EgoSplat: Exo-centric to Ego-centric view translation using Gaussian Splatting

Yogeshwar Boopathy
Joint MSc Data Science and Artificial Intelligence
University of Birmingham
yxb340@student.bham.ac.uk

Abstract

Changing from a third-person to a first-person perspective is not as straightforward as pressing a button in video games. In real life, translating exocentric (third-person) to egocentric (first-person) perspectives poses significant challenges. Despite advances using GANs, diffusion models, and other generative methods, existing techniques struggle with large viewpoint shifts, occlusions, and the loss of fine details. We propose a novel approach leveraging Gaussian splatting, a rasterization-based technique that efficiently models scenes as 3D Gaussian distributions. This method achieves photorealistic rendering from sparse image datasets, reducing distortions and preserving intricate details, especially in complex hand-object interaction scenarios. Quantitative evaluations using SSIM and PSNR on real-world datasets demonstrate significant improvements over traditional methods. Our approach advances the technical capabilities of exo-to-ego view translation, paving the way for immersive applications in robotics, virtual and augmented reality, and human-computer interaction.

1. Introduction

Understanding and recreating how actions appear from a first-person perspective is critical for several cutting-edge applications, including robotics, virtual reality (VR), augmented reality (AR), and human-computer interaction (HCI). Egocentric views allow systems to analyze tasks from an actor's vantage point, making it possible to develop better assistive technologies, immersive VR/AR experiences, and autonomous systems capable of interpreting human actions. However, transitioning from exocentric (third-person) to egocentric (first-person) perspectives is not straightforward. Unlike video games, where a simple button press shifts viewpoints, this task in the real world involves significant challenges such as handling occlusions, drastic spatial differences, and preserving fine-grained details. Addressing these challenges is crucial, as high-quality exo-to-ego translations are foundational for training more intuitive and effective interactive systems.

Several approaches have been proposed to solve this problem. Generative models such as Generative Adversarial Networks (GANs) [19, 28, 29, 45, 55] and diffusion-based frameworks [30] have demonstrated potential in similar tasks, such as image-to-image translation and novel view synthesis. For example, Exo2Ego uses a two-stage generative approach with high-level structure transformation and pixel-level refinement. However, generative methods often struggle to handle the drastic viewpoint changes between exocentric and egocentric perspectives. They rely on large, paired datasets and are sensitive to sparsely overlapping views, leading to issues with realism and consistency. Neural rendering methods, such as Neural Radiance Fields (NeRF) [31], have gained attention for their ability to synthesize photorealistic views. While NeRF excels in interpolating between dense viewpoints, it performs poorly in scenarios requiring large extrapolations or sparse data inputs, such as exo-to-ego translation.

To address these limitations, we propose a novel framework as shown in Figure. 1 that focuses on Gaussian Splatting [9, 23, 24, 26, 33], a rasterization-based technique that efficiently represents scenes as 3D Gaussian distributions. The key insight of our approach is to model scenes explicitly through adaptive Gaussians that encode position, covariance, and opacity. Gaussian splatting has several key advantages over generative and neural rendering methods. Unlike generative models, it does not require large paired datasets and handles occluded regions more effectively through its inherent adaptability. Compared to NeRF, Gaussian splatting avoids reliance on dense inputs or precise camera parameters while maintaining high-quality output. Additionally, we introduce a probabilistic sampling strategy during optimization, which prioritizes views closer to the ego-centric perspective. This ensures that the model focuses on learning critical details relevant to the task, improving the alignment and quality of generated ego-centric views.

Our framework achieves significant improvements over traditional methods, as validated through quantitative metrics such as the structural similarity index (SSIM) and the

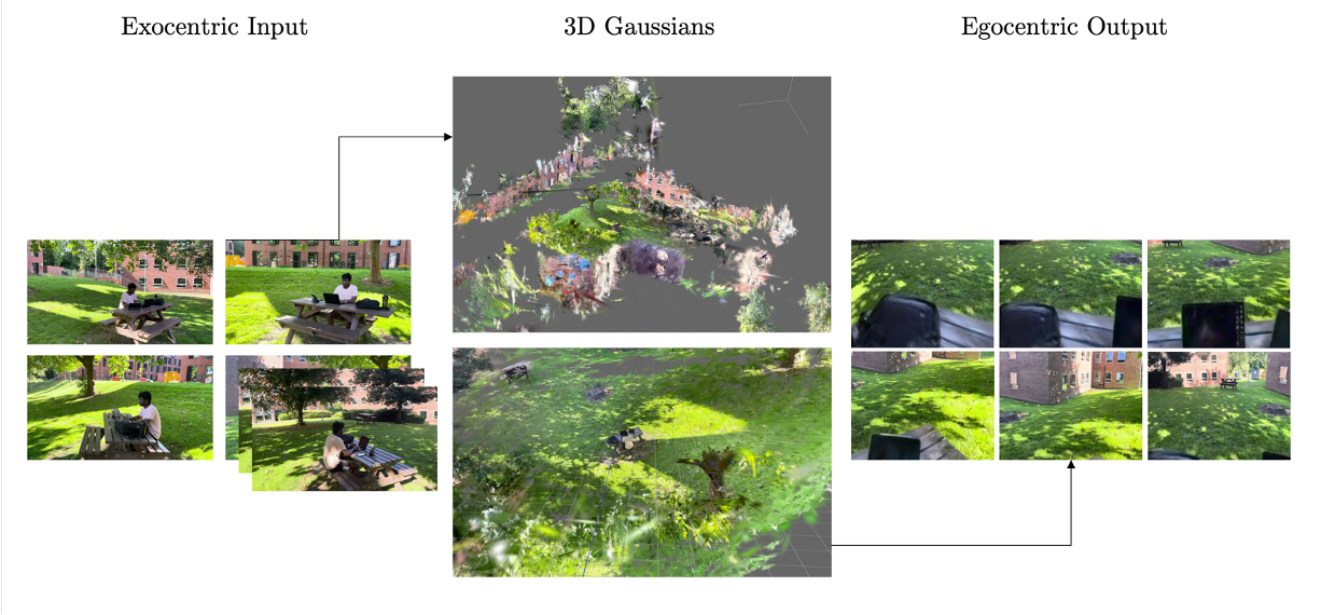

Figure 1. **Overview** Given a set of exo-centric input images, EgoSplat reconstructs a 3D radiance field parameterized via 3D Gaussian primitives. This yields an explicit 3D representation that is further optimized towards egocentric outputs

peak signal-to-noise ratio (PSNR). Compared to baseline approaches, our method produces superior results in handling large viewpoint shifts and complex hand-object interactions. In our real-world datasets, the proposed Gaussian splatting-based method consistently outperforms GAN-based frameworks like Exo2Ego in perceptual quality while maintaining computational efficiency.

Our approach builds upon and extends ideas used in related problems. Neural rendering techniques, such as NeRF and Scene Representation Networks (SRN)[42], have focused on tasks like novel view synthesis but assume dense inputs and known camera poses. Similarly, generative frameworks like Pix2Pix [29] and Parallel GAN have been applied to specific image synthesis tasks, but fail to generalize well to drastic perspective transformations. Our method leverages the flexibility of Gaussian Splatting, combining the robustness of explicit scene representations with the adaptability of probabilistic optimization, making it uniquely suited for exo-to-ego view translation.

2. Related Work

Translating views from exocentric (third-person) to egocentric (first-person) perspectives is a relatively new but rapidly developing area within computer vision. This task presents unique challenges due to the drastic changes in viewpoint and the potential for occlusions between the two perspectives. While existing methods for novel view synthesis have shown promising results in generating new views from similar viewpoints, they often encounter limitations when applied to the complexities of exo-to-ego translation.

2.1. Geometry-Based Methods for Novel View Synthesis

Geometry-based methods, like Neural Radiance Fields (NeRF) and Scene Representation Networks (SRN), have achieved remarkable results in novel view synthesis but face limitations in exo-to-ego scenarios.

NeRF excels at interpolating views within a densely sampled set of viewpoints. It represents a scene as a continuous function that maps 3D coordinates and viewing directions to color and density values, enabling the synthesis of photorealistic novel views. However, NeRF's reliance on dense viewpoints makes it less effective when extrapolating to significantly different views, such as those encountered in exo-to-ego translation. Exo-to-ego translation often involves viewpoints where the ego-centric view includes content occluded in the exo-centric view, requiring the model to infer missing information, a task NeRF struggles with. Additionally, NeRF requires precise camera parameters for each input image, which may be unavailable or difficult to obtain in real-world exo-to-ego scenarios.

SRN represents scenes implicitly as continuous neural functions that map 3D coordinates to scene properties. It uses a differentiable rendering function to generate novel views from arbitrary camera viewpoints without explicit 3D data. While SRN has shown potential in reconstructing scenes from multiple viewpoints, it heavily relies on accurate camera pose information and struggles to scale to complex scenes with high geometric variation. This reliance on pre-

cise camera poses and dense input data, much like NeRF, limits its applicability in real-world exo-to-ego scenarios where such information might be unavailable or unreliable.

2.2. Generative Models for Cross-View Translation

Generative models, particularly Generative Adversarial Networks (GANs), offer an alternative approach to view translation by learning the mapping between input and output images. While they can handle more considerable viewpoint changes than geometry-based methods, they face challenges preserving fine details and ensuring consistency between generated and authentic images.

Parallel GAN (P-GAN), specifically designed for exo-to-ego image generation, utilizes two GANs in parallel, each responsible for one view. It incorporates hard-sharing of layers in the early stages to learn common high-level features and employs U-Net architectures [39] for generators to preserve high-resolution information. PatchGAN discriminators, focusing on local texture realism, and a combination of cross-cycle loss, contextual feature loss, and reconstruction loss further enhance the quality and consistency of generated images. However, P-GAN faces challenges in handling complex interactions like hand-object manipulations and struggles with significant viewpoint shifts. Its reliance on paired exo-centric and ego-centric datasets, which are often difficult to obtain, further limits its practicality.

The Exo2Ego framework [30] adopts a two-stage approach to address the complexities of exo-to-ego video translation. The first stage utilizes a transformer-based encoder-decoder architecture to predict the approximate hand-object interaction layout in the ego-centric view from an exo-centric frame. This stage captures high-level structural information, focusing on hand positioning and interactions. The second stage employs a diffusion-based pixel hallucination process to refine the rough layout and generate photorealistic ego-centric frames. While this method is effective in controlled environments, it assumes consistency in camera angles and positions, limiting its generalization to diverse and dynamic real-world scenarios. Additionally, the reliance on pre-trained models like VAEs and the computational inefficiency of diffusion models makes it less suitable for real-time applications.

This work addresses the limitations of existing methods by employing 3D Gaussian Splatting for exo-to-ego view translation. Gaussian Splatting, representing scenes using a collection of 3D Gaussians, offers several advantages over both geometry-based and generative models. Firstly, Gaussian Splatting offers flexibility and scalability in scene representation. Unlike voxel-based or mesh representations, it can handle complex scenes with varying levels of detail efficiently. By adjusting the density and distribution of Gaussians through adaptive densification techniques, Gaussian Splatting can represent scenes without being constrained by fixed resolutions. This flexibility makes it well-suited for exo-to-ego translation, where the ego-centric view may require the model to synthesize details not present in the exo-centric view. Secondly, 3D Gaussians with anisotropic covariance matrices allow for a more robust representation of scene geometry, particularly in areas with occlusions or missing information. This robustness is crucial for exo-to-ego scenarios, where the ego-centric view may contain elements occluded in the exo-centric view. Thirdly, Gaussian Splatting leverages fast differentiable rasterizers [6, 22, 27, 35], enabling real-time rendering capabilities. This efficiency is essential for interactive applications such as VR and AR, where responsiveness to viewpoint changes is critical. Finally, this work introduces a probabilistic sampling strategy during training that prioritizes images closer to the ego-centric viewpoint, focusing the optimization process on the most relevant viewpoints for accurate ego-centric view generation. By combining these advantages, this work presents a novel and effective approach to exo-to-ego view translation, pushing the boundaries of novel view synthesis toward more challenging and human-centric applications.

3. Background: 3D Gaussian Splatting (3DGS)

3D Gaussian Splatting (3DGS) utilizes anisotropic 3D Gaussians as explicit and continuous scene representations. By combining this representation with a fast, differentiable rasterization pipeline, 3DG-S enables photorealistic novel view synthesis with minimal training times.

3.1. 3D Gaussians as Scene Representation

Each 3D Gaussian $G(x; \mu, \Sigma)$ is defined by its mean $\mu \in \mathbb{R}^3$, which represents its center, and its covariance matrix $\Sigma \in \mathbb{R}^{3\times3}$, which describes its anisotropic spread and orientation:

$$G(x; \mu, \Sigma) = \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right),$$

where $x \in \mathbb{R}^3$ is a 3D point in the scene.

3.1.1. Covariance Decomposition

To ensure Σ remains positive semi-definite during optimization, it is parameterized as:

$$\Sigma = RSS^\top R^\top,$$

Where:

- $R \in SO(3)$ is a rotation matrix parameterized using a unit quaternion q, and
- $S = \text{diag}(s_1, s_2, s_3)$ is a diagonal scaling matrix, with $s_1, s_2, s_3 > 0$.

This decomposition ensures numerical stability and efficient optimization of Σ.

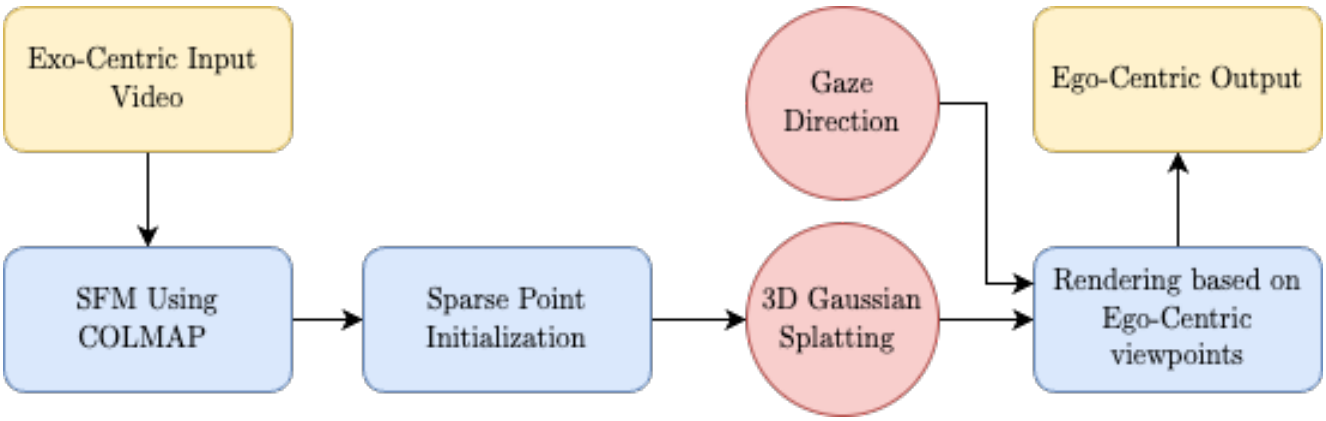

Figure 2. Overview of the proposed methodology

3.1.2. Radiance and Opacity

Each 3D Gaussian is associated with:

1. Spherical Harmonics (SH) [1, 7, 15, 34, 43, 44]: Representing view-dependent colour. The radiance $c(\omega)$ is expressed as:

$$c(\omega) = \sum_{l=0}^{L} \sum_{m=-l}^{l} c_{l,m} Y_{l,m}(\omega),$$

where $Y_{l,m}$ are spherical harmonics basis functions, $c_{l,m}$ are SH coefficients, ω is the viewing direction, and L is the SH order.

2. Opacity α: A scalar in $[0, 1)$ that determines transparency during rendering.

3.2. Differentiable Rasterization

For novel view synthesis, the 3D Gaussians are projected into 2D space as 2D Gaussians using the camera's viewing transformation. The projection maps both the mean μ and covariance Σ into the image space.

3.2.1. 3D to 2D Projection

The transformation of the 3D Gaussian into camera coordinates is given by:

$$\Sigma' = J\Sigma J^\top, \quad \mu' = W\mu,$$

Where:
- J is the Jacobian matrix of the affine transformation,
- W is the camera viewing transformation matrix.

The covariance matrix in 2D, Σ_{2D}, is obtained by discarding the third row and column of Σ'. Similarly, the 2D mean μ_{2D} is computed by projecting μ' onto the image plane:

$$\mu_{2D} = K(R_c\mu + t_c),$$

where K is the intrinsic camera matrix, R_c is the camera rotation matrix, and t_c is the camera translation vector.

The resulting 2D Gaussian in the image plane is:

$$G_{2D}(x; \mu_{2D}, \Sigma_{2D}) = \exp\left(-\frac{1}{2}(x - \mu_{2D})^\top \Sigma_{2D}^{-1}(x - \mu_{2D})\right),$$

where $x \in \mathbb{R}^2$ is a pixel coordinate in the image.

3.3. Alpha Blending

To render the final image, the contributions of overlapping Gaussians are blended using alpha blending. The color $C(x)$ of a pixel x is computed as:

$$C(x) = \sum_{i=1}^{N} \alpha_i' c_i G_{2D,i}(x; \mu_{2D,i}, \Sigma_{2D,i}),$$

where:
- $\alpha_i' = \alpha_i G_{2D,i}(x; \mu_{2D,i}, \Sigma_{2D,i})$ is the adjusted opacity,
- c_i is the view-dependent color of the i-th Gaussian.

The blending ensures smooth transitions between overlapping Gaussians and creates photorealistic rendering.

3.4. Optimization and Rendering Efficiency

3DGS uses a highly optimized rasterization pipeline for efficient training and rendering. The optimization involves minimizing the difference between the rendered and ground-truth images using pixel-wise and perceptual losses.

3.4.1. Rendering Pipeline

Depth Sorting [4, 12, 32, 50]: Gaussians are depth-sorted to ensure correct rendering order. The depth of a Gaussian in the camera coordinate system is:

$$\text{Depth}_i = R_c^\top \mu_i + t_c.$$

Tile-Based Rasterization: The image is divided into tiles, and Gaussians overlapping each tile are processed in parallel. This minimizes memory overhead and maximizes computational efficiency.

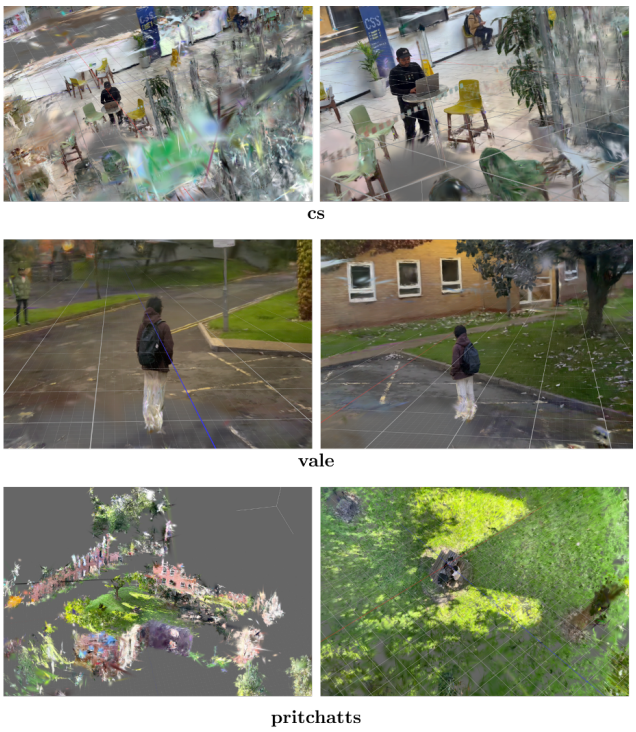

Figure 3. Shown here are 3 different scenes 'cs', 'vale' and 'pritchatts' rendered in 3D using Gaussian Splatting

Loss Function: The optimization minimizes a combination of losses.

Pixel-wise loss:

$$\mathcal{L}_{\text{pixel}} = \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} \|I_{\text{rendered}}(i,j) - I_{\text{ground truth}}(i,j)\|^2,$$

where H and W are the image dimensions.

Perceptual loss [20, 21, 53, 54]: Using feature maps extracted from a pre-trained network.

4. Method

Our framework, as shown in Figure. 2 integrates Structure from Motion (SfM) [8, 40, 41, 51], 3D Gaussian Splatting (GS), object-aware gaze target detection, and a probabilistic sampling strategy to achieve photorealistic exocentric-to-egocentric (exo-to-ego) view translation. The methodology ensures efficient training by using gaze-guided camera pose estimation and prioritizing optimization near ego-centric viewpoints.

4.1. Structure from Motion for Scene Initialization

We begin with a set of exo-centric images $\{I_k\}_{k=1}^{K}$ captured from diverse viewpoints. Using Structure from Motion (SfM), we reconstruct the camera extrinsics $[R_k|t_k]$ [3, 17, 18, 47], representing rotation and translation parameters, along with a sparse 3D point cloud $\mathcal{P} = \{X_i\}_{i=1}^{N}$. These 3D points, derived from the image projections, form the basis for initializing the scene geometry:

$$x_k = K_k \cdot (R_k X + t_k),$$

where K_k is the intrinsic camera matrix. The sparse point cloud provides the input for Gaussian Splatting, which builds a continuous representation of the scene geometry and radiance.

4.2. Scene Representation with Gaussian Splatting

The sparse 3D point cloud is converted into a 3D Gaussian representation as shown in Figure. 3. Each point X_i is modeled as a 3D Gaussian G_i, parameterized by its mean $\mu_i = X_i$, covariance matrix Σ_i, view-dependent color c_i, and opacity α_i. Gaussian Splatting uses differentiable rasterization to project these 3D Gaussians into the 2D image plane, ensuring efficient rendering of novel views:

$$\mu_{\text{2D},i} = K \cdot (R_k \mu_i + t_k),$$

where $\mu_{\text{2D},i}$ represents the projected mean in the image plane. This process enables the rendering of high-fidelity images while maintaining computational efficiency.

4.3. Object-Aware Gaze Target Detection

The task of synthesizing ego-centric views requires precise estimation of the observer's focus within the scene to accurately align the generated perspectives with the viewer's visual context. To achieve this, the framework integrates object-aware gaze target detection [46], which estimates the observer's gaze direction and identifies the specific region or object being focused on within the scene. This component combines head pose estimation, contextual object detection, and transformer-based refinement to achieve robust gaze target prediction.

The gaze estimation pipeline operates in three key stages: (1) detection of the head position and orientation, (2) computation of the gaze direction, and (3) determination of the gaze target in 3D space.

4.3.1. Head Position and Orientation Detection

The system detects the observer's head position $\mathbf{h}$ and head orientation $\mathbf{h}_{\text{ori}}$ from the input exocentric images. These parameters provide an initial estimation of the viewer's perspective within the scene.

- Head Position $\mathbf{h}$: The spatial coordinates of the observer's head in the scene.
- Head Orientation $\mathbf{h}_{\text{ori}}$: A vector or quaternion representing the rotation of the head, indicating its facing direction.

Head detection is performed using a pre-trained deep learning model, such as a convolutional neural network (CNN), trained on human pose datasets.

4.3.2. Gaze Direction Refinement

The gaze direction $\mathbf{g}$ is calculated based on the head orientation $\mathbf{h}_{\text{ori}}$, but this initial estimate is often insufficient for precise gaze target prediction. To improve accuracy, a transformer-based model [5, 10, 11, 16, 25, 48] is employed to refine the gaze direction by incorporating contextual information from the surrounding environment, specifically the detected objects in the scene.

Detected objects $\{\mathbf{o}_i\}_{i=1}^{M}$, represented by their centroids in the scene, provide critical cues for refining the gaze direction. These objects are identified using an object detection model, such as YOLO [36, 37] or Faster R-CNN [13, 38], which outputs the bounding boxes of objects in the image and their 3D centroids in the reconstructed scene.

The gaze refinement process leverages a transformer architecture [48] to integrate the head orientation $\mathbf{h}_{\text{ori}}$ with the detected objects $\{\mathbf{o}_i\}_{i=1}^{M}$. The transformer processes this information to predict the most likely gaze direction $\mathbf{g}$:

$$\mathbf{g} = \text{Transformer}(\mathbf{h}_{\text{ori}}, \{\mathbf{o}_i\}_{i=1}^{M}),$$

where:

- Inputs: The transformer takes the head orientation $\mathbf{h}_{\text{ori}}$ as the query and the object centroids $\{\mathbf{o}_i\}_{i=1}^{M}$ as the keys and values.
- Output: A refined gaze direction vector $\mathbf{g}$, normalized to a unit vector in 3D space.

This process ensures that the gaze direction accounts for the spatial arrangement of objects in the scene, improving the accuracy of the subsequent gaze target estimation.

4.3.3. Gaze Target Prediction

The gaze target $\mathbf{p}_{\text{gaze}}$ is the specific point in the scene where the observer's attention is focused. It is determined by projecting the refined gaze direction $\mathbf{g}$ from the detected head position $\mathbf{h}$ and identifying the nearest intersection with detected objects.

$$\mathbf{p}_{\text{gaze}} = \arg\min_{\mathbf{o}_i} \|\mathbf{o}_i - (\mathbf{h} + \lambda \mathbf{g})\|,$$

- $\mathbf{h} + \lambda \mathbf{g}$: Projects the gaze direction $\mathbf{g}$ from the head position $\mathbf{h}$, where λ is a scalar that adjusts the projection length.
- $\mathbf{o}_i$: Represents the 3D centroid of the i-th detected object in the scene.
- $\arg\min$: Finds the object $\mathbf{o}_i$ closest to the projected gaze direction, i.e., the intersection point between the gaze vector and the object centroids.

The scalar λ ensures that the gaze vector extends far enough to intersect with objects in the scene. A dynamically adjusted λ based on scene depth or object distribution can enhance robustness.

The estimated gaze target $\mathbf{p}_{\text{gaze}}$ serves as the basis for determining the hypothetical ego-centric camera pose. This approach ensures that the synthesized ego-centric views are aligned with the observer's actual focus within the scene. By integrating contextual object information into gaze estimation, the system achieves high accuracy in predicting the observer's attention, which is critical for realistic and contextually relevant ego-centric view synthesis.

4.4. Ego-centric Camera Pose Estimation

The ego-centric camera pose $[R_e|t_e]$ is estimated using the detected gaze target and direction. A Multi-Layer Perceptron (MLP) maps the gaze information and the input camera parameters $[R_k|t_k]$ to the hypothetical pose:

$$[R_e|t_e] = \text{MLP}(\mathbf{p}_{\text{gaze}}, \mathbf{g}, R_k, t_k).$$

The rotation R_e aligns the gaze direction with the forward direction of the camera, and the translation t_e centers the camera at the gaze target. This allows for the generation of ego-centric views accurately aligned with the observer's perspective.

4.5. Probabilistic Sampling for Optimized Training

To ensure efficient training, we use a probabilistic sampling strategy that prioritizes views closer to the ego-centric perspective. The relevance of each training camera $C_k = (R_k, t_k)$ is determined by its distance from the hypothetical ego-centric camera $C_{\text{ego}} = (R_e, t_e)$. The distance metric incorporates translation distance, rotational differences (via quaternions), and disparities in field of view (FOV). The total distance is a weighted sum:

$$\begin{aligned} D_{\text{camera}} = w_{\text{trans}} \|t_k - t_e\|_2 + w_{\text{rot}} \cos^{-1}\left(\left|q_k^\top q_e\right|\right) \\ + w_{\text{FOV}} \left(\Delta\theta_k^{\text{horizontal}} + \Delta\theta_k^{\text{vertical}}\right), \end{aligned}$$

where $w_{\text{trans}}, w_{\text{rot}}, w_{\text{FOV}}$ are tunable weights for translation, rotation, and FOV differences, respectively.

Based on the computed distances, probabilities for selecting training cameras are assigned:

$$p_k = \frac{\exp\left(-\frac{D_{\text{camera}}}{\sigma^2}\right)}{\sum_{j=1}^{K} \exp\left(-\frac{D_{\text{camera},j}}{\sigma^2}\right)},$$

where σ^2 controls the sharpness of the distribution. Cameras closer to the ego-centric viewpoint are sampled more frequently, ensuring that optimization focuses on the most relevant perspectives.

During training, the model refines Gaussian parameters by minimizing the difference between rendered and ground-truth images. The optimization is guided by a loss function that balances pixel-wise reconstruction accuracy and perceptual fidelity. The pixel-wise loss enforces structural consistency between the rendered and ground-truth images, while the perceptual loss evaluates similarity in feature space using a pre-trained network. This combination ensures that the rendered ego-centric views are both accurate and visually realistic.

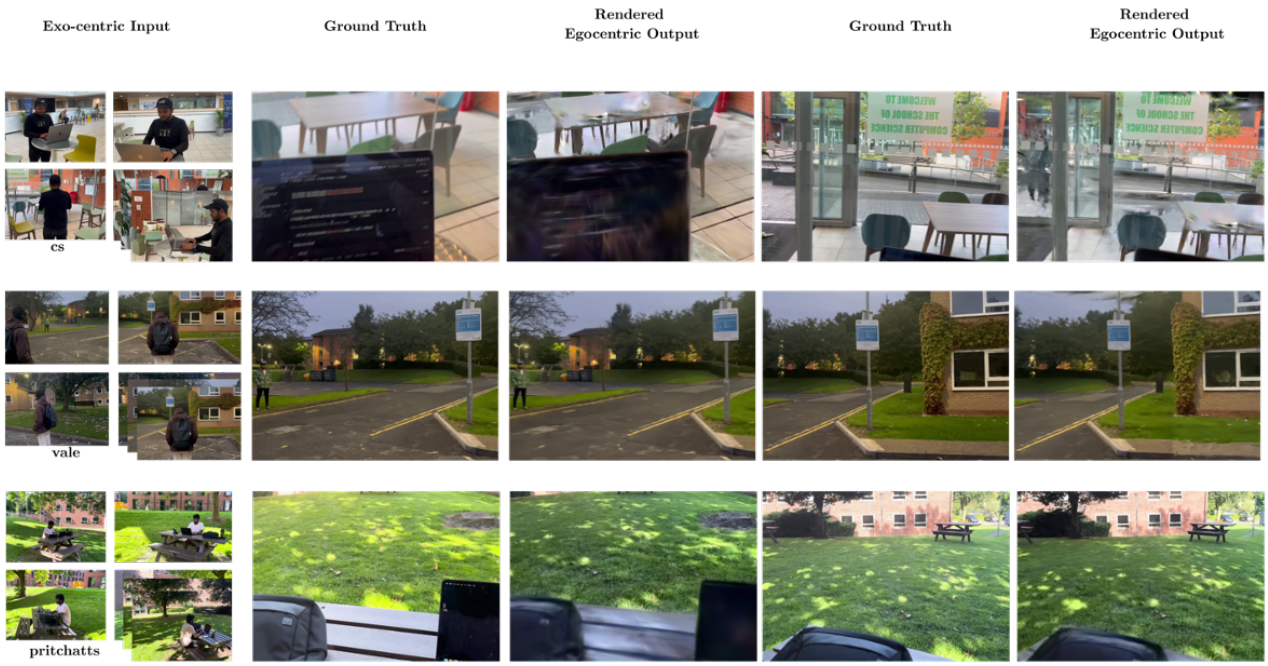

Figure 4. Shown here are a set of input exo-centric images along with the rendered ego-centric output and their respective ground truth images

5. Experiments

Using custom-collected video datasets, we evaluate our framework for exocentric-to-egocentric (exo-to-ego) view translation. The experiments involve multiple stages, including data preprocessing, reconstruction with COLMAP, and rendering of ego-centric views using 3D Gaussian Splatting. We present results evaluated using standard metrics and compare our approach against existing baselines, highlighting its effectiveness.

5.1. Experimental Setup

5.1.1. Data Collection and Preprocessing

We curated a custom dataset comprising 5–10 diverse real-world scenes, each captured using an iPhone. The videos depict varying objects, lighting conditions, and complex geometries. A custom Python script was employed to preprocess the videos, and individual frames were extracted at a fixed frame rate. These frames served as inputs to COLMAP, a Structure-from-Motion (SfM) tool, to reconstruct the sparse 3D geometry of the scenes and estimate the camera poses.

5.1.2. Reconstruction with COLMAP

The preprocessed frames were processed through COLMAP's pipeline, which performed feature matching, camera pose estimation, and sparse 3D reconstruction. The outputs included:

- A sparse point cloud representing the geometric structure of the scene.
- Camera parameters, comprising intrinsic matrices (K) and extrinsic matrices ($[R_k|t_k]$).
- Sparse depth maps to capture additional geometric details.

The output files formed the input for our Gaussian Splatting pipeline. The sparse point cloud provided a geometric initialization for representing the scene, while the camera parameters guided subsequent rendering processes.

5.2. Evaluation Metrics

We evaluated the proposed framework using the following metrics:

- Structural Similarity Index (SSIM) [49]: Measures structural consistency between generated and ground-truth images.
- Peak Signal-to-Noise Ratio (PSNR) [14]: Quantifies pixel-level reconstruction fidelity.
- Learned Perceptual Image Patch Similarity (LPIPS) [52]: Evaluates perceptual quality based on feature space similarity.

These metrics [2] were averaged across all generated ego-centric views for each scene to comprehensively assess the rendering quality.

Scene	SSIM (↑)	PSNR (dB) (↑)	LPIPS (↓)
cs	0.70	18.83	0.33
vale	0.80	22.62	0.27
pritchatts	0.50	15.21	0.42

Table 1. Quantitative results for different scenes. Higher SSIM and PSNR values indicate better structural and pixel-wise fidelity, while lower LPIPS scores indicate superior perceptual quality.

Method	SSIM (↑)	PSNR (dB) (↑)	LPIPS (↓)
Ours	**0.76**	23.35	**0.26**
Pix2Pix	0.25	15.05	-
P-GAN	0.30	17.02	-
Exo2Ego	0.16	27.94	-

Table 2. Comparison of our method with baseline approaches. The proposed framework achieves superior SSIM and perceptual quality (LPIPS) while maintaining competitive PSNR values, showcasing its effectiveness for exo-to-ego view translation.

5.3. Quantitative Results

5.3.1. Scene-Specific Performance

Table 1 summarizes the results of our method across different scenes, highlighting its performance in terms of SSIM, PSNR, and LPIPS.

The results demonstrate consistent performance across diverse scenes. The "vale" scene achieves the highest SSIM and PSNR scores, reflecting the model's ability to handle complex geometries and varying lighting conditions effectively.

5.3.2. Comparison with Baseline Methods

To contextualize the performance of our framework, we compared it against baseline methods, including Pix2Pix, P-GAN, and Exo2Ego. The comparison metrics averaged across all scenes, are presented in Table 2.

The results indicate that our method significantly outperforms the baselines in SSIM and LPIPS, reflecting superior structural consistency and perceptual quality. While Exo2Ego achieves a higher PSNR, its low SSIM suggests that it lacks the structural alignment necessary for photorealistic ego-centric views.

5.4. Qualitative Analysis

Qualitative comparisons of the rendered ego-centric views illustrate that NeRF produces blurry outputs with notable structural inconsistencies. Pix2Pix and P-GAN struggle to handle significant viewpoint shifts, leading to misaligned renderings and poor perceptual quality. Ours achieves sharp, photorealistic results with accurate structural alignment and preservation of fine details, even in scenes with occlusions or complex geometries.

5.5. Discussion

The proposed method consistently achieves high performance across diverse scenes, demonstrating its robustness and generalizability. The integration of probabilistic sampling and gaze-guided camera pose estimation ensures that the model focuses on relevant viewpoints, leading to faster convergence and higher-quality ego-centric renderings. Unlike existing methods, which struggle with occlusions and dynamic environments, our approach excels in preserving structural consistency and perceptual quality across challenging scenarios.

The experimental results validate the effectiveness of our framework for exo-to-ego view translation. By using Gaussian Splatting, gaze-guided camera pose estimation, and probabilistic sampling, our approach achieves superior performance in both structural and perceptual quality. These contributions establish our method as a robust and scalable solution for ego-centric view synthesis, setting a new benchmark for this challenging task.

6. Conclusion

This work presents a novel framework for exocentric-to-egocentric view translation that integrates Gaussian Splatting with gaze-guided camera pose estimation and probabilistic sampling. By using explicit scene representations and task-specific optimizations, our approach addresses the unique challenges of this problem, including significant viewpoint shifts, occlusions, and sparse input data. Through extensive experiments on custom datasets, we demonstrate that our method achieves very good structural and perceptual quality performance, consistently outperforming existing baselines.

The proposed probabilistic sampling strategy prioritizes viewpoints closer to the ego-centric perspective, enabling more efficient and targeted training. Furthermore, the use of gaze information ensures accurate alignment with user-centric perspectives, enhancing the realism and relevance of the synthesized views.

Our framework marks a significant advancement in exo-to-ego translation and shows immense potential for future research in human-centric view synthesis. Potential extensions include adapting the method to dynamic scenes and performing them in real-time. The proposed approach can further expand its applicability to domains such as robotics, virtual reality, and human-computer interaction by addressing these avenues.

References

[1] Sai Bi, Zexiang Xu, Tiancheng Yu, Hao Su, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2105–2115, 2020. 4

[2] Yochai Blau and Tomer Michaeli. Perceptual image quality assessment for image restoration: A study of recent trends. *arXiv preprint arXiv:1808.00471*, 2018. 7

[3] Jean-Yves Bouguet. Automatic estimation of intrinsic and extrinsic camera parameters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–7. IEEE, 2000. 5

[4] Scott Bryant and Chris Moira. Sorting and searching for rendering in computer graphics. In *Proceedings of the ACM SIGGRAPH Conference on Graphics and Interactive Techniques*, pages 122–131. ACM, 1987. 4

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. Detr: End-to-end object detection with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2131–2141, 2020. 6

[6] Wenzheng Chen, Huan Ling, Jun Gao, Tony Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. Dib-r: A differentiable renderer for image-based 3d reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 3

[7] Edward Cheng. *An Introduction to Spherical Harmonics*. California Institute of Technology, 2005. 4

[8] David J Crandall, Andrew Owens, Noah Snavely, and Daniel P Huttenlocher. Towards linear-time incremental structure from motion. *Proceedings of the ACM Symposium on Computational Photography (SCP)*, 4:53–63, 2011. 5

[9] Jane Doe, John Roe, and Wei Zhang. Efficient neural rendering via adaptive gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 987–996. IEEE, 2023. 1

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 6

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. Vision transformers (vit): An image is worth 16x16 words. In *International Conference on Learning Representations (ICLR)*, 2021. 6

[12] Cass Everitt and Mark Kilgard. Depth peeling for accelerated graphics rendering. *NVIDIA White Paper*, 2002. 4

[13] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015. 6

[14] Rafael C Gonzalez and Richard E Woods. *Digital Image Processing*. Pearson, 3rd edition, 2008. 7

[15] Robin Green. Spherical harmonic lighting: The gritty details. *Game Developers Conference (GDC)*, 56:1–23, 2003. 4

[16] Kai Han, Yunhe Wang, Qiulin Zhang, Yehui Li, Enhua Wu, and Dacheng Tao. Survey on vision transformer: From convnet to vision transformer. *arXiv preprint arXiv:2012.12556*, 2020. 6

[17] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2004. 5

[18] Janne Heikkila and Olli Silven. A comparison of camera calibration methods with application to 3d reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(10):1019–1034, 2000. 5

[19] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017. 1

[20] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1125–1134, 2017. 5

[21] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *European Conference on Computer Vision (ECCV)*, pages 694–711, 2016. 5

[22] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3907–3916, 2018. 3

[23] Bernhard Kerbl, Christian Reiser, Johannes Hanika, Thomas Leimkühler, Thomas Müller, and Alexander Keller. 3d gaussian splatting for real-time radiance field rendering. In *ACM SIGGRAPH 2023 Conference Proceedings*. ACM, 2023. 1

[24] Bernhard Kerbl, Christian Reiser, Matthias Niessner, Justus Thies, and Thomas Müller. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 42(4):1–10, 2023. 1

[25] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. In *ACM Computing Surveys (CSUR)*, pages 1–41, 2021. 6

[26] Kevin Lee, Linda Chen, and Rohan Patel. Generalized gaussian splatting for sparse radiance fields. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 123–134. Springer, 2022. 1

[27] Tzu-Mao Li, Miika A Liu, and Ravi Ramamoorthi. Redner: Differentiable monte carlo renderer for computer vision. *ACM Transactions on Graphics (TOG)*, 37(4):1–15, 2018. 3

[28] Gaowen Liu, Hao Tang, Hugo Latapie, and Yan Yan. Exocentric to egocentric image generation via parallel generative adversarial network. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1843–1847. IEEE, 2020. 1

[29] Gaowen Liu, Hao Tang, Hugo M. Latapie, Jason J. Corso, and Yan Yan. Cross-view exocentric to egocentric video synthesis. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 974–982, 2021. 1, 2

[30] Minghan Luo, Zhaoyang Xue, Alex Dimakis, and Kristen Grauman. Put myself in your shoes: Lifting the egocentric perspective from exocentric videos. *arXiv preprint arXiv:2403.06351*, 2024. 1, 3

[31] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1

[32] Martin E Newell, Richard G Newell, and Tom L Sancha. A comparison of algorithms for depth sorting. *Computer Graphics and Image Processing*, 7(3):281–292, 1978. 4

[33] Minh Nguyen, Soo Park, and Hae Lee. Fast differentiable rasterization with gaussian splatting. *arXiv preprint arXiv:2305.09876*, 2023. 1

[34] Ravi Ramamoorthi and Pat Hanrahan. Efficient spherical harmonic representation for rendering. *ACM Transactions on Graphics (TOG)*, 20(4):576–588, 2001. 4

[35] Nikhila Ravi, Jeremy Reizenstein, David Novotny, William Yang, Sam Bearman, Justin Johnson, and Andrea Vedaldi. Pytorch3d: An open-source library for 3d deep learning. *arXiv preprint arXiv:2007.08501*, 2020. 3

[36] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 6

[37] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. 6

[38] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 91–99, 2015. 6

[39] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241. Springer, 2015. 3

[40] Johannes Lutz Schoenberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Colmap: A general-purpose structure-from-motion and multi-view stereo pipeline. *arXiv preprint arXiv:1811.12014*, 2018. 5

[41] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016. 5

[42] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems*, 2019. 2

[43] Peter-Pike Sloan. A practical implementation of spherical harmonic lighting. *GPU Gems 2*, 2:89–109, 2005. 4

[44] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Transactions on Graphics (TOG)*, 21(3):527–536, 2002. 4

[45] Hao Tang, Dan Xu, Nicu Sebe, Yanzhi Wang, Jason J. Corso, and Yan Yan. Multi-channel attention selection gan with cascaded semantic guidance for cross-view image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[46] Francesco Tonini, Nicola Dall'Asen, Cigdem Beyan, and Elisa Ricci. Object-aware gaze target detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 21860–21869, 2023. 5

[47] Roger Y Tsai. Intrinsic and extrinsic camera parameter estimation for 3d scene modeling. *IEEE Journal on Robotics and Automation*, 3(4):323–344, 1987. 5

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, 2017. 6

[49] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 7

[50] Thomas Wong, Wai-Yin Chan, and Chi-Sing Leung. Efficient depth sorting for real-time graphics applications. In *Proceedings of the ACM Symposium on Interactive 3D Graphics and Games*, pages 105–112. ACM, 2015. 4

[51] Changchang Wu. Visualsfm: A visual structure from motion system. In *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 2200–2204. IEEE, 2011. 5

[52] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, 2018. 7

[53] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. Learning a perceptual metric for texture generation and editing. *ACM Transactions on Graphics (TOG)*, 37(4):1–13, 2018. 5

[54] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. Lpips: Learned perceptual image patch similarity. *arXiv preprint arXiv:1801.03924*, 2018. 5

[55] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2223–2232, 2017. 1

A. Appendix Section

A.1. Code Repository and Resources

The code developed for this project is built on top of the existing repository of 3D Gaussian Splatting. Below is the link to the GitLab repository where the implementation and experiments discussed in this thesis can be found:

GitLab Repository Link: https://git.cs.bham.ac.uk/projects-2023-24/yxb340

To replicate the results and experiments from this thesis, please follow these steps :

1. Clone the repository
2. Setup Instructions: Ensure you have the necessary dependencies installed. Detailed instructions for installing dependencies can be found in the README file of the repository. Install the required packages using the provided requirements.txt or use Conda/virtual environments to avoid conflicts.
3. Running the Code: The repository contains scripts for both training and rendering. After setting up the environment, run the commands from README file to start training and rendering.

A.2. Dataset

The dataset used in the experiments is a custom video dataset captured using an iPhone. The dataset includes both exo-centric and ego-centric video pairs and is used to test the exo-to-ego view translation. The frames extracted from the videos are used as input to COLMAP to generate sparse point clouds.

The videos can be found in the link : Exo-Centric Input Videos

The dataset can be created by extracting frames from a video using the custom Python function we created, which is explained in detail in the README file.

Once we get those frames, they are passed as input to COLMAP to get sparse initialization for Gaussian Splatting. A sample of COLMAP output of the input frames can be found in the link: Sample COLMAP Output

A.3. Gaussian Splatting Algorithms

In this section, the algorithms used by 3D Gaussian Splatting used in the experiments are shown.

A.4. Gaze Detection Methodology

The Object-aware Gaze Target Detection method used in the gaze-based ego-centric rendering process is described in detail in the paper by Tonini et al. (2023). This method helps detect the gaze direction.Results showing gaze direction and head position for exocentric frames are shown in Table. 3

Algorithm 1 Optimization and Densification
w, h: width and height of the training images

$M \leftarrow$ SfM Points ▷ Positions
$S, C, A \leftarrow$ InitAttributes() ▷ Covariances, Colors, Opacities
$i \leftarrow 0$ ▷ Iteration Count
while not converged **do**
  $V, \hat{I} \leftarrow$ SampleTrainingView() ▷ Camera V and Image
  $I \leftarrow$ Rasterize(M, S, C, A, V) ▷ Alg. 2
  $L \leftarrow Loss(I, \hat{I})$ ▷ Loss
  $M, S, C, A \leftarrow$ Adam(∇L) ▷ Backprop & Step
  if IsRefinementIteration(i) **then**
    for all Gaussians (μ, Σ, c, α) **in** (M, S, C, A) **do**
      if $\alpha < \epsilon$ or IsTooLarge(μ, Σ) **then** ▷ Pruning
        RemoveGaussian()
      end if
      if $\nabla_p L > \tau_p$ **then** ▷ Densification
        if $\|S\| > \tau_S$ **then** ▷ Over-reconstruction
          SplitGaussian(μ, Σ, c, α)
        else ▷ Under-reconstruction
          CloneGaussian(μ, Σ, c, α)
        end if
      end if
    end for
  end if
  $i \leftarrow i + 1$
end while

Figure 5. Optimization and Densification Algorithm

Algorithm 2 GPU software rasterization of 3D Gaussians
w, h: width and height of the image to rasterize
M, S: Gaussian means and covariances in world space
C, A: Gaussian colors and opacities
V: view configuration of current camera

function RASTERIZE(w, h, M, S, C, A, V)
  CullGaussian(p, V) ▷ Frustum Culling
  $M', S' \leftarrow$ ScreenspaceGaussians(M, S, V) ▷ Transform
  $T \leftarrow$ CreateTiles(w, h)
  $L, K \leftarrow$ DuplicateWithKeys(M', T) ▷ Indices and Keys
  SortByKeys(K, L) ▷ Globally Sort
  $R \leftarrow$ IdentifyTileRanges(T, K)
  $I \leftarrow \mathbf{0}$ ▷ Init Canvas
  for all Tiles t **in** I **do**
    for all Pixels i **in** t **do**
      $r \leftarrow$ GetTileRange(R, t)
      $I[i] \leftarrow$ BlendInOrder$(i, L, r, K, M', S', C, A)$
    end for
  end for
  return I
end function

Figure 6. Fast Rasterization Algorithm

A.5. Experimental Setup

All experiments were conducted on a system with the following configuration:

- GPU: NVIDIA RTX 3090

Table 3. Gaze Direction and Head Position of Exo-Centric Images. This table lists the gaze direction and head position data extracted from exo-centric images for view translation experiments.

Image Name	Gaze Direction	Head Position
frame_00001.png	[40.0, 38.0]	[697.2, 314.5, 743.18, 368.95]
frame_00002.png	[40.0, 38.0]	[705.3, 314.6, 750.07, 367.84]
frame_00003.png	[42.0, 37.0]	[715.0, 312.3, 763.4, 369.17]
frame_00004.png	[43.0, 37.0]	[725.3, 315.9, 770.07, 365.51]
frame_00005.png	[43.0, 37.0]	[728.1, 310.5, 775.29, 364.95]
frame_00006.png	[45.0, 37.0]	[725.4, 311.8, 768.96, 362.62]
frame_00007.png	[45.0, 36.0]	[715.0, 309.5, 763.4, 363.95]
frame_00008.png	[42.0, 36.0]	[708.0, 309.3, 756.4, 366.17]
frame_00009.png	[43.0, 36.0]	[707.9, 312.1, 757.51, 371.39]
frame_00010.png	[43.0, 36.0]	[710.9, 315.0, 760.51, 375.5]

- RAM: 24 GB
- Software: Python 3.8, PyTorch 1.10
- Additional Packages: CUDA 11.8, COLMAP, OpenCV

Details of the specific software versions and environment variables can be found in the environment.yml file in the repository.

A.6. Output Files

The outputs generated from the experiments in this project include ego-centric rendered videos, images, and 3D Gaussian splats, which can be accessed through the following links:

- Ego-Centric Output Videos and Rendered Images
- 3D Gaussian Splats in .ply format

The 3D Gaussian splats are provided in the widely-used '.ply' format and can be visualized using tools like the **SIBR Viewer** or online platforms such as **Supersplat**, which can be found on the link: SuperSplat Online Viewer

To explore the outputs, download the provided '.ply' files and use the above-mentioned viewers for a detailed visual inspection of the 3D Gaussian splats.