Real-time Synthesis of Multiple Video Streamings based on 3D Gaussian Splatting: A Proposal

Siyan HU

October 31, 2023

1 Problem and Motivation

The project aims to develop a synthesis based on two different video streams in real-time. The challenge lies in combining or transitioning between two different video streams with different camera parameters (extrinsics and intrinsics) and as less overlapping of views as possible, while maintaining 3D visual consistency and doing so at a speed[1] that allows for real-time viewing and interaction.

The proposed approach is to use 3D Gaussian splatting[2] which can efficiently create smooth transitions within single video stream by treating each video as a point cloud in a 3D space and splatting into a continuous 3D field. The field is then rendered to produce a 2D image. With the basis of single streaming input, it is possible to add more input sources and create the synthesised scene in high efficiency.

Other possible optimisations include multiple resolution for different areas which provides a good balance between quality and performance. It can help in two ways: 1) keep the privacy of irrelevant objects in the video by downgrading the resolution; 2) Upgrade computing efficiency by rendering only once of the positions which have not changed much among frames, allowing for high-resolution rendering in focused areas and object of interest, while saving computational resources in less important or detailed areas.

2 Background - Neural Radiance Field (NeRF)

NeRF[3][4] represents the future of the research in 3D scene reconstruction, and has already been transferred into multiple applications, such as virtual roadshow, 3D Vlog, medical imaging, and slam[5].

To put it simple, the process is to query the corresponding set of colour and density by using spatial location information. In NeRF, light field is a complete representation of the collection of rays in space. For each ray, multiple points are sampled along its path in the light field. For each sampled point, a 5D coordinate is adopted which include via current point (x, y, z) radiance emits, and towards direction (θ, ϕ).

A MLP (Multi-Perception Layer) used in NeRF typically consists of several fully connected layers, each followed by a non-linear activation function ReLU (Rectified Linear Unit). The input to the MLP is the 5D coordinate we mentioned above. The output of the MLP is a 4D vector, representing the volume density sigma and RGB value (r, g, b) at that point along that viewing direction. The colour values are then blended using alpha compositing which adopts both the colour and opacity at each point, as well as the distances between the points. The result of this blending process is the colour of a single pixel in the final rendered image.

And then the results are compared with the ground truth, the network parameters are reversely optimized, and continuous iterations are made to make the radiation field close to the true value. For the full process please check Figure 1.

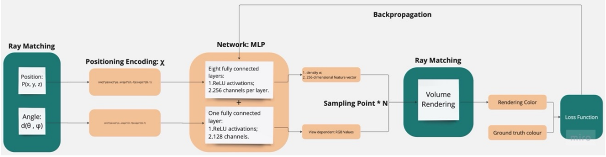

Figure 1: Working Flow of Classical NeRF (Created with Miro)

This is a departure from traditional computer graphics (explicit geometry), where scenes are typically represented using geometry or raster information, and it allows for a highly compact and efficient representation of complex 3D scenes.

3 Method - 3D Gaussian Representation and Splatting

There are several popular rendering methods for NeRF. Other than the classical volumetric rendering, Signed Distance Function (SDF) is also a very popular way. SDF is a function that gives the shortest distance between a point and a surface. It (the signed function) can differentiate between points inside and outside the surface. The function returns a positive value for points outside the surface ($SDF > 0$), zero on the surface ($SDF = 0$), and a negative value ($SDF < 0$) for points inside the surface. SDF determines that the sampling points closer to the surface have a higher colour contribution.

In NeRF each ray can potentially interact with a different number of objects in the scene which could possibly cause ray reflection and refraction. Thus computations cannot be done in parallel, which reduces the efficiency of the GPU. Also classical NeRF is run in Pytorch - CUDA environment which mostly are considered run under general-purpose computing on graphics processing units (GPGPU). While processing traditional rasterisation as we all know is fast but it is a discrete process, as well as SDF. One of the main contents of traditional rasterisation to process 3D scenes is to map three-dimensional meshes to the projection plane and do the pixelation. The projection here is called splatting[6].

In 3D Gaussian representation, a single 3D Gaussian splatting can be optimized as a small differentiable space, and multiple Gaussian can be rendered in parallel rasterisation. This can be seen as a delicate balance between differentiability and discreteness.

But what is 3D Gaussian? 1D Gaussian, as we know, is the normal distribution and the shape is a simple bell symmetric curve as shown in Figure 2(a).

A set of μ and σ can determine a 1D Gaussian distribution function, and then determine

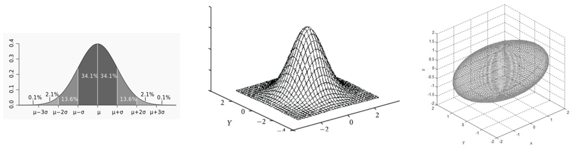

Figure 2: Fig 2(a) Normal distribution: 1D Gaussian[7]; 2(b) Visualisation of a 2D Gaussian model[8]; 2(c) Visualization of a 3D Gaussian model[9]. Here we have $\mu_x = 0$, $\mu_y = 0$, $\mu_z = 0$ and $\sigma = 1$.

a 1D line segment $[\mu - 3\sigma, \mu + 3\sigma]$. By changing μ and σ can express a line segment on the 1D number axis. Similarly, if it is expanded to a three-dimensional Gaussian distribution it is an ellipsoid, as shown in Figure 2(c). The basic elements of the 3D Gaussian construction can cover enough diverse geometries, as defined below,

$$G_s(x) = e^{-\frac{1}{2}x^T\Sigma^{-1}(x-\mu)}$$

Here $x = [a, b, c]^T$ is the 3D column vector coordinates because 3D Gaussian and ellipsoid are isomorphic in geometry, which can be obtained by scaling and rotating the ball along the axis. μ is the ellipsoid centre; the covariance matrix $\Sigma = RSS^TR^T$; S is scaling and R is rotation. If some 3D Gaussian of a point generated is too large to fully fit the details of that point, the point is segmented and expressed by two points; if the points at a certain point are too dense, multiple 3D Gaussian represents are merged into one 3D Gaussian, as shown in Figure 3.

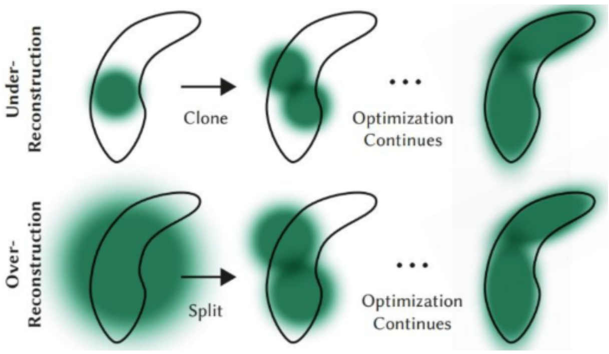

Figure 3: Auto-adjust point cloud distribution according to gradient[2].

The axial integral of 3D Gaussian is equivalent to that of 2D Gaussian, which mathematically gets rid of the limitation of sampling amount (unlike the discrete accumulation to fit continuous Integral in classical NeRF). The amount of calculation is determined by the number of Gaussian, and Gaussian can be quickly rendered in parallel using the rasterisation pipeline. Though in the paper they did not provide details of the way to transfer 2D Gaussian to pixel, it is basically a rasterisation process to transfer plane triangle is transformed into pixels through algorithms such as scan lines and barycentric coordinate methods. The 3D Gaussian splatting requires a transferring of images to 3D structure of motion (SfM[10]) points as prerequisite, which estimates the 3D structure of a scene from a set of 2D images.

4 Schedules

The project of synthesizing one source of video frames to another in real time based on 3D Gaussian splatting contains the following steps:

1. 3D Reconstruction: The first step in this process would be to reconstruct the 3D geometry of the scene from the input video with SfM.

2. SfM Point Cloud Generation: Once the 3D structure of the scene is obtained, a point cloud representation of the scene can be generated. Each point in the point cloud represents a point in the 3D scene, and include additional data such as colour information.

3. 3D Gaussian Splatting: The point cloud data is then processed using 3D Gaussian splatting. Each point is splatted into a Gaussian distribution that adds smoothly to its neighbours, resulting in a smooth 3D field that represents the original point cloud data.

4. Rasterisation: The 3D field is then rasterised to produce a 2D image. This involves casting rays from the camera position through each pixel in the 2D image, and sampling the 3D field along each ray to determine the pixel's colour.

5. Video Synthesis: The rasterization process is repeated for each frame in the output video, possibly with a different camera position or other parameters for each frame. This generates a sequence of 2D images that form the output video.

6. Output Streaming: The synthesized video is then streamed to the recipient. This could involve encoding the video in a suitable format, transmitting it over a network, and decoding it on the recipient's end.

5 Expected Result

In the project a practical tool could be developed and can be used for real-time video synthesis in various contexts, such as live streaming, teleconferencing, virtual reality, and video games. Also an evaluation of the effectiveness and limitations of the proposed 3D Gaussian splatting technique for real-time video synthesis will be conducted for future research.

References

[1] 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, July 2022.

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

[3] 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. In *ECCV*, 2020.

[4] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, SIGGRAPH '23, 2023.

[5] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R. Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[6] M. Zwicker, H. Pfister, J. van Baar, and M. Gross. Ewa volume splatting. In *Proceedings Visualization, 2001. VIS '01.*, pages 29–538, 2001.

[7] Wikipedia contributors. Normal distribution — Wikipedia, the free encyclopedia, 2023. [Online; accessed 30-October-2023].

[8] S. Dinesh and P. Radhakrishnan. Linear and nonlinear approach for dem smoothening. *Discrete Dynamics in Nature and Society*, 2006(1):Article ID 63245, 10 p.–Article ID 63245, 10 p., 2006.

[9] Jae-Han Park and et al. Spatial uncertainty model for visual features using a kinect™ sensor. *Sensors (Basel, Switzerland)*, 12, 2012.

[10] Noah Snavely, Steven M. Seitz, and Richard Szeliski. *Photo Tourism: Exploring Photo Collections in 3D*. Association for Computing Machinery, New York, NY, USA, 1 edition, 2023.