E., Hoffman, N., et al.: Real-time rendering. AK Peters/CRC Press (2018)
3. Almkvist, G., Berndt, B.: Gauss, landen, ramanujan, the arithmetic-geometric mean, ellipses, π, and the ladies diary. The American Mathematical Monthly **95**(7), 585–608 (1988)
4. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: CVPR (2022)
5. Beckers, B., Beckers, P.: Fast and accurate view factor generation. In: FICUP (2016)
6. Bortolon, M., Tsesmelis, T., James, S., Poiesi, F., Del Bue, A.: Iffnerf: Initialization free and fast 6dof pose estimation from a single image and a nerf model. In: ICRA (2024)
7. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorf: Tensorial radiance fields. In: ECCV (2022)
8. Chen, S., Song, S., Zhao, J., Feng, T., Ye, C., Xiong, L., Li, D.: Robust dual quadric initialization for forward-translating camera movements. RAL **6**(3), 4712–4719 (2021)
9. Crocco, M., Rubino, C., Del Bue, A.: Structure from motion with objects. In: CVPR (2016)
10. Ding, M., Wang, Z., Sun, J., Shi, J., Luo, P.: Camnet: Coarse-to-fine retrieval for camera re-localization. In: ICCV (2019)
11. Gaudillière, V., Simon, G., Berger, M.O.: Camera relocalization with ellipsoidal abstraction of objects. In: ISMAR (2019)
12. Gaudillière, V., Simon, G., Berger, M.O.: Perspective-2-ellipsoid: Bridging the gap between object detections and 6-dof camera pose. RAL **5**(4), 5189–5196 (2020)
13. Gay, P., Rubino, C., Bansal, V., Del Bue, A.: Probabilistic structure from motion with objects (psfmo). In: ICCV
14. Gay, P., Stuart, J., Del Bue, A.: Visual graphs from motion (vgfm): Scene understanding with object geometry reasoning. In: ACCV (2019)
15. He, X., Sun, J., Wang, Y., Huang, D., Bao, H., Zhou, X.: Onepose++: Keypoint-free one-shot object pose estimation without cad models. In: NeurIPS (2022)
16. Hosseinzadeh, M., Latif, Y., Pham, T., Suenderhauf, N., Reid, I.: Structure aware slam using quadrics and planes. In: ACCV (2019)
17. Jacques, L., Masset, L., Kerschen, G.: Direction and surface sampling in ray tracing for spacecraft radiative heat transfer. Aerospace Science and Technology **47** (2015)

18. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. TCG **42**(4) (2023)
19. Kim, S., Min, J., Cho, M.: Transformatcher: Match-to-match attention for semantic correspondence. In: CVPR (2022)
20. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. TCG **36**(4) (2017)
21. Laidlow, T., Davison, A.J.: Simultaneous localisation and mapping with quadric surfaces. In: 3DV (2022)
22. Lee, J., Kim, B., Cho, M.: Self-supervised equivariant learning for oriented keypoint detection. In: CVPR (2022)
23. Lee, J., Kim, B., Kim, S., Cho, M.: Learning rotation-equivariant features for visual correspondence. In: CVPR
24. Liao, Z., Hu, Y., Zhang, J., Qi, X., Zhang, X., Wang, W.: So-slam: Semantic object slam with scale proportional and symmetrical texture constraints. RAL **7**(2), 4008–4015 (2022)
25. Lin, Y., Müller, T., Tremblay, J., Wen, B., Tyree, S., Evans, A., Vela, P.A., Birchfield, S.: Parallel inversion of neural radiance fields for robust pose estimation. In: ICRA (2023)
26. Liu, L., Gu, J., Lin, K.Z., Chua, T.S., Theobalt, C.: Neural sparse voxel fields. In: NeurIPS (2020)
27. Lowe, D.G.: Object recognition from local scale-invariant features. In: ICCV
28. Luiten, J., Kopanas, G., Leibe, B., Ramanan, D.: Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In: 3DV (2024)
29. Maggio, D., Mario, C., Carlone, L.: Verf: Runtime monitoring of pose estimation with neural radiance fields. In: ICCV (2023)
30. Malley, T.: A shading method for computer generated images. Master's thesis, Dept. of Computer Science, University of Utah (1988)
31. Masset, L., Brüls, O., Kerschen, G.: Partition of the circle in cells of equal area and shape. Tech. rep., Structural Dynamics Research Group, Aerospace and Mechanical Engineering Department, University of Liege, 'Institut de Mecanique et G ´enie Civil (B52/3) (2011)
32. Meng, Y., Zhou, B.: Ellipsoid slam with novel object initialization. In: CASE (2022)
33. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020)
34. Moreau, A., Piasco, N., Bennehar, M., Tsishkou, D., Stanciulescu, B., de La Fortelle, A.: Crossfire: Camera relocalization on self-supervised features from an implicit representation. In: ICCV (2023)
35. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.Y., Li, S.W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: Dinov2: Learning robust visual features without supervision (2023)
36. Sarlin, P.E., DeTone, D., Malisiewicz, T., Rabinovich, A.: Superglue: Learning feature matching with graph neural networks. In: CVPR (2020)
37. Shan, M., Feng, Q., Jau, Y.Y., Atanasov, N.: Ellipsdf: joint object pose and shape optimization with a bi-level ellipsoid and signed distance function description. In: ICCV (2021)
38. Shazeer, N., Stern, M.: Adafactor: Adaptive learning rates with sublinear memory cost. In: ICML (2018)

39. Sinkhorn, R.: A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices. The Annals of Mathematical Statistics **35**(2), 876 – 879 (1964)
40. Tancik, M., Srinivasan, P.P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J.T., Ng, R.: Fourier features let networks learn high frequency functions in low dimensional domains. In: NeurIPS (2020)
41. Tombari, F., Salti, S., di Stefano, L.: Unique signatures of histograms for local surface description. In: ECCV (2010)
42. Tsesmelis, T., Hasan, I., Cristani, M., Bue, A.D., Galasso, F.: Rgbd2lux: Dense light intensity estimation with an rgbd sensor. In: WACV (2018)
43. Wang, A., Kortylewski, A., Yuille, A.: Nemo: Neural mesh models of contrastive features for robust 3d pose estimation. In: ICLR (2020)
44. Wang, A., Wang, P., Sun, J., Kortylewski, A., Yuille, A.: Voge: a differentiable volume renderer using gaussian ellipsoids for analysis-by-synthesis. In: ICLR (2022)
45. Xie, T., Zong, Z., Qiu, Y., Li, X., Feng, Y., Yang, Y., Jiang, C.: Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In: CVPR (2024)
46. Yen-Chen, L., Florence, P., Barron, J.T., Rodriguez, A., Isola, P., Lin, T.Y.: iNeRF: Inverting neural radiance fields for pose estimation. In: IROS (2021)
47. Zins, M., Simon, G., Berger, M.O.: Oa-slam: Leveraging objects for camera relocalization in visual slam. In: ISMAR (2022)

6DGS: 6D Pose Estimation from a Single Image and a 3D Gaussian Splatting Model - Supplementary material

Bortolon Matteo[1,2,3] , Theodore Tsesmelis[1] , Stuart James[1,4] ,
Fabio Poiesi[2] , and Alessio Del Bue[1]

[1] PAVIS, Fondazione Istituto Italiano di Tecnologia (IIT), Genoa, IT
[2] TeV, Fondazione Bruno Kessler (FBK), Trento, IT
[3] Università di Trento, Trento, IT
[4] Durham University, Durham, UK

1 Introduction

The provided supplementary material shows extra details, results, and figures to support the main findings presented in the main manuscript. In Section 2 we present the inference time performance of each method for the task of 6D pose estimation. Section 4 offers additional figures showing qualitative results and discussions as mentioned in the main paper. These extra results aim to make the study clearer and more complete, providing further useful insights into our proposed 6DGS pipeline.

2 Inference time

In Tab. 1 we detail the timings of each method. It can be noted from the table that our approach is the only one that is able to run under $1s$ and near to real-time performance at 16FPS. In general, it is possible to notice how time slightly increases in the Mip-NeRF 360°dataset due to the higher complexity of the scenes. The increase in computational complexity can be attributed to several factors, including the size of the neural models required for iNeRF and Parallel iNeRF, as well as the quantity of ellipsoids used in NeMo+VoGE for each of the two datasets (*i.e.* ≈ 150000 ellipsoids for Tanks&Temples and ≈ 300000 for Mip-NeRF 360°, while in NeRF models Tanks&Temples use $\approx 50\%$ less parameters than Mip-NeRF 360°). As it can be noticed 6DGS is able to handle both datasets and especially the the Mip-NeRF 360°scenes without issues.

Considering the different methods, we can notice that NeMo+VoGE requires the most time. This derive from the processing algorithm that struggles with large number of ellipsoids. In regards to the analysis-by-synthesis methods, iNeRF represents the fastest method, with Parallel NeRF being the second fastest. However, all methods show a considerable higher running time in contrast to 6DGS.

Project page: https://mbortolon97.github.io/6dgs/
Corresponding author: mbortolon@fbk.eu

3   Additional ablations studies

To assess our method's robustness to 3DGS model quality, we conducted two experiments using the Mip360° dataset's Bonsai and Bicycle scenes. First, we investigated the effect of 3DGS model accuracy on pose estimation by introducing varying noise levels to the Gaussian centers. Second, we analyzed the performance of our method under sparse viewpoint conditions.

The results below show that 6DGS is resilient for translation and minimal increase for rotational error, both with high noise level and even with only six views ($\approx 60°$ between viewpoints).

4   Additional qualitative results

In this section, we report additional qualitative results of other scenes not presented in the paper. Fig. 1 shows the Tanks&Temples dataset [20] scenes *Barn*, *Caterpillar*, *Ignatius*, and *Truck*. On the other hand, Fig. 2 presents views from the *Garden*, *Kitchen*, *Stump*, *Room*, *Bicycle*, and *Counter* scenes of the Mip-NeRF 360°dataset [4]. In each figure, we show the ground truth camera position (green) and our estimated camera pose (red) in addition to the baseline methods (see legend for camera colors and details). In addition to the 3D camera positions estimated by our method and the other baselines, we also show the rendered images using the 3DGS model from ground truth and estimated 6DGS camera pose. From the rendered images, we can assess visually how the synthesized image viewpoint is different from the tested target image.

Looking closer at Fig. 1, our method predicts poses with a more accurate rotation, aligning closer to the ground truth. The translation can be slightly less accurate, resulting in either being slightly closer, *e.g.* for the *Caterpillar*, *Ignatius*, and *Truck* scenes, or further, *e.g.* for the *Barn* scene, from the object. A similar estimation of accuracy in rotation can be seen for the scenes *Kitchen*,

Table 1: Average computation time to estimate the pose of an image. Comparison between our method and state-of-the-art approaches across the two datasets. **Bold** font indicates best performance. Time is reported in seconds.

	Tanks and Temples	Mip-NeRF 360
iNeRF (pose prior by [46])	6.321	8.214
NeMo+VoGE (pose prior by [46])	251.4	290.4
Parallel iNeRF (pose prior by [46])	16.9	16.8
iNeRF (random pose)	6.2	8.5
NeMo+VoGE (random pose)	240.5	284.4
Parallel iNeRF (random pose)	16.5	17.1
6DGS (Ours) (no pose)	**0.05**	**0.06**

Gauss. noise (‰)	MAE (°)	MTE (u)	No. views	MAE (°)	MTE (u)
0.0‰u (None)	11.3	0.024	All	11.3	0.024
0.1‰u	18.3	0.025	12	15.5	0.024
0.5‰u	22.5	0.025	6	18.6	0.026

Stump of the Mip-NeRF 360°dataset, see in Fig. 2 while showing a closer camera position though in a more limited amount. Mip-NeRF 360°also showcases a degradation in performance on camera rotation estimation related to the scenes *Room, Stump, Garden*. As for the baselines, in general, we can see how Parallel iNeRF tends to be more precise compared to the ones from iNeRF, but it can still fail in the complex scenes as shown. Finally, for NeMo + VoGE, the qualitative examples are consistent with the quantitative results, which tend to produce poses with a higher error than the other approaches.
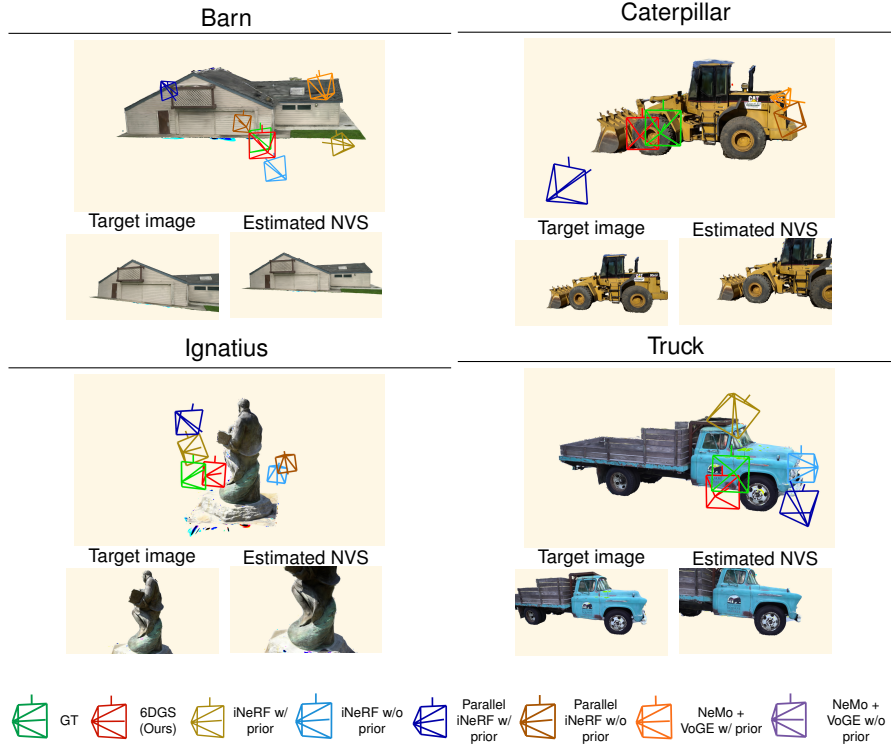
Fig. 1: Additional scenes from the Tanks&Temple dataset. For each scene, we show a visualization of the camera poses in regards to the model (top) for 6DGS as well as the baselines, which are visualized with different colors as indicated in the image legend. In addition, for each scene, we showcase the target image (bottom left) along with their corresponding Novel View Synthesis (NVS) output (bottom right) of the estimated camera pose by 6DGS.
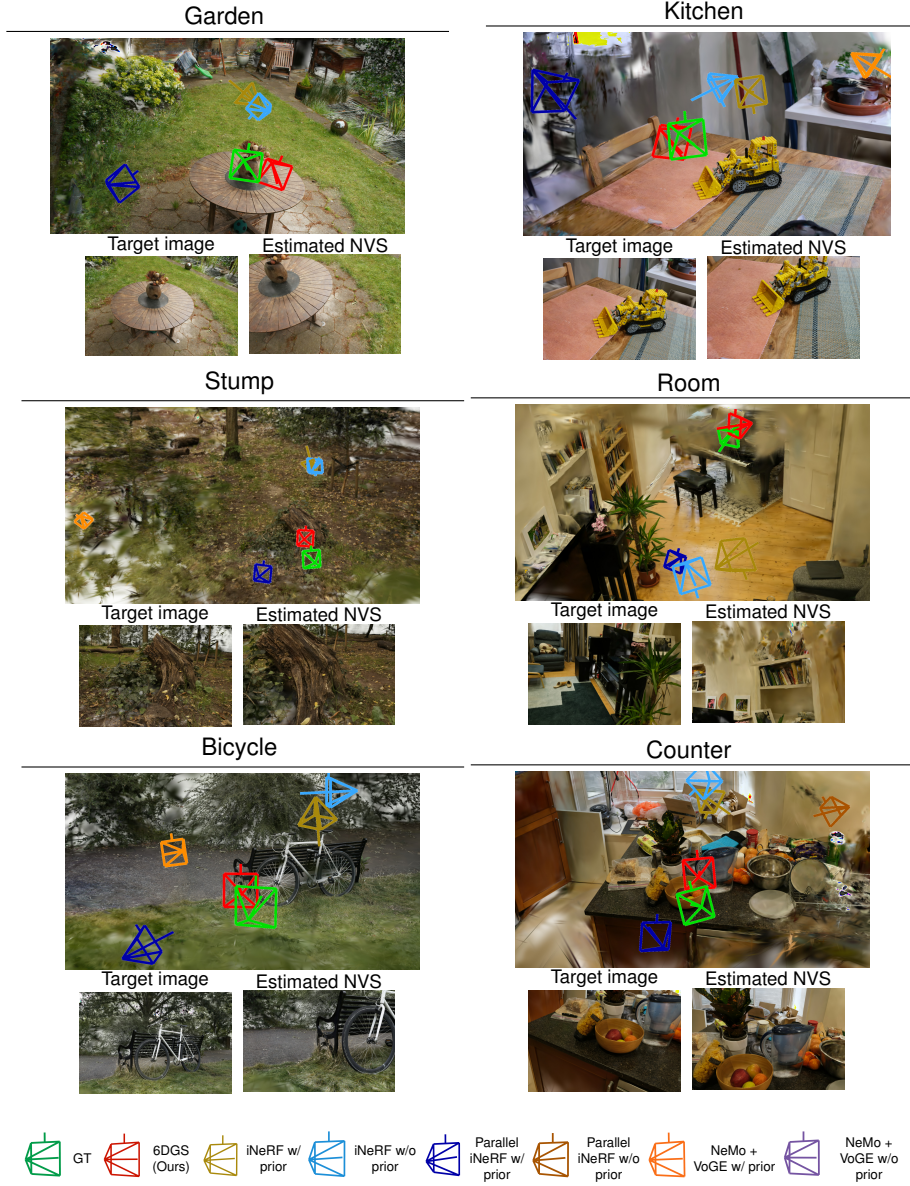
Fig. 2: Additional scenes from the Mip-NeRF 360°dataset. For each scene, we show a visualization of the camera poses in regards to the model (top) for 6DGS as well as the baselines, which are visualized with different colors as indicated in the image legend. In addition, for each scene, we showcase the target image (bottom left) along with their corresponding 3DGS Novel View Synthesis (NVS) output (bottom right) of the estimated camera pose by 6DGS.