Rendering and Optimization of Gaussian Radiance Fields

Valerie Dagley, Cameron Durbin, Andrew Rehmann

October 16, 2023

1 Introduction

The topic of novel view synthesis is a large and ongoing research area. Novel view synthesis (NVS) is the process of taking a set of images of a scene and generating new images from viewpoints not covered by the original data set. We plan to re-implement a recent NVS approach utilizing 3D Gaussians to create a scene representation capable of real time rendering of novel views.

1.1 Description of Area

We chose to focus on the area of NVS relating to creating photorealistic renderings of physical scenes. Techniques exist to create other types of novel views (e.g. people in novel poses, scene relighting), however these are mostly unrelated to our problem area.

Early examples of NVS in the area of photorealistic rendering include point cloud rendering and photogrammetry. The generation of point clouds acts as a basis for many discrete scene representations, and can be generated with a varying number of photos. However, directly rendering scenes from point clouds fails when points become too sparse–requiring larger data sets and higher resolution base photos to create dense point clouds. Photogrammetry is the process of creating mesh and texture data from these point clouds. This works extremely well for existing rendering architectures, which optimize performance for high resolution mesh rendering. However, as with direct point cloud rendering, large data sets and high resolution photos are required for a quality output. Both of these techniques have applications in the real world. Point clouds are used extensively in fields like surveying and engineering due to the simplicity of construction for large areas with drones or laser technology. Photogrammetry is utilized in modern games, TV, and movie development. However, the "cost of entry" for these techniques prevents them from accessible individual use.

Neural Radiance Fields (NeRFs) are a much more modern approach to NVS which require far fewer images to create a quality scene representation. NeRFs use machine learning techniques to generate a volumetric "radiance function" that takes in a position and view direction to output a view-dependent emitted radiance for the specific point. They then use a volumetric ray-tracing approach to render this radiance field. NeRFs allow for high quality view synthesis within a small range of viewpoints, however they have a few key failures. First, the use of volumetric ray tracing is extremely computationally expensive, preventing real time rendering in many situations. Second, radiance fields are inefficient when it comes to representing high frequency details (wires, fences, etc.). Third, radiance fields are difficult to combine with existing rendering methods.

1.2 Problems to be Explored

We plan to re-implement a very recent approach to NVS proposed by Kerbl et. al. [2]. They propose a unique scene representation using 3D Gaussians (points with a sphere of influence, with a transparent "fog" tapering off with distance) to convert a point cloud representation into a differentiable volumetric representation. Their approach begins by generating a point cloud from a set of images using a structure from motion (SfM) algorithm [3]. Using this as a basis, they create a set of Gaussians. They then optimize this set by splitting, merging, and adjusting different parameters to minimize the total loss between the rasterized result and the reference dataset. Because a key part of this optimization process is comparing the rasterized result to the references, the rasterizer must be very fast. They propose a tile based parallel CUDA rasterizer that allows realtime rasterization of millions of Gaussians with minimal error. Through the combination of the training model and the rasterizer, their system can create near-photorealistic novel views based on relatively small sets of images.

2 Motivation

2.1 Reasons for Choosing this Area

We chose this topic for our project because it appeals to the mixed strengths of our group, as well as to the desire to work on new and novel material. Valerie's main focus of interest is in computer graphics. She had also skimmed the paper to determine if it was possible to replicate. Cameron has prior experience in the domains of High Performance Computing (HPC) and machine learning optimization. Andrew was interested in diving deeper into the world of gradient descent optimization, especially as a way to manipulate 3D Gaussians.

2.2 Relevance and Importance of the Problem

As consumers continue to demand faster and higher quality graphics, new solutions must be researched. Devices that provide immersive experiences such as virtual reality and video games will be the first to adopt these new graphics solutions. Optimization in the rendering process of Gaussian Radiance Fields would provide enhanced realism in real-time rendering of complex scenes.

3 Directions of Investigation

3.1 Possible Research Approaches

In section 8 of the paper by Kerbl et al., [2] they note that one possible improvement would be to implement the optimization steps of the code entirely in C.

"The majority (~80%) of our training time is spent in Python code, since we built our solution in PyTorch to allow our method to be easily used by others. Only the rasterization routine is implemented as optimized CUDA kernels. We expect that porting the remaining optimization entirely to CUDA, as e.g., done in InstantNGP [Müller et al . 2022], could enable significant further speedup for applications where performance is essential."

Since we are writing all of our code in C anyway, we may find that we already achieve a speedup as compared to the original paper, simply by porting it to C.

Our approach will be to read the paper and get a firm grasp of the underlying principles in order to be able to implement it ourselves. There are two key aspects: the rasterization/rendering and the optimization of the Gaussians. They provide some pseudocode for the rendering side, as well as some equations for point-based alpha-blending rendering and the gradient computation.

3.2 Methodologies to be Employed

In measuring our performance, the simplest metric is training time. Nonetheless, it is equally crucial for our model to accurately represent the scene. The paper employs popular image similarity metrics, including peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), and learned perceptual image patch similarity (LPIPS) [2]. Python offers accessible libraries for these metrics, which would enable us to benchmark our newly generated images. This ensures that our model can be effectively compared to other state-of-the-art models.

4 Expected Results

4.1 Potential Outcomes

A 3D Gaussian consists of an (X, Y, Z) coordinate triple representing position, a 3x3 covariance matrix representing the stretching and scaling of the distribution, and value α for transparency. Through gradient descent, we determine the position, covariance, and α transparency of 3D Gaussian distributions to accurately match with the known images of the scene.

Algorithm 1 Gradient Descent

```
   Input: $\Theta, \hat{I}, E_p, \eta$
   Output: $\hat{\Theta}$
1: procedure GradDescent
2:     for $i = 1$ to $E_p$ do
3:         $I \leftarrow \text{rasterize}(\Theta)$
4:         $g \leftarrow \frac{1}{m}\sum_{j=1}^{m} \nabla \text{Loss}_j(I, \hat{I})$
5:         $\Theta \leftarrow \Theta - \eta g$
6:     end for
7:     return $\hat{\Theta}$
8: end procedure
```

4.1.1 Potential Outcome: Parallel Batch Gradient Descent

The batch, g, is partly computed by taking the average of the j gradients from the $\text{Loss}_j(I, \hat{I})$. Using j threads to independently compute the gradients from $\text{Loss}_j(I, \hat{I})$ and a reduction for g, we could potentially increase the performance of the algorithm. Each thread would compute the derivative of the loss function with respect to the (X, Y, Z) coordinate pair, the 3x3 covariance matrix, and the α transparency to determine the gradient. The gradient will be used to update the parameters of the image. The returned parameters build a scene entirely made out of Gaussian distributions.

4.2 Future Research Directions

One possible future research topic would be on how accurately this method can reproduce any sort of reflections. Based on the lighting model (spherical harmonic) and the data of the Gaussians, it seems like it wouldn't be possible to render reflections. Another underlying issue with this would be the ability of SfM to accurately render points on a mirror. In general, mirrors can interfere with SfM as they don't appear to be the flat surfaces that they actually are.

The paper by Kerbl et al. [2] brings up a possible research continuation that would involve attempting to convert the Gaussians into a 3D mesh. Then, the 3D mesh could be used more generally by other software.

Furthermore, the performance of the optimization steps could be improved by looking towards advanced optimization techniques [4], such as momentum-based learning (AdaGrad, RMSProp, Adam) and computationally efficient variations of the Newton Method (Conjugate Gradient, BFGS algorithm).

References

[1] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields, 2022.

[2] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering, 2023.

[3] Jaeryun Ko and Yo-Sung Ho. Point cloud generation using structure from motion with multiple view images. 2016. URL https://api.semanticscholar.org/CorpusID:45273242.

[4] Aatila Mustapha, Lachgar Mohamed, and Kartit Ali. Comparative study of optimization techniques in deep learning: Application in the ophthalmology field, jan 2021. URL https://dx.doi.org/10.1088/1742-6596/1743/1/012002.

[5] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhöfer. Deepvoxels: Learning persistent 3d feature embeddings, 2019.