SplatCraft: Diffusion-based Sketch to 3D

Lakshya Gupta, Anannya Popat, Shreya Vedhanarayanan

Team Members

1. Lakshya Gupta (1009865013)
2. Anannya Popat (1009792547)
3. Shreya Vedhanarayanan (1010269898)

Main Idea

The main idea is to generate a 3D scene reconstruction of an input image (like a sketch) by first generating sparse multiviews from a diffusion model [1] and then feeding these multiviews as input to a Gaussian Splatting model for 3D reconstruction of the scene. We are basically interested in working on a project which combines generative models like diffusion as well as 3D reconstruction through Gaussian Splatting.

The objectives more specific to the course, i.e, optimization, can be incorporated in both the diffusion model and the gaussian splatting model, but we are more interested in exploring model optimization of Gaussian Splatting because it's a newer algorithm. That's why we will probably only look into the diffusion model-specific optimization if we are not able to generate decent scene multiviews (component 2 described below) and have to use the existing Sync Dreamer codebase.

Individual Component Efforts in Mind

<u>For Sync Dreamer (Diffusion Model) :</u>

1. **Model Optimization & Acceleration on Diffusion Model**
   The original paper does not talk about any model optimization or acceleration techniques being used. Their conclusion says one of the limitations is that it's not able to generate a lot of dense views for the input image and requires a lot more GPU to be able to do that. We want to use some of the model compression and acceleration techniques discussed in class to their original model.

   The paper builds on the Zero123 paper which used a fine-tuned Stable Diffusion model.

We can try:

a. Model Pruning
b. Quantization and weight clustering
c. ONNX

We want to try the different techniques on the next component described below, but in the event that it does not work, we will try to work with the original Sync Dreamer codebase.

2. **Extending Multiview from single-object to a whole scene**
From an application standpoint, we also want to extend their efforts of multiview image generation for a single object to a scene which has multiple objects.

Some ideas to attempt doing this:

a. First, we run some sort of object detection on the input images to detect the different objects. Then, we use CNN or some feature extractor to get the latent representation of each cropped object in the image.

b. Using some sort of scene graph to capture the different spatial relationships between objects in the scene. Embedding these geometric/spatial information in the scene representation (which can be generated using graph neural networks?)

c. Synchronized Diffusion Models: Leveraging diffusion models, use these latent representations to generate new views of each object or scene component in a synchronized manner. This means adjusting the generation process so that the spatial relations and interactions between objects remain consistent across different views. Techniques like cross-attention mechanisms could be employed to ensure that the generation of one object takes into account the presence and state of others, maintaining a coherent scene structure.

For Gaussian Splatting:

Model optimization of gaussian parameters would be done with the aid of either vector quantization [2] or sensitivity-based vector quantization and quantization-aware training [3].

1. **Methodology by [2] for Parameter Compression:**
a. Vector Quantization with K-means Algorithm of Gaussian Parameters
b. Compression of Indices using Run-Length Encoding (RLE)

    c. **Results:** Reduces the storage cost of Gaussian Splatting by 20X

2. **Methodology by [2] for Parameter Compression:**
    a. Sensitivity-Aware Vector Clustering (identifying which aspects of the scene are less noticeable to viewers and compressing those more aggressively)
    b. Quantization-Aware Training
    c. **Limitation:** This paper also explored the compression of spatial positions or 3D coordinates of each Gaussian Splat. Compressing these positions involves reducing the precision with which these coordinates are stored. The idea explored was to quantize these positions to a lattice, a grid-like structure that limits the possible locations to a predefined set of points. This approach reduces the variability (and hence the storage) of position data but at the risk of "snapping" splats to the nearest grid point, potentially distorting the scene's geometry or introducing artifacts.
    d. **Results:** Reduces the storage cost of Gaussian Splatting by 31X and 4X increase in rendering speed

Conclusion:
Upon comparing the two approaches, our preference leans towards the method described in [3], primarily because it employs Sensitivity-Aware Vector Clustering, which yields superior results. Nonetheless, we plan to deviate from the paper's strategy of compressing spatial positions, as this can introduce noise or artifacts in the rendered images, potentially compromising quality.

Things we need feedback on

1. **Scope of the multiview extension described above**
    a. Do you think the ideas we have in mind for extending from a single object to a whole scene looks doable?
    b. If the above is not possible, is experimenting with different model compression techniques discussed in class for the existing Sync Dreamer diffusion model architecture good enough?

2. **Hardware Limitations**
    a. Will running diffusion models and gaussian splatting with the limited compute be possible? None of us have good laptops and we're only going to be relying on the CS compute-servers.

3. **Compression of GSplat Model:**

a. Which proposed solution for model compression do you think would be good to use given considerations like time constraint and the complexity of the process?
b. If we plan to concretely pursue the methodology suggested by [3], do you think it would be a good idea to further explore the compression of spatial positions as it introduces artifacts?

Dataset

For the dataset, we are looking towards the following:

1. MVImgNet [5]
2. OpenSketch [4]

References

1. Sync Dreamer (https://arxiv.org/abs/2309.03453)
2. Compact3D: Compressing Gaussian Splat Radiance Field Models with Vector Quantization (https://www.researchgate.net/publication/377266663_Compact3D_Compressing_Gaussian_Splat_Radiance_Field_Models_with_Vector_Quantization)
3. Compressed 3D Gaussian Splatting for Accelerated Novel View Synthesis (https://arxiv.org/abs/2401.02436)
4. SketchDesc: Learning Local Sketch Descriptors for Multi-view Correspondence (https://arxiv.org/pdf/2001.05744.pdf)
5. MVImgNet: Large-scale dataset of Multiview Images [Dataset] (https://gaplab.cuhk.edu.cn/projects/MVImgNet/)