Final Project: Mini-Movie Making with dynamic 3DGS

Kim Chae-eun (2023100064)

Korea University, Department of Computer Science and Engineering

Introduction to Computer Vision

Moon Gyeongsik

December 7, 2025

1. Introduction

This project aims to create a dynamic mini-movie using 3D Gaussian Splatting (3DGS) technology. We reconstructed one scene and two objects (Kirby and Cat) from real-world videos, and composed them to generate a 14-second cinematic sequence combining camera motion and object animation.

The entire pipeline consists of COLMAP-based Structure-from-Motion, 3DGS reconstruction using gsplat, and depth-aware composition with dynamic rendering. All animations are defined in JSON files to ensure reproducibility.

2. Implementation

2.1. Data Acquisition and Preprocessing

Three videos were captured using a mobile phone to reconstruct the scene and objects. The scene video was recorded at Hana Square in Korea University, filming the KU statue in a semi-circular trajectory to ensure sufficient parallax and coverage. For the objects, two personal figurines, Kirby and Cat, were used and captured with 360-degree rotation to obtain complete geometric information from all angles.

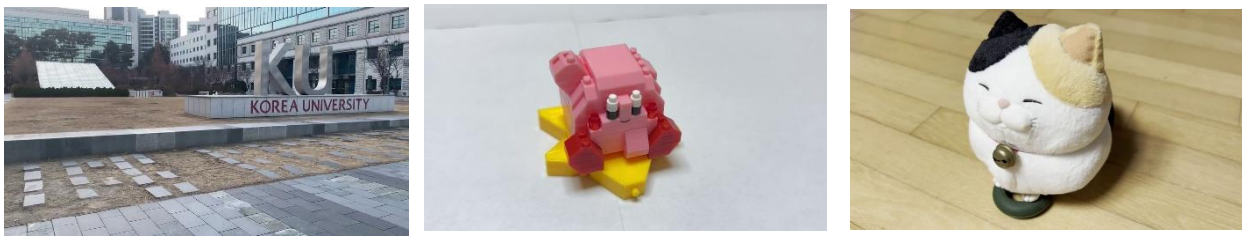

Figure 1. Sample frames from scene, Kirby, and Cat videos

When the Cat object was initially filmed, the yellowish floor was unintentionally reconstructed along with the object. To address this issue, video segmentation using SAM (Segment Anything Model) was attempted, but it failed to accurately isolate the object. Based on this experience, the Kirby object was subsequently filmed on a completely white background to minimize background interference and facilitate cleaner object extraction in later stages.

All videos included a 3-second identification frame showing student ID and name at the beginning to comply with academic integrity requirements.

I did not know the correct spelling of the Kirby character from the game, so the actual code implementation uses 'curby' instead of 'kirby'... I learned the correct spelling of Kirby while writing this report.

2.2. Structure-from-Motion with COLMAP

After extracting frames from the captured videos, Structure-from-Motion (SfM) was performed using COLMAP. COLMAP generates camera poses and a sparse 3D point cloud for each frame through three stages: feature extraction, feature matching, and sparse reconstruction.

Frames were extracted at 2fps for the scene video and 6fps for the object videos. COLMAP was executed independently for the scene and each object, and the resulting camera intrinsics and extrinsics were used as initialization data for subsequent 3DGS training. Due to the 360-degree rotation capture, stable pose estimation was achieved from all angles for the objects.

2.3. 3D Gaussian Splatting Reconstruction

Using the camera poses and sparse point cloud obtained from COLMAP as initial values, 3D Gaussian Splatting training was performed using the gsplat library. 3DGS represents 3D space with numerous 3D Gaussian primitives, where each Gaussian has parameters including position (mean), covariance, color, and opacity.

Training was conducted independently for the scene and each object (Kirby, Cat) for a total of 30,000 iterations. During training, Gaussian parameters were optimized to minimize photometric loss (L1 + SSIM). Additionally, adaptive density control was applied to dynamically adjust the number of Gaussians, adding Gaussians in areas lacking detail and removing unnecessary ones.

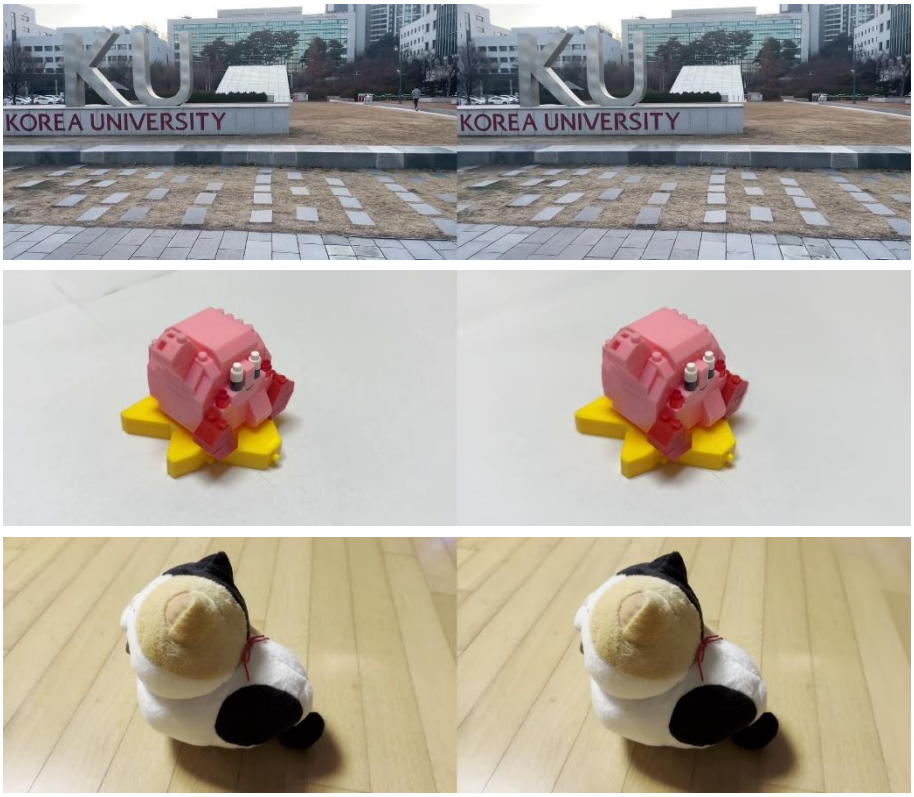

Figure 2. Trained 3DGS models - validation rendering results of scene, Kirby, and Cat (left: GT, right: rendered)

The trained models were saved as checkpoint files and subsequently used for scene composition and dynamic rendering. Since the objects were trained with their backgrounds, a filtering process to extract only the objects was required in the next step.

2.4. Scene Composition with Depth-Aware Occlusion

The most significant challenge in composing the trained scene and objects was removing the background from the objects. The initial attempt to separate the background using SAM for video segmentation failed, resulting in objects being trained with their backgrounds. Consequently, when the objects were composed into the scene and rendered, they appeared hazy, as if covered by yellow dust.

Figure 3. Composition result before filtering - Kirby covered by background Gaussians

Particularly for Cat, the high saturation of the floor made the object completely invisible at first, leading to the mistaken belief that 3DGS training had failed. However, by rotating the object in various directions, it was confirmed that the Cat object existed at the origin, surrounded by Gaussians reconstructed from the background.

To address this issue, a multi-stage filtering strategy was applied. First, the object was centered at the origin, followed by density-based outlier filtering to remove sparse Gaussians far from the center. Next, color filtering was applied based on the core object region to eliminate low-saturation Gaussians similar to background colors (yellowish/white). Thresholds of percentile 40% for Kirby and percentile 70% for Cat were used, adjusted to match each object's characteristics.

Figure 4. Composition result after filtering - cleanly extracted objects composed into the scene

The filtered objects were rendered together with the scene's Gaussians through depth-aware rasterization. The 3DGS rendering pipeline naturally implements occlusion by sorting each Gaussian by depth from the camera and performing alpha blending, without requiring additional post-processing. Object position, rotation, and scale were controlled via 4×4 transformation matrices.

2.5. Dynamic Animation and Cinematic Rendering

The final mini-movie presents a narrative where Kirby appears, spins and transforms into Cat, which then grows enormous and rides away on a cloud. The entire video is 14 seconds long at 30fps, consisting of 420 frames total, divided into five sequences, each with independent cinematic motion.

Sequence 1: Camera Movement (2s, 60 frames)
The camera moves through the scene to introduce the environment. It smoothly transitions from the initial viewpoint to another, with the Kirby object statically placed at position [-0.8, 0, 0].

Sequence 2: Kirby Spin and Morph (3s, 90 frames)
Kirby rotates three times (1080 degrees) around the yaw axis in place, while gradually becoming transparent and disappearing. At the same location, the Cat object slowly appears, replacing Kirby to create a morphing effect.

Sequence 3: Cat Scale-Up (4s, 120 frames)
The Cat slowly enlarges from 0.5× to 2.0× scale, becoming a dominant presence in the scene. The 4× size change generates a dramatic visual effect.

Sequence 4: Orbital Camera Motion (3s, 90 frames)
The camera rotates around the Cat in a 120-degree arc from -60 to +60 degrees. It showcases the enlarged Cat from various angles, emphasizing the cinematic feel.

Sequence 5: Cat Exit and Camera Return (2s, 60 frames)
The Cat moves to position [1, 0, 2], riding away on a cloud and exiting the scene, while the camera returns to the initial viewpoint to conclude the video.

All camera trajectories and object motions were defined in JSON files. The camera trajectory includes 4×4 camera-to-world transformation matrices for each frame, while object motion contains translation, rotation (Euler angles), scale, and visibility for each frame. This ensures complete reproducibility, allowing the same result to be rendered at any time from the same JSON files.

Rendering was performed by composing the Gaussians of the scene and objects for each frame, with output at 1280×720 resolution. Rendering the entire 420 frames took approximately 10 minutes.

3. Conclusion

Through this project, a complete pipeline was implemented to reconstruct real environments and objects with 3D Gaussian Splatting and compose them to generate dynamic animations.

Up to the 3D GS reconstruction stage, progress was relatively fast as a publicly available 3D GS implementation (gsplat) could be used. Both the scene and objects were reconstructed with high quality, and validation rendering results achieved visual quality nearly indistinguishable from the original images (Figure 4). However, unexpected difficulties were encountered during the actual composition of objects into the scene.

The biggest challenge, as explained in Section 2.4, was extracting pure objects from those trained with backgrounds. After automatic segmentation using SAM failed, a manual filtering strategy combining density-based filtering and color filtering had to be developed. This process required much trial and error to find appropriate thresholds for each object. While satisfactory results were ultimately achieved , future projects would benefit from carefully selecting objects that can be well-segmented by SAM to produce visually cleaner animations.

The final mini-movie consists of 14 seconds and 420 frames, successfully implementing a narrative where Curby transforms into Cat, grows enormous, and rides away on a cloud. Thanks to depth-aware rendering, occlusion between objects and the scene was handled naturally, and the JSON-based animation system provided complete reproducibility.

Limitations of this project include the manual tuning required for object filtering and the support for only rigid transformations, making non-rigid deformation impossible. Future work could overcome these limitations by introducing learning-based segmentation or 4D Gaussian Splatting techniques.

Nevertheless, the experience of creating an animation from start to finish with my favorite characters was immensely enjoyable and provided a valuable opportunity to directly experience both the potential and limitations of 3DGS technology.